



UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE ESTUDOS SOCIAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM ECONOMIA

FELIPE SOARES ESTEVES

MANUAL TÉCNICO DE CONSTRUÇÃO DE BASES DE DADOS APLICADAS

MANAUS – AM

2026

FELIPE SOARES ESTEVES

MANUAL TÉCNICO DE CONSTRUÇÃO DE BASES DE DADOS APLICADAS

Produto Técnico-Tecnológico apresentado ao Programa de Pós-Graduação em Economia da Universidade Federal do Amazonas (PPGECO/UFAM), como requisito parcial para obtenção do título de Mestre em Economia.

Orientador: Prof. Dr. Yunier Sarmiento Ramírez.

MANAUS – AM

2026

Ficha Catalográfica

Elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

E79m Esteves , Felipe Soares
Manual Técnico de Construção de Bases de Dados Aplicadas [recurso eletrônico] / Felipe Soares Esteves , Yunier Sarmiento Ramírez.
1 recurso eletrônico (106 f.): il., color.

Produto educacional (Programa de Pós-Graduação em Economia) -
Universidade Federal do Amazonas, Manaus, 2026.

1. manual técnico. 2. bases de dados. 3. linguagem r. 4. inteligência artificial. I. Ramírez, Yunier Sarmiento. II. Universidade Federal do Amazonas. Programa de Pós-Graduação em Economia. III. Título

SUMÁRIO

INTRODUÇÃO	1
DESCRIÇÃO GERAL DO PRODUTO.....	3
BASE TEÓRICA UTILIZADA.....	5
RELEVÂNCIA DO PRODUTO.....	7
DOCUMENTOS COMPROBATÓRIOS E EVIDÊNCIAS	9
1. VISÃO GERAL DO PROCESSO	10
1.1 FLUXO GERAL E ETAPAS DO PROCESSO DE CONSTRUÇÃO DA BASE	10
1.2 TIPOS DE DADOS E FONTES	11
1.3 ESTRUTURA DA BASE DE DADOS	13
2. REQUISITOS	14
2.1 SOFTWARES E FERRAMENTAS UTILIZADAS.....	14
2.2 PACOTES DA LINGUAGEM R	15
2.3 SCRIPT DE CONFIGURAÇÃO INICIAL.....	16
3. ESTRUTURA DO PROJETO.....	19
3.1 ORGANIZAÇÃO DO AMBIENTE DE TRABALHO E ESTRUTURA DE DIRETÓRIOS	19
3.2 DADOS DE ENTRADA DO PROJETO	21
3.3 PARÂMETROS GLOBAIS DO PROJETO	23
4. DADOS CLIMÁTICOS.....	26
4.1 DADOS CLIMÁTICOS UTILIZADOS	26
4.1.1 Estrutura esperada dos arquivos climáticos	27
4.2 EXTRAÇÃO ESPACIAL DAS VARIÁVEIS CLIMÁTICAS	27
4.3 AGREGAÇÃO TEMPORAL DOS DADOS CLIMÁTICOS	30
4.4 CONSTRUÇÃO DA TEMPERATURA MÉDIA ANUAL.....	31

4.5	CONSOLIDAÇÃO DO PAINEL CLIMÁTICO	33
4.6	VERIFICAÇÕES E VALIDAÇÃO DOS DADOS CLIMÁTICOS	35
5.	DADOS ECONÔMICOS	36
5.1	DADOS ECONÔMICOS UTILIZADOS.....	36
5.1.1	Estrutura esperada dos arquivos econômicos.....	37
5.2	CONSTRUÇÃO DO PAINEL FISCAL MUNICIPAL	38
5.3	INTEGRAÇÃO DE PIB, POPULAÇÃO E ÍNDICE DE PREÇOS.....	40
5.4	DEFLACIONAMENTO DAS VARIÁVEIS MONETÁRIAS	42
5.5	CONSTRUÇÃO DAS VARIÁVEIS PER CAPITA	44
5.6	CONSOLIDAÇÃO DO PAINEL ECONÔMICO FINAL	46
5.7	VERIFICAÇÕES E VALIDAÇÃO DOS DADOS ECONÔMICOS.....	47
6.	INTEGRAÇÃO E AUDITORIA	49
6.1	INTEGRAÇÃO DAS BASES CLIMÁTICAS E ECONÔMICAS.....	49
6.2	GERAÇÃO DO DICIONÁRIO DE VARIÁVEIS	52
6.3	AUDITORIA E VALIDAÇÃO DA BASE FINAL	54
6.4	BASE FINAL CONSOLIDADA	57
7.	ADAPTAÇÃO E REUTILIZAÇÃO DO PIPELINE.....	59
7.1	ADAPTAÇÃO PARA OUTROS RECORTES ESPACIAIS	59
7.2	INCLUSÃO DE NOVAS VARIÁVEIS CLIMÁTICAS.....	60
7.3	INCLUSÃO DE NOVOS INDICADORES ECONÔMICOS	61
7.4	ALTERAÇÃO DO PERÍODO TEMPORAL DO PAINEL	62
7.5	RECOMENDAÇÕES PARA REUTILIZAÇÃO DO PIPELINE	63
	CONSIDERAÇÕES FINAIS	65
	REFERÊNCIAS.....	67
	APÊNDICE A – Script de configuração inicial do projeto (00_config.R)	68
	APÊNDICE B – Script de extração das variáveis climáticas	
	(01_extract_one_variable.R)	71

<i>APÊNDICE C – Script de construção da temperatura média anual</i>	
<i>(02_build_tmean.R)</i>	76
<i>APÊNDICE D – Script de consolidação do painel climático</i>	
<i>(03_merge_climate_all.R)</i>	79
<i>APÊNDICE E – Script de construção do painel fiscal municipal</i>	
<i>(04_econ_build_fiscal_panel.R)</i>	82
<i>APÊNDICE F – Script de integração econômica, deflacionamento e cálculo de variáveis per capita (05_econ_add_pib_pop_deflate_and_finalize.R)</i>	
	89
<i>APÊNDICE G – Script de integração final das bases</i>	
<i>(06_build_panel_master_2014_2024.R)</i>	94
<i>APÊNDICE H – Script de auditoria e validação estrutural da base consolidada</i>	
<i>(07_audit_panel_master_2014_2024.R)</i>	99
<i>APÊNDICE I – Evidências dos produtos gerados pelo pipeline</i>	
	105

INTRODUÇÃO

A construção de bases de dados aplicadas constitui uma etapa fundamental para o desenvolvimento de pesquisas quantitativas, formulação de políticas públicas e produção de evidências empíricas em diferentes áreas do conhecimento. O crescente volume de informações disponibilizadas por instituições públicas e organismos nacionais e internacionais tem ampliado as possibilidades analíticas, mas também tem imposto desafios relacionados à extração, padronização, integração e validação de dados provenientes de múltiplas fontes.

No contexto da pesquisa aplicada, especialmente em estudos econômicos e socioambientais, a construção de bases de dados frequentemente envolve a combinação de informações produzidas em diferentes formatos, escalas espaciais e frequências temporais, exigindo procedimentos sistemáticos capazes de garantir consistência, rastreabilidade e reprodutibilidade dos resultados. Nesse cenário, a utilização de rotinas computacionais automatizadas e pipelines de processamento de dados tem se consolidado como estratégia relevante para reduzir erros operacionais, ampliar a transparência metodológica e facilitar a replicação dos procedimentos adotados.

Foi nesse contexto que se desenvolveu o presente Produto Técnico-Tecnológico (PTT), concebido a partir da experiência acumulada na construção da base de dados utilizada na dissertação de mestrado intitulada *"Impactos Econômicos das Mudanças Climáticas nos Municípios do Estado do Amazonas: uma Perspectiva Longitudinal"*, desenvolvida no âmbito do Programa de Pós-Graduação em Economia da Universidade Federal do Amazonas (PPGECO/UFAM). A elaboração da referida pesquisa demandou a integração de informações econômicas, fiscais, demográficas e climáticas provenientes de diferentes bases públicas, evidenciando a necessidade de sistematização e documentação das rotinas computacionais empregadas.

Sob o ponto de vista científico e tecnológico, o produto busca contribuir para a disseminação de práticas associadas à reprodutibilidade computacional, à organização de pipelines de dados e à utilização de ferramentas digitais aplicadas à pesquisa quantitativa. Do ponto de vista social e institucional, o manual pode auxiliar estudantes, pesquisadores, técnicos e gestores na construção de bases de dados organizadas, auditáveis e passíveis de replicação, contribuindo para o fortalecimento da produção de evidências e para o

aprimoramento de estudos e diagnósticos voltados ao planejamento e à formulação de políticas públicas.

Entre as características centrais do produto destacam-se sua abordagem prática, a organização modular das rotinas em linguagem R (2024), o caráter reproduzível dos procedimentos descritos, a possibilidade de adaptação para diferentes contextos de pesquisa e a incorporação de ferramentas de Inteligência Artificial como suporte auxiliar à implementação e adaptação das rotinas computacionais. O enfoque do manual está na aplicação prática de conhecimentos teórico-metodológicos relacionados à construção de bases de dados, utilizando a linguagem R como ferramenta operacional. Assim, não se busca detalhar exaustivamente a sintaxe da linguagem ou o funcionamento individual dos pacotes empregados, mas orientar sua utilização no contexto do tratamento, integração e validação de dados aplicados. Complementarmente, são apresentadas possibilidades de utilização de ferramentas de Inteligência Artificial, como ChatGPT (2026) e Gemini (2026), como suporte auxiliar à interpretação de erros, adaptação de scripts e resolução de problemas operacionais ao longo da execução das rotinas.

Neste manual você aprenderá a:

- ✓ Organizar projetos de dados em R
- ✓ Estruturar pipelines reproduzíveis
- ✓ Integrar bases climáticas e econômicas
- ✓ Construir painéis municipais anuais
- ✓ Automatizar auditorias de dados
- ✓ Adaptar pipelines para novos projetos

DESCRIÇÃO GERAL DO PRODUTO

O presente Produto Técnico-Tecnológico consiste em um manual técnico de caráter didático destinado à orientação do processo de construção, organização, integração e validação de bases de dados aplicadas por meio da linguagem R. O produto foi concebido para apoiar atividades de ensino, pesquisa e desenvolvimento institucional, oferecendo orientações operacionais voltadas à implementação de pipelines reproduzíveis de processamento de dados.

O produto materializa-se na forma de um documento técnico estruturado em capítulos sequenciais, contendo orientações metodológicas, fluxos operacionais, descrição de procedimentos, exemplos de aplicação, recomendações para organização do ambiente computacional, exemplos de utilização de ferramentas de Inteligência Artificial e procedimentos de auditoria e validação das bases produzidas. Embora utilize a linguagem R como principal ferramenta computacional, o produto não tem como finalidade ensinar exaustivamente a sintaxe da linguagem ou apresentar descrições detalhadas de todos os pacotes empregados, mas orientar sua utilização no contexto da construção, integração, auditoria e validação de bases de dados aplicadas.

O processo de elaboração do produto teve origem nas rotinas computacionais desenvolvidas durante a construção da base de dados utilizada na dissertação de mestrado do autor. A partir dessas rotinas, procedeu-se à sistematização, documentação e adaptação dos procedimentos originalmente empregados na pesquisa, com vistas à sua transformação em material didático e instrumento de apoio à construção de novas bases de dados.

Para a concepção do produto foram utilizados como instrumentos metodológicos a análise documental das rotinas computacionais implementadas, a revisão e reorganização dos scripts desenvolvidos no âmbito da dissertação, bem como a documentação das etapas operacionais necessárias à construção do pipeline de dados. O processo envolveu ainda a estruturação de fluxos operacionais, elaboração de quadros explicativos, organização de exemplos práticos e desenvolvimento de orientações voltadas à reutilização e adaptação do pipeline.

O pipeline descrito no manual é composto por scripts independentes e integrados entre si, responsáveis pelas etapas de configuração do ambiente, extração e tratamento de dados climáticos, construção do painel econômico, integração das bases e auditoria estrutural da base consolidada. Embora utilize como referência aplicada a construção de

uma base municipal para os municípios do estado do Amazonas no período de 2014 a 2024, a estrutura proposta apresenta caráter modular e adaptável, podendo ser utilizada em diferentes áreas do conhecimento, contextos institucionais e escalas territoriais.

Além de sua aplicação no contexto acadêmico, o produto pode ser utilizado como instrumento de apoio em atividades de pesquisa aplicada, monitoramento socioeconômico, elaboração de diagnósticos territoriais e desenvolvimento de estudos técnicos em instituições públicas e privadas, ampliando o potencial de disseminação e reutilização das rotinas descritas ao longo do manual.

BASE TEÓRICA UTILIZADA

A elaboração deste Produto Técnico-Tecnológico encontra-se diretamente vinculada à dissertação intitulada *Impactos Econômicos das Mudanças Climáticas nos Municípios do Estado do Amazonas: uma Perspectiva Longitudinal*, da qual constitui um dos produtos derivados. Nesse sentido, o manual compartilha o mesmo referencial teórico que fundamenta a pesquisa, especialmente no que se refere à necessidade de integração de informações econômicas, fiscais, demográficas e climáticas para a análise dos impactos econômicos das mudanças climáticas em escala municipal. Assim, o produto representa um desdobramento metodológico da dissertação, sistematizando os procedimentos empregados na construção da base de dados utilizada na investigação empírica.

Além da fundamentação substantiva oriunda da dissertação, o desenvolvimento do manual apoia-se em princípios relacionados à gestão e organização de dados científicos. A integração de informações provenientes de diferentes fontes, a padronização de procedimentos de tratamento de dados e a documentação sistemática das etapas analíticas constituem elementos essenciais para assegurar consistência, transparência e reutilização das informações produzidas. Nesse contexto, o produto dialoga com os princípios FAIR, propostos por Wilkinson et al. (2016), segundo os quais os dados científicos devem ser encontráveis, acessíveis, interoperáveis e reutilizáveis.

A estrutura do manual também se fundamenta nos princípios da reprodutibilidade computacional, entendida como a possibilidade de reproduzir procedimentos analíticos a partir do acesso aos dados, códigos e documentação metodológica empregados na pesquisa (Peng, 2011). Em estudos quantitativos que envolvem múltiplas etapas de extração, tratamento e integração de dados, a utilização de rotinas automatizadas e devidamente documentadas contribui para ampliar a transparência metodológica, reduzir erros operacionais e favorecer a replicação dos resultados.

Adicionalmente, o produto incorpora conceitos associados à organização de *pipelines* reproduzíveis de dados, compreendidos como fluxos estruturados de extração, transformação, integração e validação de informações. A adoção de scripts modulares, organizados sequencialmente e acompanhados de procedimentos de auditoria, busca garantir rastreabilidade, padronização e flexibilidade para adaptação do processo a diferentes contextos de pesquisa.

Por fim, o manual incorpora ferramentas de Inteligência Artificial como recurso complementar ao desenvolvimento computacional, empregadas como suporte auxiliar à interpretação de erros, adaptação de scripts e resolução de problemas operacionais. Todavia, a utilização dessas ferramentas não substitui a avaliação crítica do pesquisador, permanecendo sob sua responsabilidade a validação dos procedimentos e dos resultados produzidos.

RELEVÂNCIA DO PRODUTO

O manual elaborado apresenta elevado nível de complexidade técnico-metodológica, uma vez que sistematiza procedimentos relacionados à extração, tratamento, padronização, integração e auditoria de dados provenientes de múltiplas fontes públicas, contemplando informações climáticas, econômicas, fiscais e demográficas. O produto foi concebido a partir das necessidades metodológicas identificadas durante o desenvolvimento da dissertação de mestrado do autor, mantendo aderência à área de concentração em Economia Aplicada e Desenvolvimento da Amazônia e à linha de pesquisa Políticas Macroeconômicas e Desenvolvimento Regional do Programa de Pós-Graduação em Economia da Universidade Federal do Amazonas, ao fornecer instrumentos metodológicos aplicáveis à análise econômica regional e à produção de evidências para subsidiar estudos e políticas voltadas ao desenvolvimento amazônico.

Além disso, o manual organiza e documenta um conjunto de rotinas computacionais originalmente desenvolvidas para fins de pesquisa aplicada, transformando procedimentos técnicos dispersos em um material didático estruturado, reproduzível e passível de reutilização em diferentes contextos acadêmicos e institucionais.

No que se refere ao potencial inovador, o produto apresenta como principal diferencial a integração entre procedimentos tradicionais de construção de bases de dados e o uso complementar de ferramentas de Inteligência Artificial no apoio à implementação, adaptação e validação de rotinas computacionais. Embora existam materiais destinados ao ensino da linguagem R, o manual propõe uma abordagem aplicada, orientada à operacionalização de pipelines reproduzíveis de dados, priorizando a resolução de problemas concretos relacionados à integração de bases provenientes de diferentes fontes. Adicionalmente, a estrutura modular do pipeline, organizada em scripts independentes e acompanhada de procedimentos de auditoria automatizada, amplia a capacidade de adaptação do produto para diferentes temas de pesquisa, recortes territoriais e períodos de análise.

Quanto à aplicabilidade, o produto apresenta elevada potencialidade de utilização em atividades de ensino, pesquisa e desenvolvimento institucional. Sua estrutura permite que estudantes, pesquisadores, técnicos e gestores adaptem os procedimentos descritos para a construção de bases de dados aplicadas em diferentes áreas do conhecimento,

especialmente aquelas que demandam integração de informações socioeconômicas, ambientais e territoriais. Além do contexto acadêmico, o manual pode ser empregado por instituições públicas, centros de pesquisa e órgãos de planejamento interessados na organização e sistematização de dados para fins de monitoramento, diagnóstico e formulação de políticas públicas. A utilização de linguagem acessível, associada à descrição detalhada das etapas operacionais, favorece sua replicação e utilização por usuários com conhecimentos básicos em programação e manipulação de dados.

O impacto potencial do produto decorre de sua capacidade de ampliar a transparência, a padronização e a reprodutibilidade dos procedimentos de construção de bases de dados aplicadas. Ao sistematizar rotinas computacionais e disponibilizar orientações para sua adaptação e reutilização, o manual contribui para a redução do tempo necessário à organização de bases de dados, diminuição de erros operacionais e fortalecimento da qualidade metodológica das pesquisas desenvolvidas. Em perspectiva mais ampla, o produto pode apoiar a produção de evidências empíricas destinadas à análise econômica regional, ao monitoramento socioeconômico e ambiental e ao planejamento de políticas públicas, produzindo benefícios tanto no âmbito acadêmico quanto institucional.

DOCUMENTOS COMPROBATÓRIOS E EVIDÊNCIAS

A validação do presente Produto Técnico-Tecnológico está fundamentada no conjunto de evidências incorporadas ao próprio manual e em seus materiais complementares. Ao longo do documento são apresentados registros operacionais que evidenciam a concepção, implementação e aplicação prática do produto, incluindo fluxogramas, representações esquemáticas do pipeline, imagens ilustrativas da estrutura de diretórios, exemplos de entradas e saídas produzidas pelos scripts, além da descrição detalhada das etapas de extração, tratamento, integração e auditoria dos dados.

Adicionalmente, os Apêndices A a H disponibilizam os scripts completos utilizados na implementação do pipeline, documentando integralmente os procedimentos computacionais empregados na construção das bases de dados. Esses materiais permitem a reprodução, adaptação e validação das rotinas descritas, constituindo evidências objetivas do funcionamento operacional do produto.

Como complemento, o Apêndice I apresenta evidências dos produtos gerados durante a execução do pipeline, incluindo registros dos arquivos finais produzidos, relatórios de auditoria, dicionário de variáveis e registros de execução (*logs*), demonstrando o funcionamento efetivo das rotinas implementadas e a capacidade do produto de gerar bases consolidadas e auditáveis.

Destaca-se, ainda, que o manual foi desenvolvido a partir das rotinas computacionais efetivamente utilizadas na construção da base de dados empregada na dissertação de mestrado do autor, configurando uma aplicação prática do produto em contexto real de pesquisa. Dessa forma, o conjunto de evidências apresentado demonstra não apenas a viabilidade técnica do pipeline proposto, mas também sua aplicabilidade, reprodutibilidade e potencial de utilização em diferentes contextos acadêmicos e institucionais.

1. VISÃO GERAL DO PROCESSO

A construção de bases de dados aplicadas requer a organização sistemática de diferentes etapas operacionais, envolvendo desde a obtenção das informações até a consolidação de uma estrutura final adequada à análise quantitativa. Neste manual, esse processo é abordado de forma sequencial e reproduzível, com foco na implementação prática de rotinas de extração, tratamento, integração e validação de dados.

A abordagem adotada considera a utilização de dados provenientes de múltiplas fontes, com destaque para informações econômicas e climáticas utilizadas como referência aplicada na construção da base de dados apresentada ao longo do manual. Essa estrutura foi desenvolvida a partir da experiência empírica do autor na construção de uma base municipal integrada para o estado do Amazonas, servindo como modelo operacional para demonstração das etapas e procedimentos descritos.

Embora utilize um caso aplicado específico como referência, os procedimentos apresentados possuem caráter modular e adaptável, podendo ser utilizados em diferentes contextos de pesquisa e áreas do conhecimento que envolvam organização e integração de dados para fins analíticos.

Este capítulo apresenta uma visão geral do processo de construção da base de dados, abordando o fluxo operacional das etapas, os tipos de dados utilizados, as fontes de informação consideradas e a estrutura final da base consolidada.

1.1 FLUXO GERAL E ETAPAS DO PROCESSO DE CONSTRUÇÃO DA BASE

O processo de construção da base de dados inicia-se com a obtenção das informações a partir de fontes primárias, como bases públicas e arquivos digitais disponibilizados em diferentes formatos. Em seguida, os dados passam por etapas de tratamento e padronização, nas quais são realizadas operações de limpeza, organização, seleção de variáveis e compatibilização das estruturas utilizadas pelas diferentes fontes.

Na sequência, são construídas variáveis derivadas e realizadas agregações temporais, permitindo a organização dos dados em frequência anual e sua adequação para análises quantitativas. Posteriormente, as diferentes bases são integradas por meio de chaves comuns, como identificadores geográficos e temporais, possibilitando a consolidação das informações em uma única estrutura de dados em painel.

Por fim, são aplicados procedimentos de auditoria e controle de qualidade, incluindo verificações de consistência estrutural, identificação de valores ausentes, detecção de duplicidades e validação das chaves de integração, resultando na construção da base final consolidada, denominada `panel_master_2014_2024`.

A operacionalização do processo ocorre por meio de scripts independentes e integrados entre si, organizados conforme as etapas de configuração do ambiente, extração de dados, tratamento das informações, integração das bases e auditoria da estrutura final. Essa abordagem modular permite maior controle das etapas executadas, facilita a reprodução das rotinas e possibilita a adaptação do pipeline para diferentes projetos de dados.

A Figura 1 apresenta o fluxo integrado do processo de construção da base de dados, destacando as etapas operacionais empregadas em sua execução.

Figura 1 – Fluxo do processo



1.2 TIPOS DE DADOS E FONTES

A construção da base de dados apresentada neste manual baseia-se na integração de informações provenientes de diferentes fontes, com destaque para dados de natureza econômica e climática. A identificação adequada dessas fontes e a compreensão de suas características são fundamentais para a execução das etapas de extração, tratamento e integração descritas nos capítulos seguintes.

Os dados econômicos são obtidos a partir de bases públicas nacionais, com destaque para o Sistema de Informações Contábeis e Fiscais do Setor Público Brasileiro – SICONFI, mantido pela Secretaria do Tesouro Nacional (2026), e o Sistema IBGE de

Recuperação Automática – SIDRA, do Instituto Brasileiro de Geografia e Estatística – IBGE (2023). Essas bases disponibilizam informações fiscais, orçamentárias e socioeconômicas em diferentes níveis de agregação, permitindo a construção de indicadores relacionados à atividade econômica, à estrutura fiscal e à dinâmica populacional.

No que se refere aos dados climáticos, utiliza-se como fonte a base TerraClimate (2018), que disponibiliza variáveis ambientais em alta resolução espacial e frequência temporal mensal. Os dados extraídos são posteriormente agregados para frequência anual, permitindo sua integração com indicadores econômicos e fiscais organizados em estrutura de painel.

As variáveis climáticas consideradas incluem indicadores relacionados à precipitação, temperatura, evapotranspiração, déficit hídrico, umidade do solo e escoamento superficial, possibilitando a construção de bases integradas voltadas à análise quantitativa em diferentes contextos de pesquisa.

Para fins de sistematização, o quadro abaixo apresenta uma síntese dos tipos de dados considerados neste manual, com exemplos representativos de variáveis e suas respectivas fontes. Ressalta-se que a seleção das variáveis pode ser adaptada conforme o objetivo da análise, mantendo-se, contudo, a lógica de integração entre diferentes fontes e a padronização das informações para fins analíticos.

Tipos de dados e fontes utilizadas

Tipo de dado	Exemplos de variáveis	Fonte
Econômicos	Produto Interno Bruto (PIB), Receita	SIDRA/IBGE; SICONFI/STN
	Corrente Líquida, Despesa Corrente,	
	Fundo de Participação dos Municípios, População	
Climáticos	Precipitação, Temperatura,	TerraClimate
	Evapotranspiração, Déficit hídrico,	
	Umidade do solo, Escoamento superficial, Índice de seca (PDSI)	

1.3 ESTRUTURA DA BASE DE DADOS

A base de dados resultante do processo descrito neste manual apresenta estrutura do tipo painel, organizada a partir da combinação de unidades espaciais e temporais. No caso da aplicação utilizada como referência, as unidades espaciais correspondem aos municípios do Estado do Amazonas, identificados por códigos padronizados de sete dígitos (ibge7), enquanto a dimensão temporal é representada por períodos anuais, abrangendo o intervalo de 2014 a 2024.

Essa estrutura permite a organização sistemática das observações, em que cada linha representa uma unidade espacial em determinado período, enquanto cada coluna corresponde a uma variável de interesse econômico, fiscal ou climático. A base final consolidada, denominada `panel_master_2014_2024`, integra informações provenientes de diferentes fontes previamente tratadas e padronizadas ao longo do pipeline operacional descrito neste manual.

A estrutura em painel facilita a aplicação de métodos quantitativos e contribui para a organização das informações em formatos compatíveis com análises estatísticas e econométricas. Além disso, a organização modular do pipeline permite que a base seja atualizada mediante a reexecução das rotinas descritas ao longo do documento.

A tabela a seguir apresenta uma representação simplificada da estrutura da base de dados consolidada.

Estrutura da base de dados em painel

ibge7	Ano	pib_pc_real	rc_pc_real	ppt_anual	tmean_anual
1300029	2014	...	...	...	...
1300029	2015	...	...	...	...
1300300	2014	...	...	...	...
1300300	2015	...	...	...	...

2. REQUISITOS

A execução das etapas descritas neste manual requer a utilização de ferramentas computacionais básicas para organização, processamento e validação de dados. Esses requisitos são necessários para garantir o funcionamento adequado dos scripts, a leitura correta dos arquivos de entrada e a geração das bases intermediárias e finais.

Este capítulo apresenta os principais recursos necessários para a execução do pipeline, com destaque para os softwares utilizados, os pacotes da linguagem R e o script responsável pela configuração inicial do projeto.

2.1 SOFTWARES E FERRAMENTAS UTILIZADAS

A construção da base de dados é realizada principalmente em ambiente R, com apoio do RStudio (2025) para organização e execução dos scripts. Também são utilizadas planilhas eletrônicas para inspeção visual, conferência de arquivos e validação complementar das saídas geradas.

A seguir são apresentados os softwares utilizados.

Softwares e ferramentas utilizadas

Ferramenta	Finalidade
RStudio	Execução de rotinas em linguagem R para extração, tratamento, integração e validação dos dados
Microsoft Excel ou equivalente	Visualização, conferência e inspeção das bases em formato tabular
ChatGPT, Gemini ou similares	Apoio na interpretação de erros, adaptação de scripts e organização das rotinas

Recomenda-se que o usuário possua conhecimento básico em R, manipulação de arquivos e organização de planilhas eletrônicas, uma vez que o manual pressupõe a execução de scripts e a verificação das bases geradas ao longo do processo.



Uso de IA

Nesta etapa, as ferramentas de Inteligência Artificial podem auxiliar na instalação dos softwares, na configuração inicial do ambiente de trabalho e na resolução de erros relacionados a permissões, dependências ou incompatibilidades entre pacotes.

Exemplos de prompts:

- “Como instalar o pacote sf no R no Windows?”
- “Erro ao carregar o pacote terra no RStudio [colar erro]. Como resolver?”
- “Como configurar o diretório de trabalho automaticamente no R?”

2.2 PACOTES DA LINGUAGEM R

A execução dos scripts requer a instalação e o carregamento de pacotes específicos da linguagem R. Esses pacotes são responsáveis por diferentes funções, como manipulação de dados, leitura e escrita de arquivos, limpeza de bases, processamento geoespacial e integração com bases públicas.

O carregamento dos pacotes é automatizado pelo script 00_config.R, que verifica a existência das bibliotecas necessárias, instala eventuais pacotes ausentes e carrega os recursos utilizados ao longo do pipeline.

Os principais pacotes utilizados são exibidos a seguir.

Principais pacotes utilizados no projeto

Grupo	Pacotes	Finalidade
Manipulação de dados	dplyr, tidyr, purrr, stringr	Tratamento, transformação e organização das bases
Leitura e escrita de arquivos	readr, readxl, writexl, openxlsx	Importação e exportação de arquivos CSV e Excel
Limpeza de dados	Janitor	Padronização de nomes de colunas e limpeza inicial das bases

Dados geoespaciais	terra, sf, exactextractr	Leitura de arquivos raster e shapefiles, além da extração espacial ponderada
Bases públicas	Sidrar	Acesso a informações do SIDRA/IBGE
Estrutura auxiliar	tibble, rstudioapi	Organização de objetos e seleção interativa de diretórios

Para facilitar a execução, recomenda-se iniciar o projeto pelo script de configuração, pois ele centraliza os pacotes, diretórios e parâmetros globais utilizados pelos demais scripts.

2.3 SCRIPT DE CONFIGURAÇÃO INICIAL

O script `00_config.R` é o ponto de partida do pipeline. Ele tem a função de preparar o ambiente de trabalho, carregar os pacotes necessários, criar a estrutura de diretórios e definir os principais parâmetros utilizados nos scripts seguintes, conforme o quadro abaixo.

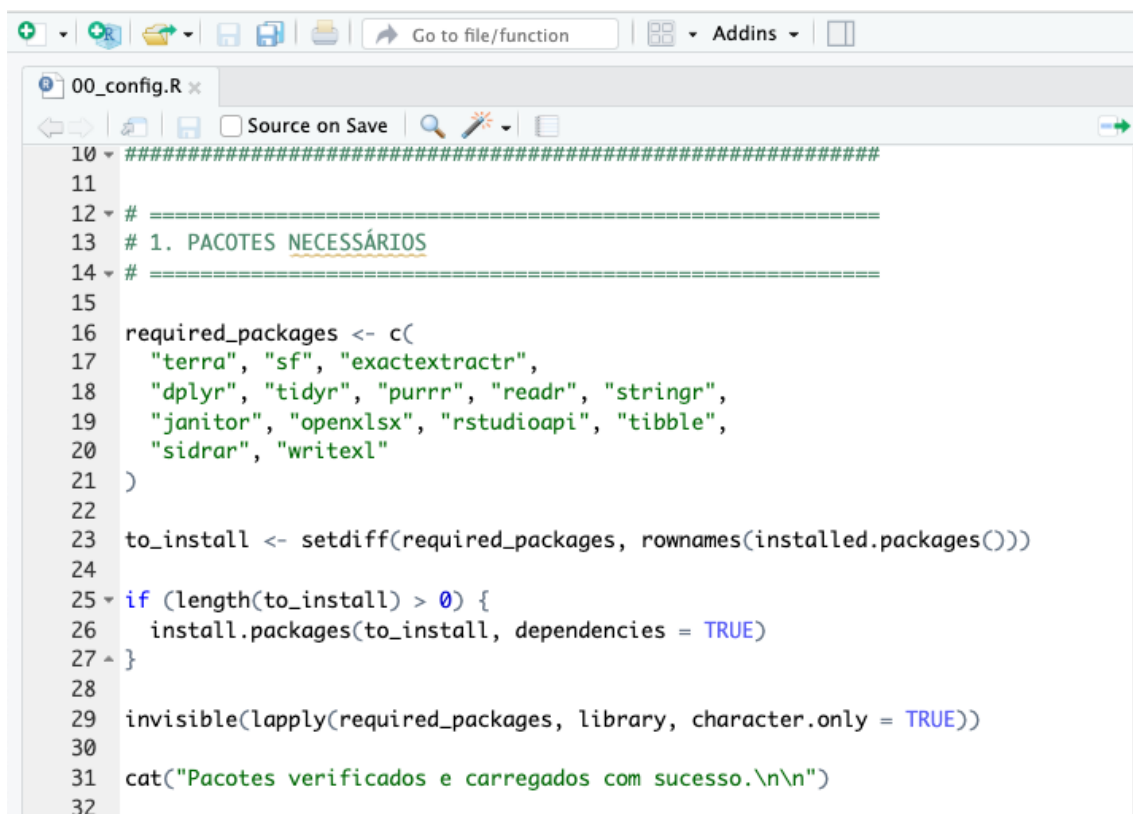
Principais pacotes utilizados no projeto

Item	Descrição
Script	<code>00_config.R</code>
Função	Configurar o ambiente do projeto
Principais ações	Carregar pacotes, criar pastas, definir parâmetros e validar o shapefile
Etapa do pipeline	Preparação do ambiente

Execução
recomendada

Antes de qualquer outro script

Recorte do script



```
10 #####
11
12 # =====
13 # 1. PACOTES NECESSÁRIOS
14 # =====
15
16 required_packages <- c(
17   "terra", "sf", "exactextractr",
18   "dplyr", "tidyr", "purrr", "readr", "stringr",
19   "janitor", "openxlsx", "rstudioapi", "tibble",
20   "sidrar", "writexl"
21 )
22
23 to_install <- setdiff(required_packages, rownames(installed.packages()))
24
25 if (length(to_install) > 0) {
26   install.packages(to_install, dependencies = TRUE)
27 }
28
29 invisible(lapply(required_packages, library, character.only = TRUE))
30
31 cat("Pacotes verificados e carregados com sucesso.\n\n")
32
```

Entre os principais elementos definidos nesse script estão:

- diretório raiz do projeto;
- pastas de dados brutos e processados;
- diretórios específicos para dados climáticos e econômicos;
- período de análise;
- unidade geográfica;
- chave municipal de integração;
- lista de variáveis climáticas;
- parâmetros de extração econômica;
- caminho do shapefile municipal.

A execução correta do 00_config.R é fundamental para que os demais scripts consigam localizar os arquivos de entrada, salvar as saídas nos diretórios adequados e utilizar os mesmos parâmetros de projeto.



Uso de IA

Nesta etapa, a IA pode auxiliar na interpretação de erros de instalação, mensagens de conflito entre pacotes e problemas relacionados à seleção de diretórios.

Exemplos de prompts:

- “O R não encontra o arquivo 00_config.R.
- Como verificar se o caminho está correto?”
- “Como adaptar este script para outro diretório raiz?”

3. ESTRUTURA DO PROJETO

A construção de bases de dados aplicadas requer não apenas a obtenção e o tratamento das informações, mas também a organização adequada do ambiente de trabalho e dos arquivos utilizados ao longo do processo. Uma estrutura bem definida facilita a execução das rotinas, reduz a ocorrência de erros operacionais e contribui para o controle das etapas de processamento.

Neste manual, o projeto de dados é organizado de forma hierárquica, separando arquivos brutos, dados processados, scripts e arquivos auxiliares. Essa organização permite maior clareza na execução das rotinas, além de facilitar a atualização, adaptação e manutenção do pipeline.

Este capítulo apresenta a estrutura operacional utilizada no projeto, incluindo a organização do ambiente de trabalho, os diretórios principais, os dados de entrada necessários e os parâmetros globais empregados na construção da base de dados.

3.1 ORGANIZAÇÃO DO AMBIENTE DE TRABALHO E ESTRUTURA DE DIRETÓRIOS

A organização do ambiente de trabalho é uma etapa fundamental para a execução adequada das rotinas descritas neste manual. O projeto foi estruturado de forma hierárquica, permitindo a separação entre dados brutos, arquivos processados, scripts e arquivos auxiliares utilizados ao longo do pipeline.

A estrutura proposta facilita a localização dos arquivos, o controle das etapas executadas e o armazenamento das saídas geradas durante o processo de construção da base de dados.

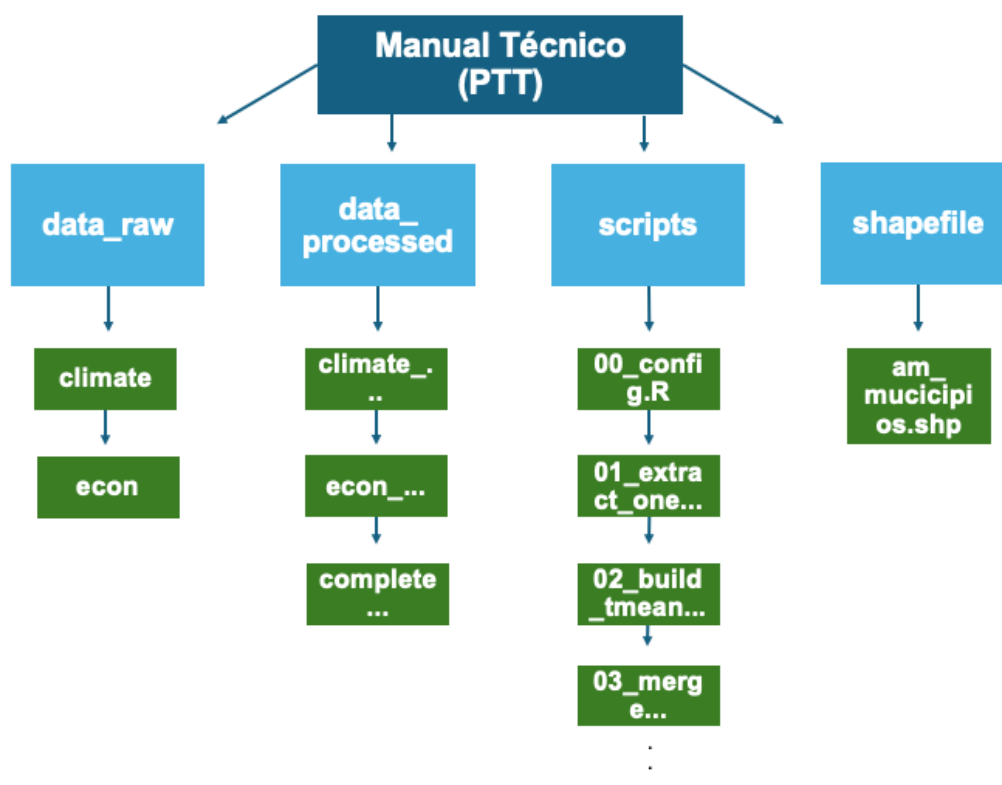
O diretório raiz concentra os principais componentes do projeto:

- `data_raw`: armazenamento dos dados brutos;
- `data_processed`: armazenamento das bases intermediárias e finais;
- `scripts`: scripts responsáveis pelas diferentes etapas do pipeline;
- `shapefiles`: arquivos espaciais utilizados na extração e integração geográfica dos dados.

A estrutura de dados climáticos e econômicos é organizada em subdiretórios específicos, permitindo maior clareza na separação das etapas do processamento.

A Figura 2 apresenta a estrutura geral de diretórios utilizada no projeto.

Figura 2 – Estrutura de diretórios do projeto



O Quadro a seguir apresenta a função dos principais diretórios utilizados no pipeline.

Estrutura dos principais diretórios utilizados no pipeline

Diretório	Finalidade
data_raw	Armazenamento dos arquivos brutos utilizados no projeto
data_processed	Armazenamento das bases intermediárias e finais
scripts	Scripts utilizados nas etapas de processamento
shapefile	Arquivos espaciais utilizados nas rotinas geoespaciais

Para evitar erros relacionados à estrutura do projeto, reiteramos um conjunto de boas práticas:

- Não alterar nomes das pastas durante a execução;
- Separar dados brutos de dados processados;
- Manter logs em diretórios específicos;
- Evitar salvar arquivos manualmente em saídas automatizadas.



Uso de IA

Nesta etapa, ferramentas de Inteligência Artificial podem auxiliar na criação automática da estrutura de diretórios, na adaptação de caminhos de arquivos e na interpretação de erros relacionados à organização do ambiente de trabalho.

Exemplos de prompts:

- “Elabore um script em R para criar automaticamente a estrutura de pastas de um projeto de dados.”
- “Como adaptar os caminhos deste script para outro diretório raiz?”
- “O R não está encontrando uma pasta do projeto. Como verificar o caminho correto?”

3.2 DADOS DE ENTRADA DO PROJETO

A execução das rotinas descritas neste manual depende da disponibilidade prévia de arquivos de entrada provenientes das fontes apresentadas no Capítulo 2. Esses arquivos são utilizados nas etapas de extração, tratamento, integração e validação da base de dados.

Os dados econômicos são organizados em formato tabular, geralmente em arquivos CSV ou Excel, contendo informações fiscais, orçamentárias e socioeconômicas. Já os dados climáticos são disponibilizados em formato raster (.nc – NetCDF), exigindo procedimentos específicos de leitura e processamento geoespacial.

Além dos dados tabulares e raster, o projeto utiliza um arquivo espacial de referência (shapefile) responsável pela delimitação geográfica das unidades de análise.

O quadro abaixo apresenta os principais arquivos de entrada utilizados no pipeline.

Principais arquivos de entrada do projeto

Diretório	Formato	Finalidade
Dados fiscais	.csv	Construção das variáveis fiscais
Dados econômicos	.csv / .xlsx	PIB e população
Dados climáticos	.nc	Variáveis climáticas mensais
Arquivo espacial	.shp	Delimitação geográfica dos municípios

Os arquivos devem ser organizados nas pastas correspondentes definidas na estrutura do projeto, permitindo que os scripts localizem automaticamente os dados necessários para cada etapa do processo.

Scripts relacionados

Script	Função
01_extract_one_variable.R	Leitura e extração dos arquivos climáticos
04_econ_build_fiscal_panel.R	Leitura e tratamento dos arquivos fiscais
05_econ_add_pib_pop_and_finalize.R	Integração econômica complementar



Uso de IA

Ferramentas de IA podem auxiliar na identificação de formatos incompatíveis, erros de leitura de arquivos e validação preliminar dos dados de entrada.

Exemplos de prompts:

- “Como ler arquivos .nc no R utilizando o pacote terra?”
- “O R não reconhece meu arquivo CSV. Como verificar codificação e separador?”
- “Como validar se um shapefile está corretamente estruturado?”

3.3 PARÂMETROS GLOBAIS DO PROJETO

A execução das rotinas descritas neste manual depende da utilização de parâmetros globais compartilhados entre os diferentes scripts do pipeline. Esses parâmetros permitem padronizar a estrutura do projeto e garantir consistência na organização das informações ao longo do processo.

Entre os principais parâmetros utilizados estão:

- período de análise;
- diretório raiz do projeto;
- unidade geográfica de interesse;
- chave municipal de integração;
- variáveis climáticas processadas;
- diretórios de entrada e saída dos dados.

A seguir são apresentados os principais parâmetros utilizados na estruturação do projeto.

Parâmetros globais do projeto

Parâmetro	Descrição
panel_years	Período de análise utilizado no painel
root_dir	Diretório raiz do projeto
ibge7	Chave municipal utilizada na integração
econ_uf	Unidade federativa utilizada nas rotinas econômicas
climate_variables	Lista de variáveis climáticas processadas

A centralização desses parâmetros facilita adaptações futuras do pipeline, permitindo alterações estruturais sem necessidade de modificações extensas nos demais scripts.

Recorte do script

```

132
133 # =====
134 # 6. PARÂMETROS GLOBAIS
135 # =====
136
137 panel_years <- 2014:2024
138
139 econ_years <- panel_years
140 econ_uf <- "AM"
141 muni_key <- "ibge7"
142
143 target_crs_epsg <- 4326
144
145 cat("Parâmetros globais definidos:\n")
146 cat("Painel:", min(panel_years), "-", max(panel_years), "\n")
147 cat("UF econômica:", econ_uf, "\n")
148 cat("Chave municipal:", muni_key, "\n\n")
149

```

Além dos parâmetros globais, o pipeline do projeto foi estruturado de forma modular, utilizando scripts independentes responsáveis por diferentes etapas do processo de construção da base de dados. Essa organização facilita a manutenção das rotinas, amplia a reprodutibilidade do projeto e permite adaptações mais simples em diferentes contextos de aplicação.

Scripts utilizados no pipeline

Script	Função principal
00_config.R	Configuração inicial do ambiente e definição dos parâmetros globais
01_extract_one_variable.R	Extração e agregação anual das variáveis climáticas
02_build_tmean.R	Construção da variável de temperatura média anual
03_merge_climate_all.R	Consolidação do painel climático
04_econ_build_fiscal_panel.R	Construção do painel fiscal municipal
05_econ_add_pib_pop_and_finalize.R	Integração econômica, deflacionamento e cálculo das variáveis per capita
06_build_panel_master_2014_2024.R	Integração final entre bases climáticas e econômicas
07_audit_panel_master_2014_2024.R	Auditoria e validação estrutural da base consolidada

O código completo dos scripts acima encontra-se nos Apêndices.



Uso de IA

Ferramentas de IA podem auxiliar na adaptação dos parâmetros do projeto para novos períodos, unidades geográficas ou estruturas de diretórios.

Exemplos de prompts:

- “Como adaptar este pipeline para outro estado?”
- “Como alterar o período do painel para 2010-2025?”
- “Como modificar a chave municipal utilizada na integração das bases?”

4. DADOS CLIMÁTICOS

Este capítulo apresenta as rotinas utilizadas para extração, tratamento e organização dos dados climáticos empregados na construção da base aplicada. O objetivo é orientar a transformação de arquivos climáticos em formato raster para uma estrutura tabular anual organizada por município.

O fluxo operacional contempla:

- leitura dos arquivos climáticos;
- extração espacial das variáveis;
- agregação temporal;
- construção das variáveis derivadas;
- consolidação do painel climático final.

As rotinas descritas são executadas em linguagem R, utilizando scripts específicos para cada etapa do processo.

4.1 DADOS CLIMÁTICOS UTILIZADOS

Os dados climáticos utilizados neste manual são provenientes da base TerraClimate, que disponibiliza variáveis ambientais em alta resolução espacial e frequência temporal mensal. Os arquivos são disponibilizados em formato NetCDF (.nc), amplamente utilizado para armazenamento de dados raster multidimensionais, exigindo procedimentos específicos de leitura e processamento geoespacial.

No projeto apresentado neste manual, os dados climáticos são organizados para os municípios do estado do Amazonas no período de 2014 a 2024, sendo posteriormente agregados para frequência anual.

As principais variáveis utilizadas são apresentadas a seguir.

Variáveis climáticas utilizadas

Variável	Descrição	Fonte
ppt	Precipitação	TerraClimate
tmax	Temperatura máxima	TerraClimate
tmin	Temperatura mínima	TerraClimate
aet	Evapotranspiração real	TerraClimate

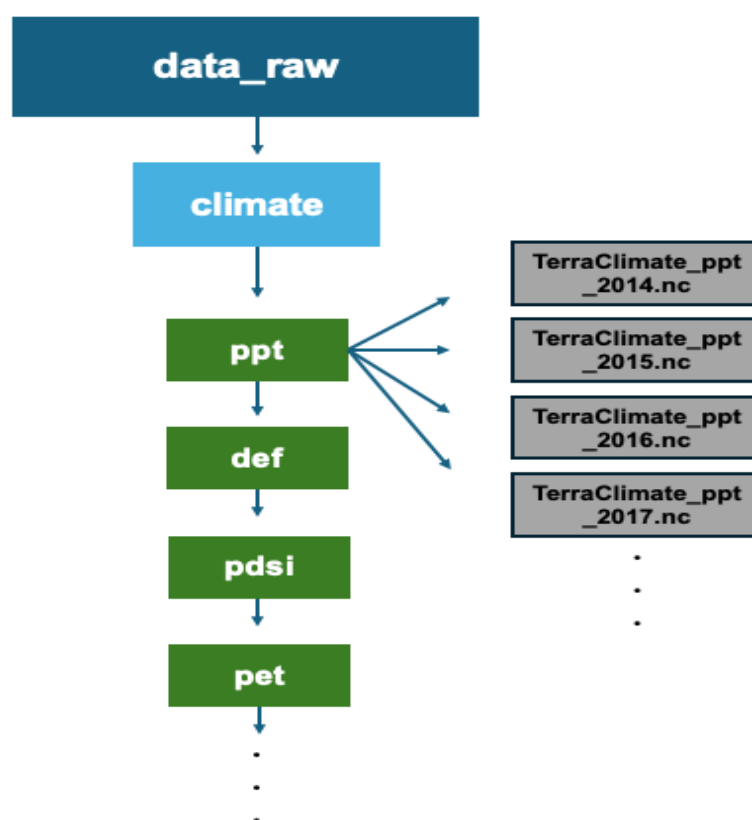
pet	Evapotranspiração potencial	TerraClimate
def	Déficit hídrico	TerraClimate
soil	Umidade do solo	TerraClimate
q	Escoamento superficial	TerraClimate
ws	Velocidade do vento	TerraClimate
pdsi	Índice de severidade de seca	TerraClimate

4.1.1 Estrutura esperada dos arquivos climáticos

Os arquivos climáticos devem ser organizados em subpastas específicas dentro do diretório `data_raw/`climate, separadas conforme a variável processada.

A Figura 3 apresenta um exemplo simplificado da organização esperada dos arquivos climáticos.

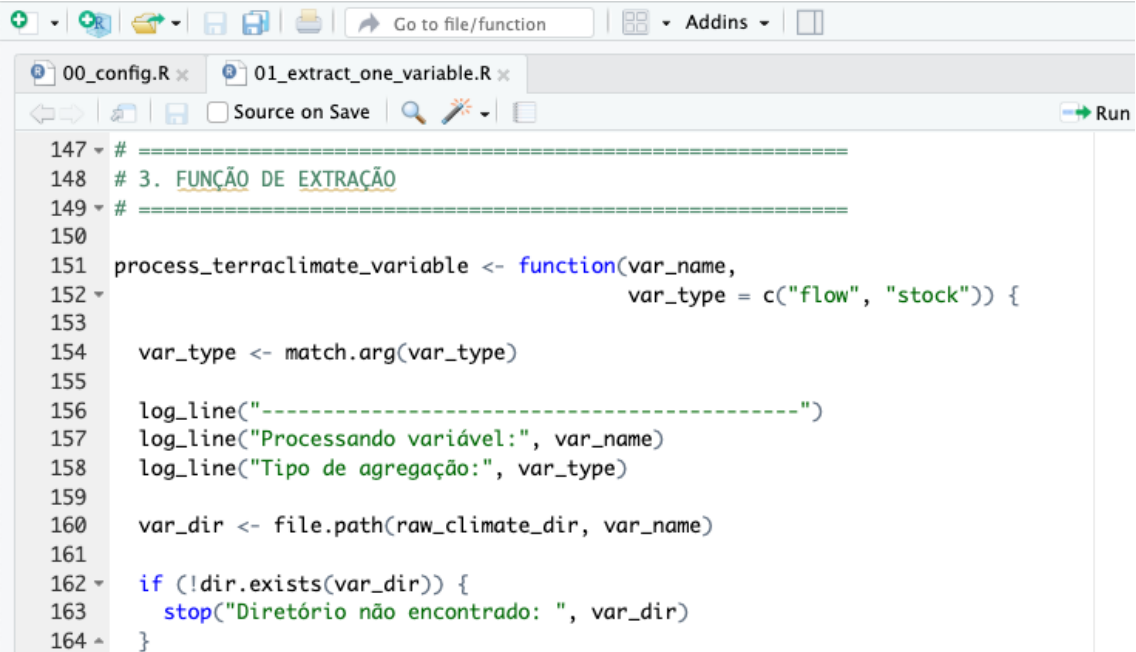
Figura 3 – Organização dos arquivos climáticos



4.2 EXTRAÇÃO ESPACIAL DAS VARIÁVEIS CLIMÁTICAS

A extração espacial das variáveis climáticas é realizada por meio do script 01_extract_one_variable.R, responsável pela leitura dos arquivos raster, extração dos valores municipais e agregação temporal dos dados.

Recorte do script



```
147 # =====
148 # 3. FUNÇÃO DE EXTRAÇÃO
149 # =====
150
151 process_terraclimate_variable <- function(var_name,
152                                           var_type = c("flow", "stock")) {
153
154   var_type <- match.arg(var_type)
155
156   log_line("-----")
157   log_line("Processando variável:", var_name)
158   log_line("Tipo de agregação:", var_type)
159
160   var_dir <- file.path(raw_climate_dir, var_name)
161
162   if (!dir.exists(var_dir)) {
163     stop("Diretório não encontrado: ", var_dir)
164   }
```

O procedimento utiliza os limites municipais definidos no shapefile de referência para calcular valores médios ponderados espacialmente para cada município.

Script utilizado

Script	Função
01_extract_one_variable.R	Leitura dos arquivos NetCDF, extração espacial e agregação anual das variáveis climáticas

Dados de entrada

Arquivo	Localização
Arquivos .nc	data_raw/climate/<variavel>/
Shapefile municipal	shapefiles/am_municipios.shp

Procedimentos executados pelo script

O script realiza as seguintes etapas:

1. Leitura dos arquivos NetCDF;
2. Identificação do período disponível;
3. Carregamento do shapefile municipal;
4. Validação da chave geográfica;
5. Extração espacial dos valores raster;
6. Organização dos dados em estrutura tabular;
7. Agregação temporal para frequência anual;
8. Exportação das bases processadas.

A extração espacial é realizada com apoio do pacote `exactextractr`, que permite calcular médias ponderadas por área para cada unidade espacial.

As variáveis são organizadas inicialmente em frequência mensal e posteriormente agregadas para valores anuais.

Saídas geradas

Arquivo	Diretório
ppt_monthly_2014_2024.csv	climate_outputs/extracted_monthly/
ppt_annual_2014_2024.csv	climate_outputs/annual/
ppt_panel_2014_2024.csv	climate_outputs/final_panel_2014_2024/

Uso de IA

Ferramentas de IA podem auxiliar na leitura de arquivos raster, interpretação de erros geoespaciais e adaptação das rotinas para novas variáveis climáticas.

Exemplos de prompts:

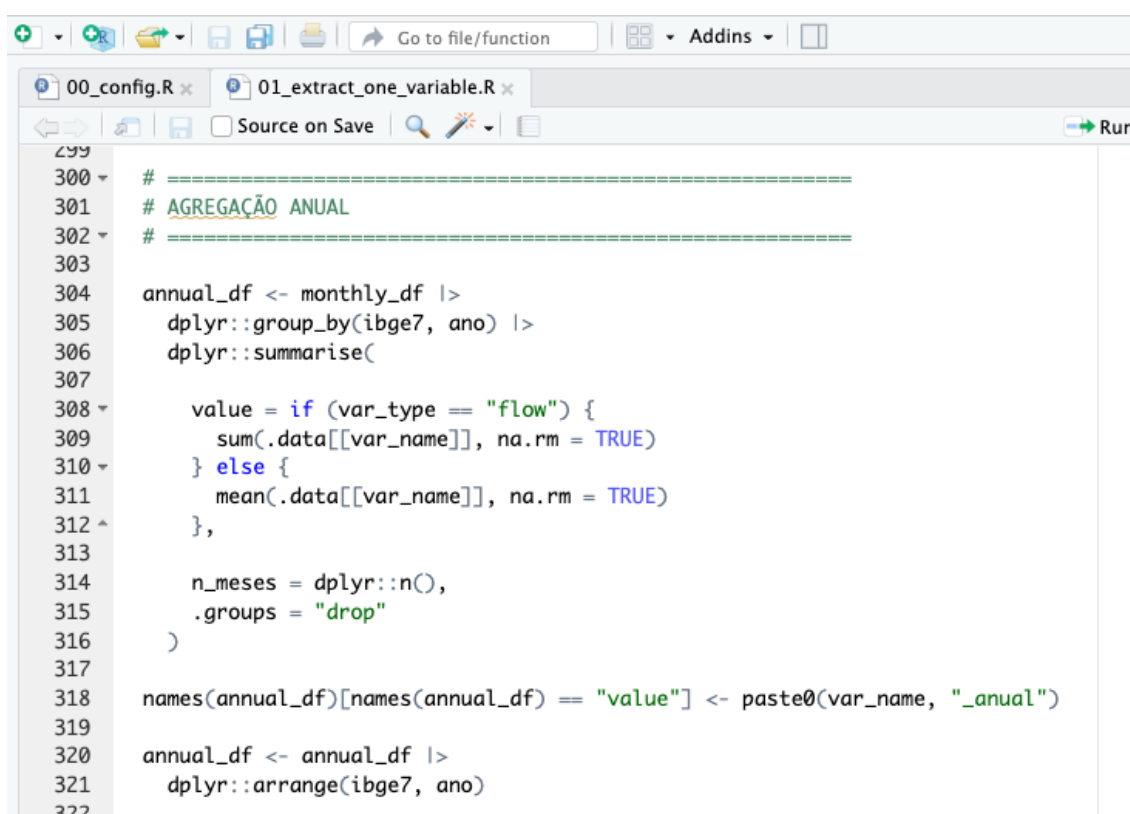
- “Como ler arquivos NetCDF no R utilizando o pacote `terra`?”
- “Explique este erro do `exactextractr`: [colar erro].”
- “Como adaptar este script para processar outra variável climática?”

4.3 AGREGAÇÃO TEMPORAL DOS DADOS CLIMÁTICOS

Após a extração espacial dos dados mensais, os valores climáticos são agregados para frequência anual, permitindo sua compatibilização com os indicadores econômicos e fiscais utilizados na construção da base final.

A agregação anual é realizada automaticamente pelo script 01_extract_one_variable.R, utilizando regras específicas conforme o tipo da variável processada.

Recorte do script



```

299
300 # =====
301 # AGREGAÇÃO ANUAL
302 # =====
303
304 annual_df <- monthly_df |>
305   dplyr::group_by(ibge7, ano) |>
306   dplyr::summarise(
307
308     value = if (var_type == "flow") {
309       sum(.data[[var_name]], na.rm = TRUE)
310     } else {
311       mean(.data[[var_name]], na.rm = TRUE)
312     },
313
314     n_meses = dplyr::n(),
315     .groups = "drop"
316   )
317
318 names(annual_df)[names(annual_df) == "value"] <- paste0(var_name, "_anual")
319
320 annual_df <- annual_df |>
321   dplyr::arrange(ibge7, ano)
322

```

Variáveis de fluxo, como precipitação (ppt) e escoamento superficial (q), são agregadas por soma anual. Já variáveis de estoque ou intensidade, como temperatura e umidade do solo, são agregadas por média anual.

O Quadro 9 apresenta a lógica de agregação utilizada.

Quadro 9 – Regras de agregação temporal das variáveis climáticas

Tipo de variável	Método de agregação
Fluxo	Soma anual

Verificações realizadas

Durante a agregação temporal, o script executa verificações automáticas relacionadas:

- ao número de meses por município;
- à presença de duplicidades;
- à estrutura do painel anual;
- à existência de valores ausentes.

Também são gerados arquivos de log contendo informações sobre a execução da rotina e possíveis inconsistências encontradas durante o processamento.



Uso de IA

Ferramentas de IA podem auxiliar na definição de regras de agregação temporal e interpretação de inconsistências encontradas durante a consolidação annual de dados.

Exemplos de prompts:

- “Quando utilizar soma ou média anual em dados climáticos?”
- “Como verificar duplicidades em uma base anual no R?”
- “Como identificar valores ausentes após agregação temporal?”

4.4 CONSTRUÇÃO DA TEMPERATURA MÉDIA ANUAL

A variável de temperatura média anual (tmean) é construída a partir das variáveis de temperatura máxima (tmax) e temperatura mínima (tmin), utilizando o script 02_build_tmean.R.

Recorte do script

```

183 # =====
184 # 5. CONSTRUIR TMEAN
185 # =====
186
187 tmean_df <- tmax_df |>
188   dplyr::select(
189     ibge7,
190     ano,
191     tmax_anual
192   ) |>
193   dplyr::inner_join(
194
195     tmin_df |>
196       dplyr::select(
197         ibge7,
198         ano,
199         tmin_anual
200       ),
201
202     by = c("ibge7", "ano")
203   ) |>
204   dplyr::mutate(
205     tmean_anual = (tmax_anual + tmin_anual) / 2
206   ) |>
207   dplyr::select(
208     ibge7,
209     ano,
210     tmean_anual
211   ) |>
212   dplyr::arrange(
213     ano,
214     ibge7
215   )
216
217 log_line("tmean construída com sucesso.")

```

O cálculo é realizado a partir da média simples entre as temperaturas máxima e mínima anuais para cada município e período do painel.

Script utilizado

Script	Função
02_build_tmean.R	Construção da variável de temperatura média anual

Equação utilizada

A temperatura média anual é calculada da seguinte forma:

$$tmean = \frac{tmax + tmin}{2}$$

Saída gerada

Arquivo	Diretório
tmean_panel_2014_2024.csv	climate_outputs/final_panel_2014_2024/

**Uso de IA**

Ferramentas de IA podem auxiliar na adaptação de formulas e validação das variáveis derivadas geradas ao longo do processo

Exemplos de prompts:

- “Como calcular temperatura média a partir de tmax e tmin no R?”
- “Como verificar se existem valores inconsistentes em uma variável derivada construída a partir do R?”

4.5 CONSOLIDAÇÃO DO PAINEL CLIMÁTICO

Após o processamento individual das variáveis climáticas, os diferentes painéis anuais são integrados em uma única base consolidada por meio do script 03_merge_climate_all.R.

A integração é realizada utilizando as variáveis:

- ibge7;
- ano.

Recorte do script

```

192 # =====
193 # 4. INTEGRAR PAINÉIS CLIMÁTICOS
194 # =====
195
196 climate_full <- Reduce(
197   function(x, y) {
198     dplyr::full_join(x, y, by = c("ibge7", "ano"))
199   },
200   panels_list
201 ) |>
202   dplyr::arrange(ano, ibge7)
203
204 log_line("Painel climático consolidado criado.")
205 log_line("Linhas =", nrow(climate_full))
206 log_line("Colunas =", ncol(climate_full))
207
208

```

O resultado é um painel climático anual consolidado contendo todas as variáveis processadas para os municípios e períodos analisados.

Script utilizado

Script	Função
03_merge_climate_all.R	Consolidação das variáveis climáticas em painel único

Procedimentos executados pelo script

O script realiza as seguintes etapas:

1. Leitura dos painéis anuais individuais;
2. Padronização das chaves de integração;
3. Integração das bases climáticas;
4. Ordenação estrutural do painel;
5. Exportação da base consolidada.

Saída gerada

Arquivo	Diretório
climate_panel_full_2014_2024.csv	climate_outputs/final_panel_2014_2024/

4.6 VERIFICAÇÕES E VALIDAÇÃO DOS DADOS CLIMÁTICOS

Após a consolidação do painel climático, recomenda-se realizar verificações estruturais para garantir a consistência da base construída.

As principais verificações incluem:

- confirmação da existência dos 62 municípios;
- validação do período de 2014 a 2024;
- ausência de duplicidades em `ibge7 + ano`;
- verificação de valores ausentes;
- conferência das variáveis climáticas exportadas;
- validação dos arquivos de saída.

Verificações recomendadas para o painel climático

Verificação	Objetivo
Número de municípios	Confirmar cobertura espacial completa
Número de anos	Validar estrutura temporal do painel
Duplicidades	Verificar unicidade das observações
Valores ausentes	Identificar inconsistências na extração
Arquivos exportados	Confirmar geração correta das saídas



Uso de IA

Ferramentas de IA podem auxiliar na interpretação de inconsistências encontradas durante a auditoria do painel climático e na construção de rotinas adicionais de validação

Exemplos de prompts:

- “Como verificar duplicidades utilizando o pacote `dplyr`?”
- “Como identificar valores ausentes em um painel de dados?”
- “Como validar a estrutura de um painel municipal anual no R?”

5. DADOS ECONÔMICOS

Este capítulo apresenta as rotinas utilizadas para extração, tratamento e organização dos dados econômicos empregados na construção da base aplicada. O objetivo é orientar a transformação de dados fiscais, econômicos e populacionais em uma estrutura tabular anual por município, compatível com a base climática construída no capítulo anterior.

O fluxo operacional contempla:

- Leitura dos arquivos fiscais;
- Padronização das chaves municipais;
- Construção das variáveis fiscais;
- Integração de PIB e população;
- Deflacionamento das variáveis monetárias;
- Cálculo de indicadores per capita;
- Geração do painel econômico final.

As rotinas descritas são executadas em linguagem R, utilizando scripts específicos para cada etapa do processo.

5.1 DADOS ECONÔMICOS UTILIZADOS

Os dados econômicos utilizados neste manual são provenientes de bases públicas nacionais, com destaque para o Sistema de Informações Contábeis e Fiscais do Setor Público Brasileiro — SICONFI — e o Sistema IBGE de Recuperação Automática — SIDRA.

O SICONFI é utilizado para obtenção de informações fiscais municipais, especialmente receitas e despesas orçamentárias. Já o SIDRA/IBGE é utilizado para obtenção de informações de Produto Interno Bruto municipal, população e índice de preços empregado no deflacionamento das séries monetárias.

Variáveis econômicas utilizadas

Variável	Descrição	Fonte
RC_LIQ	Receita Corrente Líquida municipal	SICONFI/STN
DC	Despesa Corrente municipal	SICONFI/STN

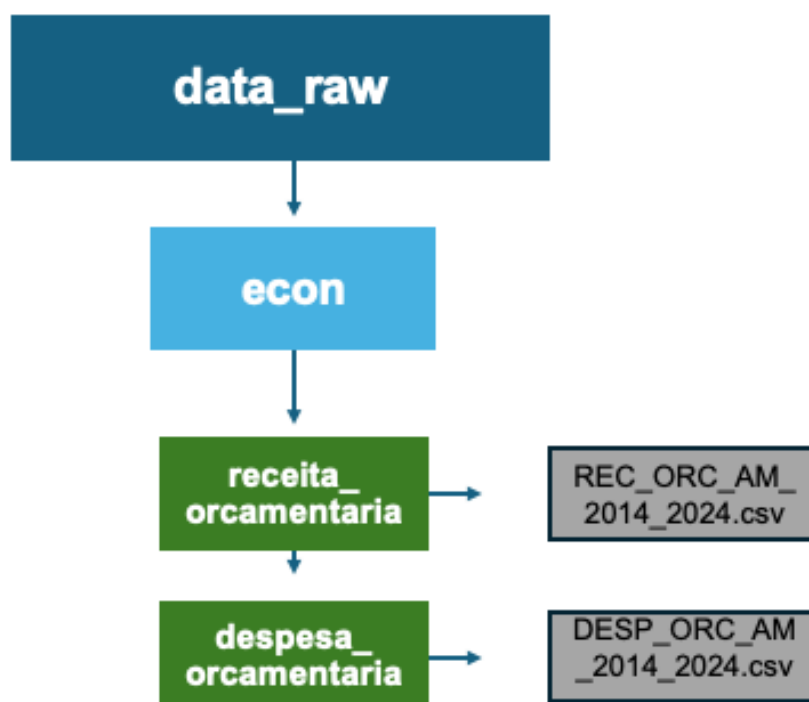
FPM_LIQ	Fundo de Participação dos Municípios líquido	SICONFI/STN
PIB	Produto Interno Bruto municipal	SIDRA/IBGE
POP	População residente	SICONFI/SIDRA
IPCA	Índice de preços utilizado no deflacionamento	SIDRA/IBGE

5.1.1 Estrutura esperada dos arquivos econômicos

Os arquivos econômicos devem ser organizados no diretório `data_raw/econ`, conforme a estrutura definida no projeto. A separação em subpastas facilita a localização dos arquivos e permite que os scripts realizem automaticamente a leitura das bases.

A Figura 4 apresenta um exemplo simplificado da organização esperada dos arquivos econômicos no diretório do projeto.

Figura 4 – Organização dos arquivos climáticos





Ferramentas de IA podem auxiliar na identificação de problemas de leitura dos arquivos, separadores incompatíveis, codificação de caracteres e padronização de nomes de colunas.

Exemplos de prompts:

- “Como identificar o separador correto de um arquivo CSV no R?”
- “O R não está lendo corretamente caracteres com acento em um CSV. Como corrigir?”
- “Como padronizar nomes de colunas utilizando janitor::clean_names()?”

5.2 CONSTRUÇÃO DO PAINEL FISCAL MUNICIPAL

A primeira etapa do tratamento econômico consiste na construção do painel fiscal municipal, realizada pelo script 04_econ_build_fiscal_panel.R.

Recorte do script

```

524 # =====
525 # 10. INTEGRAR PAINEL FISCAL
526 # =====
527 # dplyr::left_join(x, y, by = NULL, copy = FALSE, suffix = c(".x", ".y"), ...,
528 # keep = NULL)
529 econ_panel <- grid |>
530   dplyr::left_join(rc_liq, by = c("ano", "ibge7")) |>
531   dplyr::left_join(fpm_liq, by = c("ano", "ibge7")) |>
532   dplyr::left_join(dc_panel, by = c("ano", "ibge7")) |>
533   dplyr::left_join(pop_panel, by = c("ano", "ibge7")) |>
534   dplyr::mutate(
535     nm_mun = dplyr::coalesce(nm_mun, nm_mun_shp)
536   ) |>
537   dplyr::select(
538     ano,
539     ibge7,
540     nm_mun,
541     rc_liq,
542     fpm_liq,
543     dc,
544     POP
545   ) |>
546   dplyr::arrange(ano, ibge7)

```

Esse script realiza a leitura das bases fiscais do SICONFI, identifica as rubricas de interesse e organiza os dados em formato de painel municipal anual.

Script utilizado

Script	Função
04_econ_build_fiscal_panel.R	Construção do painel fiscal municipal

Dados de entrada

Arquivo	Localização
REC_ORC_AM_2014_2024.csv	data_raw/econ/receita_orcamentaria/
DESP_ORC_AM_2014_2024.csv	data_raw/econ/despesa_orcamentaria/
am_municipios.shp	shapefiles/

Procedimentos executados pelo script

O script realiza as seguintes etapas:

1. Leitura dos arquivos fiscais;
2. Padronização das colunas;
3. construção da chave municipal ibge7;
4. Identificação do ano de referência;
5. Extração da Receita Corrente Líquida;
6. Identificação do Fundo de Participação dos Municípios;;
7. Extração da Despesa Corrente;
8. Integração das variáveis fiscais;
9. Construção do painel anual;
10. Exportação das bases processadas.

A extração espacial é realizada com apoio do pacote `exactextractr`, que permite calcular médias ponderadas por área para cada unidade espacial.

As variáveis são organizadas inicialmente em frequência mensal e posteriormente agregadas para valores anuais.

Saídas geradas

Arquivo	Diretório
fiscal_panel_2014_2024.csv	data_processed/econ_outputs/panel/
fiscal_panel_2014_2024.xlsx	data_processed/econ_outputs/panel/
fiscal_build_log.txt	data_processed/econ_outputs/logs/

Uso de IA

Ferramentas de IA podem auxiliar na interpretação de rubricas fiscais, adaptação de filtros textuais e resolução de erros decorrentes de mudanças na estrutura dos arquivos SICONFI.

Exemplos de prompts:

- “Como filtrar rubricas de Receita Corrente em uma base do SICONFI no R?”
- “Como identificar despesas correntes liquidadas em uma base orçamentária?”
- “Explique este erro ao ler um CSV do SICONFI no R [colar erro].”

5.3 INTEGRAÇÃO DE PIB, POPULAÇÃO E ÍNDICE DE PREÇOS

Após a construção do painel fiscal, a segunda etapa consiste na integração das informações de PIB municipal, população residente e índice de preços.

Essa etapa é realizada pelo script `05_econ_add_pib_pop_and_finalize.R`, responsável por complementar o painel fiscal com informações obtidas via SIDRA/IBGE.

Recorte do script

```
125 # =====
126 # 3. FUNÇÕES AUXILIARES SIDRA
127 # =====
128
129 to_num <- function(x) {
130   if (is.numeric(x)) return(as.numeric(x))
131   readr::parse_number(
132     as.character(x),
133     locale = readr::locale(decimal_mark = ",")
134   )
135 }
136
137 sidra_standardize <- function(df, value_name) {
138   df <- janitor::clean_names(df)
139
140   mun_col <- names(df)[
141     stringr::str_detect(names(df), "municip") &
142     stringr::str_detect(names(df), "codigo")
143   ][1]
144
145   ano_col <- dplyr::coalesce(
146     names(df)[names(df) == "ano"][1],
147     names(df)[stringr::str_detect(names(df), "periodo")][1]
148   )
149
150   val_col <- names(df)[names(df) %in% c("valor", "value")][1]
151
152   if (is.na(mun_col) || is.na(ano_col) || is.na(val_col)) {
153     stop(
154       "SIDRA: não foi possível detectar município, ano ou valor. Colunas: ",
155       paste(names(df), collapse = ", ")
156     )
157   }
158 }
```

Script utilizado

Script	Função
05_econ_add_pib_pop_and_finalize.R	Integração de PIB, população e índice de preços

Dados de entrada

Arquivo	Localização
fiscal_panel_2014_2024.csv	data_processed/econ_outputs/panel/
PIB municipal	SIDRA/IBGE
População residente	SIDRA/IBGE
IPCA	SIDRA/IBGE

Procedimentos executados pelo script

O script realiza as seguintes etapas:

- 1. Leitura do painel fiscal municipal;
- 2. Download do PIB municipal via pacote SIDRA;
- 3. Download da população residente;
- 4. Download do índice de preços;
- 5. Padronização das chaves de integração;
- 6. Integração das bases econômicas;
- 7. Organização estrutural do painel;
- 8. Exportação das bases intermediárias.

Saídas geradas

Arquivo	Diretório
econ_panel_intermediate_2014_2024.csv	data_processed/econ_outputs/final/

 Uso de IA

Ferramentas de IA podem auxiliar na utilização da API SIDRA, interpretação de erros de download e adaptação das consultas para diferentes variáveis econômicas.

Exemplos de prompts:

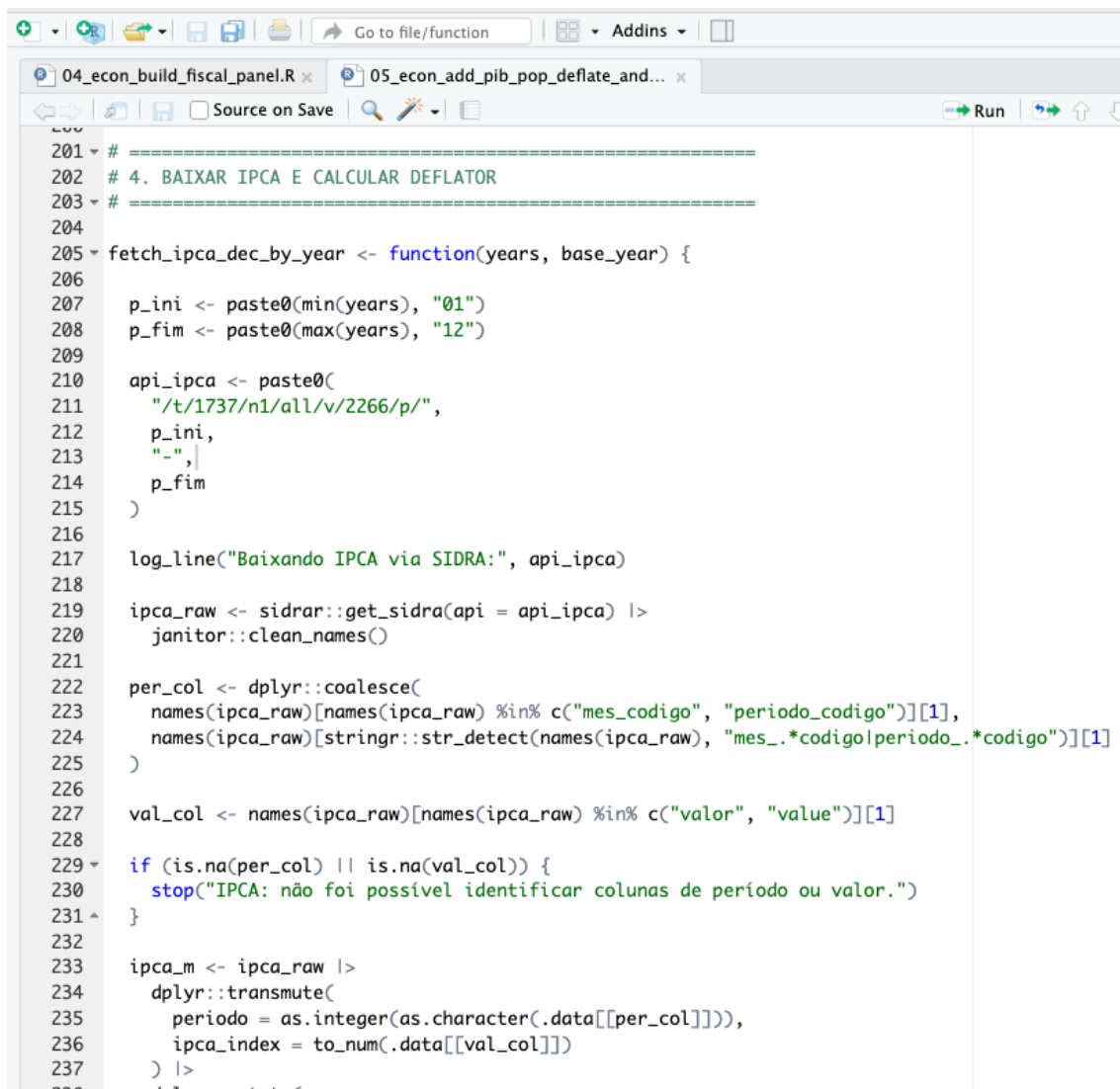
- “Como baixar dados do SIDRA utilizando o pacote sidrar?”
- “Como consultas PIB municipal via API SIDRA?”
- “Como adaptar uma consulta SIDRA para outro indicador econômico?”

5.4 DEFLACIONAMENTO DAS VARIÁVEIS MONETÁRIAS

Após a integração das informações econômicas, as variáveis monetárias são convertidas para valores reais por meio do procedimento de deflacionamento.

O objetivo do deflacionamento é permitir a comparação temporal das séries monetárias, removendo efeitos associados à inflação acumulada ao longo do período analisado.

Recorte do script



```

201 # =====
202 # 4. BAIXAR IPCA E CALCULAR DEFLATOR
203 # =====
204
205 fetch_ipca_dec_by_year <- function(years, base_year) {
206
207   p_ini <- paste0(min(years), "01")
208   p_fim <- paste0(max(years), "12")
209
210   api_ipca <- paste0(
211     "/t/1737/n1/all/v/2266/p/",
212     p_ini,
213     "-",
214     p_fim
215   )
216
217   log_line("Baixando IPCA via SIDRA:", api_ipca)
218
219   ipca_raw <- sidrar::get_sidra(api = api_ipca) |>
220     janitor::clean_names()
221
222   per_col <- dplyr::coalesce(
223     names(ipca_raw)[names(ipca_raw) %in% c("mes_codigo", "periodo_codigo")][1],
224     names(ipca_raw)[stringr::str_detect(names(ipca_raw), "mes_.*codigo|periodo_.*codigo")][1]
225   )
226
227   val_col <- names(ipca_raw)[names(ipca_raw) %in% c("valor", "value")][1]
228
229   if (is.na(per_col) || is.na(val_col)) {
230     stop("IPCA: não foi possível identificar colunas de período ou valor.")
231   }
232
233   ipca_m <- ipca_raw |>
234     dplyr::transmute(
235       periodo = as.integer(as.character(.data[[per_col]])),
236       ipca_index = to_num(.data[[val_col]])
237     ) |>
238     deflacionar(base_year)

```

O fator de deflacionamento é calculado pela razão entre o índice de preços do ano-base e o índice correspondente ao ano de referência:

$$deflator_t = \frac{IPCA_{base}}{IPCA_t}$$

Os valores reais são obtidos pela multiplicação do valor nominal pelo fator de deflacionamento:

$$ValorReal_t = ValorNominal_t \times deflator_t$$

Variáveis reais construídas

Variável	Descrição
rc_liq_real	Receita Corrente Líquida em valores reais
dc_real	Despesa Corrente em valores reais
fpm_liq_real	FPM líquido em valores reais
pib_real	PIB municipal em valores reais



Uso de IA

Ferramentas de IA podem auxiliar na adaptação do ano-base, interpretação do deflacionamento e validação das fórmulas utilizadas.

Exemplos de prompts:

- “Como deflacionar séries monetárias no R?”
- “Como alterar o ano-base de um deflator?”
- “Explique o cálculo do deflator utilizando IPCA”

5.5 CONSTRUÇÃO DAS VARIÁVEIS PER CAPITA

Após o deflacionamento, são construídas variáveis per capita por meio da divisão das variáveis monetárias pela população residente do município correspondente.

Recorte do script

```

312 # =====
313 # 7. INTEGRAR, DEFLACIONAR E CALCULAR PER CAPITA
314 # =====
315
316 econ_final <- econ_panel |>
317   dplyr::left_join(pop_sidra, by = c("ano", "ibge7")) |>
318   dplyr::left_join(pib_df, by = c("ano", "ibge7")) |>
319   dplyr::left_join(
320     ipca_defl |>
321       dplyr::select(ano, ipca_dec, ipca_dec_base, deflator),
322     by = "ano"
323   ) |>
324   dplyr::mutate(
325     pop = dplyr::coalesce(pop, POP_SIDRA),
326
327     rc_pc = dplyr::if_else(!is.na(pop) & pop > 0, rc_liq / pop, NA_real_),
328     fpm_pc = dplyr::if_else(!is.na(pop) & pop > 0, fpm_liq / pop, NA_real_),
329     dc_pc = dplyr::if_else(!is.na(pop) & pop > 0, dc / pop, NA_real_),
330     pib_pc = dplyr::if_else(!is.na(pop) & pop > 0, PIB / pop, NA_real_),
331

```

Esse procedimento permite reduzir distorções associadas ao porte populacional dos municípios e facilita comparações entre unidades territoriais de diferentes tamanhos.

$$VariavelPerCapita_{it} = \frac{Variavel_{it}}{Populacao_{it}}$$

Onde:

- i representa o município;
- t representa o ano;
- $Populacao_{\{it\}}$ representa a população residente do município i no ano t .

Variáveis reais per capita construídas

Variável	Descrição
rc_pc_real	Receita Corrente Líquida real per capita
dc_pc_real	Despesa Corrente real per capita
fpm_pc_real	FPM líquido real per capita
pib_pc_real	PIB real per capita



Uso de IA

Ferramentas de IA podem auxiliar na construção de variáveis derivadas e validação dos cálculos realizados nas bases econômicas.

Exemplos de prompts:

- “Como calcular variáveis per capita em um painel municipal no R?”
- “Como verificar divisões por zero em uma base de dados no R?”
- “Como validar variáveis derivadas no R?”

5.6 CONSOLIDAÇÃO DO PAINEL ECONÔMICO FINAL

Após o processamento das variáveis fiscais, econômicas e populacionais, o script 05_econ_add_pib_pop_and_finalize.R gera o painel econômico consolidado.

Esse painel contém as variáveis:

- Fiscais;
- Econômicas;
- Populacionais;
- Reais
- Per capita.

Recorte do script

```

377 # =====
378 # 9. EXPORTAR
379 # =====
380
381 out_csv <- file.path(econ_final_dir, "econ_panel_final_2014_2024.csv")
382 out_xlsx <- file.path(econ_final_dir, "econ_panel_final_2014_2024.xlsx")
383
384 readr::write_csv(econ_final, out_csv)
385
386 wb <- openxlsx::createWorkbook()
387
388 openxlsx::addWorksheet(wb, "econ_panel_final")
389 openxlsx::writeDataTable(wb, "econ_panel_final", econ_final)
390
391 meta <- tibble::tibble(
392   item = c(
393     "gerado_em",
394     "periodo",
395     "ano_base_deflator",
396     "fonte_pib",
397     "fonte_pop",
398     "fonte_ipca",
399     "observacao"
400   ),

```

A base consolidada será posteriormente integrada ao painel climático na etapa seguinte do pipeline.

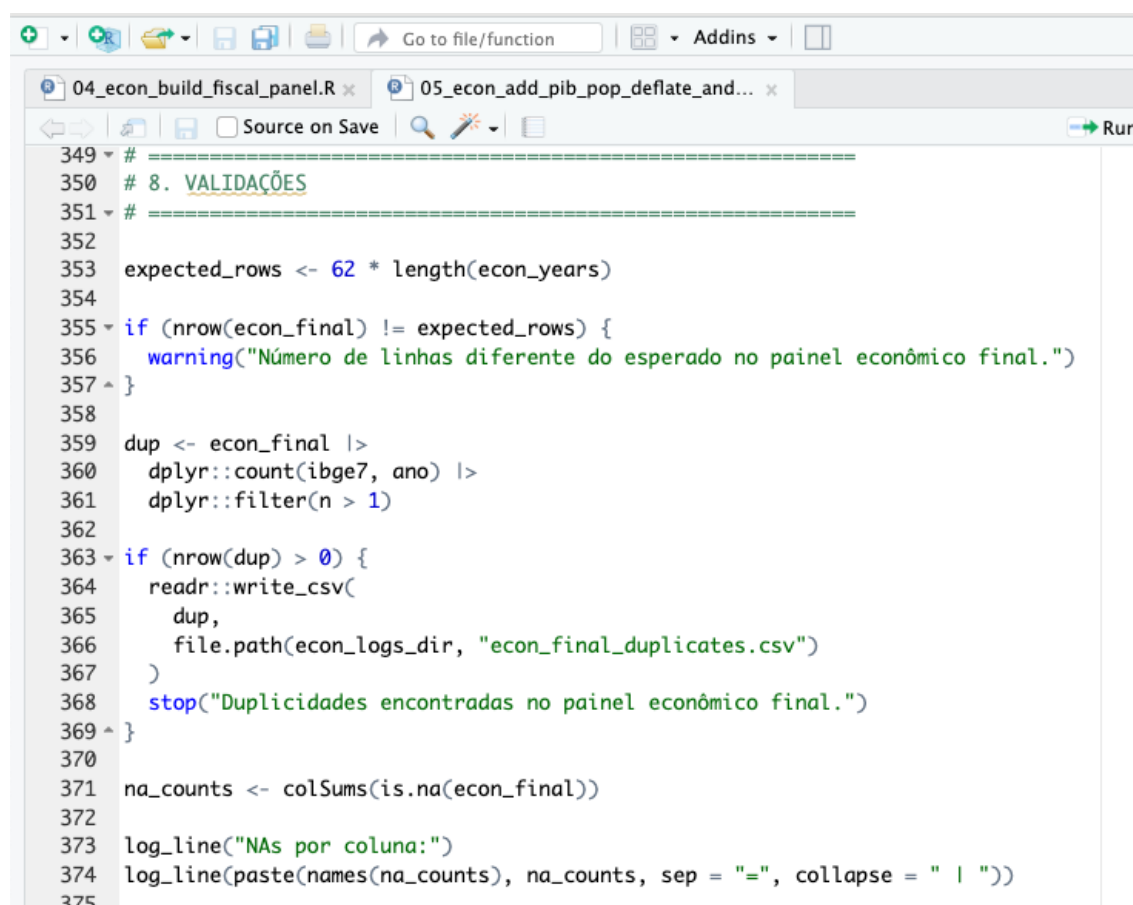
Saídas geradas

Arquivo	Diretório
econ_panel_final_2014_2024.csv	data_processed/econ_outputs/final/
econ_panel_final_2014_2024.xlsx	data_processed/econ_outputs/final/
econ_final_build_log.txt	data_processed/econ_outputs/logs/ ,

5.7 VERIFICAÇÕES E VALIDAÇÃO DOS DADOS ECONÔMICOS

Após a consolidação do painel econômico, recomenda-se realizar verificações estruturais para garantir a consistência da base construída.

Recorte do script



```

349 # =====
350 # 8. VALIDAÇÕES
351 # =====
352
353 expected_rows <- 62 * length(econ_years)
354
355 if (nrow(econ_final) != expected_rows) {
356   warning("Número de linhas diferente do esperado no painel econômico final.")
357 }
358
359 dup <- econ_final |>
360   dplyr::count(ibge7, ano) |>
361   dplyr::filter(n > 1)
362
363 if (nrow(dup) > 0) {
364   readr::write_csv(
365     dup,
366     file.path(econ_logs_dir, "econ_final_duplicates.csv")
367   )
368   stop("Duplicidades encontradas no painel econômico final.")
369 }
370
371 na_counts <- colSums(is.na(econ_final))
372
373 log_line("NAs por coluna:")
374 log_line(paste(names(na_counts), na_counts, sep = "=", collapse = " | "))
375

```

As principais verificações incluem:

- Confirmação da existência dos municípios esperados;
- Validação do período do painel;
- Ausência de duplicidades em $\text{ibge7} + \text{ano}$;
- Verificação de valores ausentes;
- Conferência das variáveis exportadas;
- Validação das variáveis reais e per capita.

A seguir são apresentadas as principais verificações recomendadas para o painel econômico.

Verificações recomendadas para o painel econômico

Verificação	Objetivo
Número de municípios	Confirmar cobertura espacial completa
Número de anos	Validar estrutura temporal do painel
Duplicidades	Verificar unicidade das observações
Valores ausentes	Identificar inconsistências na integração
Variáveis reais	Confirmar aplicação do deflator
Variáveis per capita	Validar cálculos populacionais



Uso de IA

Ferramentas de IA podem auxiliar na interpretação de inconsistência encontradas durante a auditoria do painel econômico e na adaptação das rotinas para novos contextos de análise.

Exemplos de prompts:

- “Como verificar duplicidades em um painel municipal no R?”
- “Como identificar valores ausentes em uma base no R?”
- “Como validar variáveis reais e per capita no R?”

6. INTEGRAÇÃO E AUDITORIA

Este capítulo apresenta as etapas responsáveis pela integração das bases climáticas e econômicas construídas nos capítulos anteriores, resultando na geração da base consolidada final do projeto. O objetivo dessa etapa é estruturar um painel integrado contendo informações padronizadas, organizadas e compatíveis entre si, permitindo sua utilização em análises estatísticas, econométricas e aplicações quantitativas diversas.

O processo de integração envolve a consolidação das bases intermediárias por meio de chaves comuns de identificação espacial e temporal, além da realização de verificações estruturais voltadas à consistência do painel final. As rotinas descritas contemplam:

- Padronização das chaves de integração;
- Verificação da compatibilidade entre os painéis;
- Integração das bases climáticas e econômicas;
- Geração da base consolidada final;
- Auditoria e validação estrutural do painel.

As etapas são executadas por meio dos scripts:

- 06_build_panel_master_2014_2024.R;
- 07_audit_panel_master_2014_2024.R.

6.1 INTEGRAÇÃO DAS BASES CLIMÁTICAS E ECONÔMICAS

Após a construção individual dos painéis climáticos e econômicos, realiza-se a integração final das informações em uma única estrutura em painel. Essa etapa constitui uma das fases centrais do pipeline, pois permite a consolidação de variáveis provenientes de diferentes fontes em uma base única, organizada por município e ano.

Recorte do script

```

214 # =====
215 # 5. INTEGRAR BASE MASTER
216 # =====
217
218 panel_master <- econ |>
219   dplyr::inner_join(
220     climate,
221     by = c("ibge7", "ano")
222   ) |>
223   dplyr::arrange(ano, ibge7)
224
225 expected_rows <- 62 * length(panel_years)
226
227 if (nrow(panel_master) != expected_rows) {
228   stop("panel_master possui número de linhas diferente do esperado.")
229 }
230
231 if (any(duplicated(names(panel_master)))) {
232   stop("Há nomes de colunas duplicados após o merge.")
233 }
234
235 log_line("Panel master criado.")
236 log_line("Linhas =", nrow(panel_master))
237 log_line("Colunas =", ncol(panel_master))
238
239
240 #

```

A integração é realizada por meio das chaves:

- `ibge7` — identificador municipal;
- `ano` — referência temporal anual.

Script utilizado

Script	Função
06_build_panel_master_2014_2024.R	Integração final das bases climáticas e econômicas

Dados de entrada

Arquivo	Localização
climate_panel_full_2014_2024.csv	data_processed/climate_outputs/final_panel_2014_2024/
econ_panel_final_2014_2024.csv	data_processed/econ_outputs/final/
00_config.R	scripts/

Procedimentos executados pelo script

O script realiza as seguintes etapas:

1. Leitura das bases climáticas e econômicas;
2. Padronização das chaves municipais;
3. Validação estrutural das bases;
4. Verificação de duplicidades;
5. Checagem da cobertura temporal;
6. Validação da estrutura balanceada do painel;
7. Verificação da compatibilidade entre as chaves;
8. Integração das bases por `ibge7 + ano`;
9. Ordenação e consolidação do painel final;
10. Geração automática do dicionário de variáveis;
11. Exportação dos arquivos finais do projeto.

O resultado da integração consiste em uma base consolidada em estrutura de painel municipal anual, organizada pelas chaves `ibge7` e `ano`, contendo variáveis econômicas, fiscais e climáticas previamente tratadas ao longo do pipeline.

A estrutura geral da base consolidada foi apresentada anteriormente na Seção 2.3 deste manual.

Saídas geradas

Arquivo	Diretório
<code>panel_master_2014_2024.csv</code>	Base final consolidada
<code>panel_master_2014_2024.xlsx</code>	Base final em Excel
<code>panel_master_dictionary_2014_2024.csv</code>	Dicionário das variáveis
<code>panel_master_build_log.txt</code>	Log da integração final



Uso de IA

Ferramentas de IA podem auxiliar na identificação de erros de merge, inconsistências de chaves e problemas de compatibilidade entre bases de diferentes origens.

Exemplos de prompts:

- “Como verificar se duas bases possuem cobertura 1:1 no R?”
- “Como identificar chaves ausentes após um inner_join no R?”
- “Como automatizar validações estruturais em um painel de dados no R?”

6.2 GERAÇÃO DO DICIONÁRIO DE VARIÁVEIS

Além da construção do painel consolidado, o pipeline realiza automaticamente a geração de um dicionário de variáveis contendo descrições, classificações e informações complementares sobre os campos presentes na base final.

O objetivo desse procedimento é facilitar a interpretação das variáveis, ampliar a transparência metodológica e contribuir para a reprodutibilidade da base construída.

O dicionário é produzido automaticamente durante a execução do script `06_build_panel_master_2014_2024.R`.

Recorte do script

```

240 # =====
241 # 6. GERAR DICIONÁRIO DE VARIÁVEIS
242 # =====
243
244 infer_origem <- function(v) {
245   if (v %in% c("ano", "ibge7", "nm_mun")) return("identificacao")
246   if (stringr::str_detect(v, "ipcaldeflator")) return("economica_precos")
247   if (stringr::str_detect(v, "pib")) return("economica_produto")
248   if (stringr::str_detect(v, "rc_ldclfp_m")) return("economica_fiscal")
249   if (v %in% c("pop", "pop_sidra")) return("economica_demografia")
250   if (stringr::str_detect(v, "_anual$")) return("climatica_anual")
251   return("derivada")
252 }
253
254 infer_fonte <- function(origem) {
255   dplyr::case_when(
256     origem == "identificacao" ~ "IBGE / integração municipal",
257     origem == "economica_precos" ~ "SIDRA/IBGE - IPCA",
258     origem == "economica_produto" ~ "SIDRA/IBGE - PIB Municipal",
259     origem == "economica_fiscal" ~ "SICONFI/STN",
260     origem == "economica_demografia" ~ "SICONFI/STN e SIDRA/IBGE",
261     origem == "climatica_anual" ~ "TerraClimate",
262     TRUE ~ "Derivação própria"
263   )
264 }

```

Informações geradas no dicionário

O dicionário contempla:

- Nome da variável;
- Origem da informação;
- Fonte dos dados;
- Unidade de medida;
- Descrição detalhada;
- Classificação da variável;
- Transformações aplicadas.

Estrutura simplificada do dicionário

Variável	Origem	Unidade	Descrição
pib_pc_real	Econômica	R\$/hab	PIB real per capita
ppt_anual	Climática	mm	Precipitação anual
tmean_anual	Climática	°C	Temperatura média anual

Print do dicionário

Página Inicial		Inserir	Desenhar	Layout da Página	Fórmulas	Dados	Revisão	Exibir	Automatizar				
		Calibri (Corpo) 11 N I <u>S</u>											
Q23													
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	var	origem	fonte	unidade	descricao	classe	n_unique	na_count	na_rate	min	p50	p99	max
2	aet_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		682	0	0	75,36354	90,85649	103,1753	105,9926
3	def_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		675	0	0	0	9,887064	37,92469	43,38179
4	pdsi_anual	climatica_anual	TerraClimate	índice adin	Variável climática municipal referent numeric		682	0	0	-10,5212	-0,73673	4,556251	5,254519
5	pet_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		682	0	0	85,2428	101,4927	124,8565	128,4895
6	ppt_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		682	0	0	1484,426	2325,802	3495,467	3622,91
7	q_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		682	0	0	442,5988	1251,603	2374,457	2512,428
8	soil_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referent numeric		678	0	0	18,05956	150,479	290,1196	328,8035
9	tmax_anua	climatica_anual	TerraClimate	°C	Variável climática municipal referent numeric		682	0	0	30,68148	32,03833	33,80299	34,14498
10	tmean_an	climatica_anual	TerraClimate	°C	Variável climática municipal referent numeric		682	0	0	25,23557	27,27097	28,56466	28,79061
11	tmin_anua	climatica_anual	TerraClimate	°C	Variável climática municipal referent numeric		682	0	0	19,27722	22,5409	23,86595	23,97797
12	ws_anua	climatica_anual	TerraClimate	m/s	Variável climática municipal referent numeric		682	0	0	0,686644	1,178329	1,649672	1,711906
13	pop	economica_demografia	SICONFI/STN e SIDRA/IBGE	habitantes	População residente utilizada no pai numeric		495	0	0	1755	24351	2064680	2255903
14	pop_sidra	economica_demografia	SICONFI/STN e SIDRA/IBGE	habitantes	População residente obtida pelo SID numeric		556	124	18,18	1755	25049,5	2109816	2279686
15	dc	economica_fiscal	SICONFI/STN	R\$ corrente	Despesa Corrente municipal liquidad numeric		671	9	1,32	8699514	63319727	3,94e+09	8,92e+09
16	dc_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variação per capita calculada pela div numeric		674	9	1,32	388,3018	2380,243	9057,645	37585,64
17	dc_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, pr	Variável monetária convertida para v numeric		674	9	1,32	577,3263	3199,025	9757,983	39401,48
18	dc_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para v numeric		672	9	1,32	12934417	80081876	5,71e+09	8,92e+09
19	fpm_liq	economica_fiscal	SICONFI/STN	R\$ corrente	Fundo de Participação dos Municípios numeric		351	9	1,32	6318055	23296837	5,67e+08	2,81e+09
20	fpm_liq_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para v numeric		350	9	1,32	9930041	32569449	8,14e+08	2,81e+09
21	fpm_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variação per capita calculada pela div numeric		673	9	1,32	214,7199	882,3362	5518,023	21298,85
22	fpm_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, pr	Variável monetária convertida para v numeric		673	9	1,32	358,0787	1241,462	5750,978	23359,65
23	rc_liq	economica_fiscal	SICONFI/STN	R\$ corrente	Receta Corrente Líquida municipal e numeric		670	10	1,47	195239536	80409044	5,14e+09	1,13e+10
24	rc_liq_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para v numeric		671	10	1,47	27181829	1,01e+08	7,66e+09	1,13e+10
25	rc_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variação per capita calculada pela div numeric		673	10	1,47	1060,112	2984,368	11124,95	46745,35
26	rc_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, pr	Variável monetária convertida para v numeric		673	10	1,47	1475,77	3955,547	11604,03	49003,71
27	deflator	economica_precos	SIDRA/IBGE – IPCA	adimensor	Fator de deflacionamento utilizado p numeric		11	0	0	1	3334618	1748952	1748952
28	ipca_dec	economica_precos	SIDRA/IBGE – IPCA	número-in	IPCA de dezembro do ano de referênc numeric		11	0	0	4059,86	5320,25	7100,5	7100,5
29	ipca_dec_ib	economica_precos	SIDRA/IBGE – IPCA	número-in	IPCA de dezembro do ano-base. numeric		1	0	0	7100,5	7100,5	7100,5	7100,5
30	piib	economica_produto	SIDRA/IBGE – PIB Municipal	R\$ corrente	Produto Interno Bruto municipal em numeric		620	62	9,09	58162000	2,39e+08	7,27e+10	1,28e+11
31	piib_pc	economica_produto	SIDRA/IBGE – PIB Municipal	R\$/hab	Variação per capita calculada pela div numeric		621	62	9,09	4298,185	9911,12	46222,47	82831,13



💡 Uso de IA

Ferramentas de IA podem auxiliar na documentação automática das variáveis e padronização das descrições do dicionário.

Exemplos de prompts:

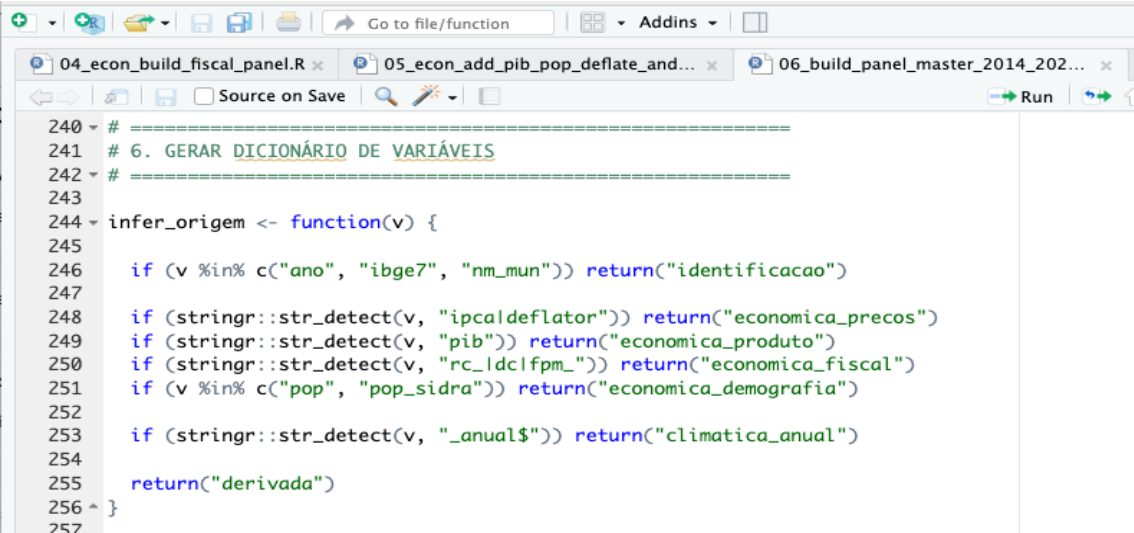
- “Como gerar automaticamente um dicionário no R?”
- “Como documentar variáveis derivadas em bases de dados no R?”
- “Como criar descrições automáticas para colunas de um painel no R?”

6.3 AUDITORIA E VALIDAÇÃO DA BASE FINAL

Após a integração do painel consolidado, são executadas rotinas de auditoria voltadas à verificação da integridade estrutural da base final. Essa etapa é fundamental para identificar inconsistências capazes de comprometer análises posteriores.

As verificações são realizadas pelo script 07_audit_panel_master_2014_2024.R, responsável pela execução automática de testes estruturais e geração de relatórios de auditoria.

Recorte do script



```
240 # =====
241 # 6. GERAR DICIONÁRIO DE VARIÁVEIS
242 # =====
243
244 infer_origem <- function(v) {
245
246   if (v %in% c("ano", "ibge7", "nm_mun")) return("identificacao")
247
248   if (stringr::str_detect(v, "ipcaldeflator")) return("economica_precos")
249   if (stringr::str_detect(v, "pib")) return("economica_produto")
250   if (stringr::str_detect(v, "rc_idclfp_m")) return("economica_fiscal")
251   if (v %in% c("pop", "pop_sidra")) return("economica_demografia")
252
253   if (stringr::str_detect(v, "_anual$")) return("climatica_anual")
254
255   return("derivada")
256 }
257
```

Script utilizado

Script	Função
07_audit_panel_master_2014_2024.R	Auditoria estrutural e validação do painel final

Dados de entrada

Arquivo	Localização
panel_master_2014_2024.csv	data_processed/complete_outputs/
panel_master_dictionary_2014_2024.csv	data_processed/complete_outputs/
00_config.R	scripts/

Procedimentos executados pelo script

- O script realiza as seguintes etapas:
1. Leitura da base consolidada;
 2. Validação da estrutura do painel;
 3. Verificação de duplicidades;
 4. Identificação de lacunas nas chaves;

5. Checagem de valores ausentes;
6. Detecção de colunas duplicadas;
7. Validação de variáveis monetárias;
8. Checagem de coerência das variáveis climáticas;
9. Geração de relatórios de auditoria;
10. Identificação preliminar de outliers.

Relatórios gerados

Arquivo	Finalidade
audit_panel_master_2014_2024.txt	Relatório geral de auditoria
audit_na_by_var.csv	Valores ausentes por variável
audit_na_by_year.csv	Valores ausentes por ano
audit_key_holes.csv	Lacunas nas chaves do painel
audit_outliers_top10.csv	Principais outliers identificados
audit_summary.csv	Resumo consolidado da auditoria

Principais verificações estruturais

Entre as verificações realizadas, destacam-se:

- Confirmação da estrutura balanceada do painel;
- Validação das combinações ibge7 + ano;
- Checagem de municípios ausentes;
- Detecção de inconsistências após merges;
- Verificação de colunas duplicadas;
- Análise de cobertura temporal das variáveis.



Uso de IA

Ferramentas de IA podem auxiliar na interpretação dos relatórios de auditoria, identificação de inconsistências estruturais e elaboração de rotinas adicionais de validação.

Exemplos de prompts:

- “Como interpretar relatórios de NA em um painel?”
- “Como identificar outliers automaticamente em bases municipais no R?”
- “Como validar a integridade estrutural de um painel no R?”

6.4 BASE FINAL CONSOLIDADA

Ao final do pipeline, obtém-se a base consolidada `panel_master_2014_2024`, contendo informações econômicas, fiscais e climáticas organizadas em estrutura de painel municipal anual.

A base final representa o principal produto operacional do processo descrito neste manual, podendo ser utilizada em:

- Análises estatísticas;
- Modelos econométricos;
- Estudos climáticos;
- Pesquisas aplicadas;
- Construção de indicadores;
- Desenvolvimento de dashboards e painéis analíticos.

A organização modular do pipeline permite:

- Atualização automática das bases;
- Adaptação para outros recortes espaciais;
- Inclusão de novas variáveis;
- Replicação para diferentes projetos de pesquisa.

Além disso, a estrutura padronizada contribui para a rastreabilidade das etapas executadas e para a reprodutibilidade metodológica do processo de construção da base de dados.

Print do panel_master_2014_2024

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	ano	ibge7	nm_m	rc_liq	fpm_li	dc	pop	pop_sic	pib	ipca_di	ica_dec	deflat	rc_pc	fpm_pi	dc_px
2	2014	1300029	Alvarães	29510766	8098596	21610450	15357	15357	1,08E+08	4059,86	7100,5	1,748952	1921,649	527,3553	1407,20
3	2014	1300060	Amatúrá	26083750	9311133	17778489	10644	10644	63736000	4059,86	7100,5	1,748952	2450,559	874,7776	1670,28
4	2014	1300086	Anamá	26394887	9311133	15913419	11981	11981	86080000	4059,86	7100,5	1,748952	2203,062	777,1583	1328,22
5	2014	1300102	Anori	34140649	13966700	26172060	18826	18826	1,69E+08	4059,86	7100,5	1,748952	1813,484	741,8836	1390,20
6	2014	1300144	Apuí	38305283	13966700	30849408	20258	20258	1,93E+08	4059,86	7100,5	1,748952	1890,872	689,4412	1522,82
7	2014	1300201	Atalaia do I	41129163	13966700	30277954	17658	17658	1,25E+08	4059,86	7100,5	1,748952	2329,208	790,9559	1714,68
8	2014	1300300	Autazes	66189354	19645302	46993840	36301	36301	2,39E+08	4059,86	7100,5	1,748952	1823,348	541,178	1294,5
9	2014	1300409	Barcelos	47769121	18622267	37241494	27273	27273	1,48E+08	4059,86	7100,5	1,748952	1751,517	682,8096	1365,50
10	2014	1300508	Barreirinha	61405127	16294483	48601269	30202	30202	2,11E+08	4059,86	7100,5	1,748952	2033,148	539,5167	1609,20
11	2014	1300607	Benjamin C	77465917	20950050	58424876	38533	38533	2,71E+08	4059,86	7100,5	1,748952	2010,379	543,6911	1516,2
12	2014	1300631	Beruri	44600659	13966700	34984726	17755	17755	1,21E+08	4059,86	7100,5	1,748952	2512,006	786,6347	1970,41
13	2014	1300680	Boa Vista d	32214352	11638917	25603317	17248	17248	98384000	4059,86	7100,5	1,748952	1867,715	674,798	1484,42
14	2014	1300706	Boca do Ac	57561458	18622267	46369727	33148	33148	2,78E+08	4059,86	7100,5	1,748952	1736,499	561,7916	1398,8
15	2014	1300805	Borba	73716245	20950050	57802601	38688	38688	2,55E+08	4059,86	7100,5	1,748952	1905,403	541,5129	1494,07
16	2014	1300839	Caapiranga	31163771	9311133	24354823	12214	12214	1,31E+08	4059,86	7100,5	1,748952	2551,48	762,3328	1994,00
17	2014	1300904	Canutama	26184273	8082212	24707413	14944	14944	88213000	4059,86	7100,5	1,748952	1752,16	540,8332	1653,33
18	2014	1301001	Carauari	52614600	16294483	42374934	27645	27645	2,36E+08	4059,86	7100,5	1,748952	1903,223	589,4188	1532,82
19	2014	1301100	Careiro	61745218	18622267	45674911	35938	35938	2,18E+08	4059,86	7100,5	1,748952	1718,104	518,1776	1270,93
20	2014	1301159	Careiro da	45414236	16294483	35167758	27357	27357	2,41E+08	4059,86	7100,5	1,748952	1660,059	595,6239	1285,51
21	2014	1301209	Coari	2,55E+08	20533679	2,74E+08	82209	82209	3,11E+09	4059,86	7100,5	1,748952	3101,101	249,7741	3332,88
22	2014	1301308	Codajás	41665010	16294483	31508193	26242	26242	6,52E+08	4059,86	7100,5	1,748952	1587,722	620,9315	1200,67
23	2014	1301407	Eirunepé	63714873	18622267	47278514	33580	33580	3,83E+08	4059,86	7100,5	1,748952	1897,405	554,5642	1407,93
24	2014	1301506	Envira	40864482	13966700	28641187	18422	18422	1,25E+08	4059,86	7100,5	1,748952	2218,243	758,1533	1554,72
25	2014	1301605	Fonte Boa	46177192	9477083	43767594	21295	21295	1,52E+08	4059,86	7100,5	1,748952	2168,452	445,0379	2055,29
26	2014	1301654	Guajará	34155488	11638917	26638373	15561	15561	1E+08	4059,86	7100,5	1,748952	2194,942	747,9543	1711,86
27	2014	1301704	Humaitá	76913863	15795138	74775152	50230	50230	4,23E+08	4059,86	7100,5	1,748952	1531,234	314,4563	1488,65
28	2014	1301803	Ipixuna	37312316	16294483	27050756	26118	26118	1,12E+08	4059,86	7100,5	1,748952	1428,605	623,8794	1035,71
29	2014	1301852	Irlanduba	89668144	15837514	80762983	45250	45250	5,3E+08	4059,86	7100,5	1,748952	1981,616	350,0003	1784,81
30	2014	1301902	Itacoatiara	1,95E+08	34916750	1,66E+08	95714	95714	1,59E+09	4059,86	7100,5	1,748952	2033,169	364,803	1729,80
31	2014	1301951	Itamarati		6983350	19840671	8205	8205	75976000	4059,86	7100,5	1,748952		851,1091	2418,1
32	2014	1302009	Itapiranga	24632222	6983350	17308582	8864	8864	80542000	4059,86	7100,5	1,748952	2778,906	787,8328	1952,68
33	2014	1302108	Japurá	28728379	6983350	20905746	5599	5599	58162000	4059,86	7100,5	1,748952	5130,984	1247,249	3733,83
34	2014	1302207	Juruá	32110976	9311133	25495283	12807	12807	84834000	4059,86	7100,5	1,748952	2507,299	727,0347	1990,7
35	2014	1302306	Jutai	38640740	9477083	37565993	16977	16977	1,82E+08	4059,86	7100,5	1,748952	2276,064	558,2307	2212,75
36	2014	1302405	Lábrea	73998614	20950050	59026435	42439	42439	3,69E+08	4059,86	7100,5	1,748952	1743,647	493,6509	1390,85
37	2014	1302504	Manacapur	1,24E+08	23692706	1,22E+08	92996	92996	1,26E+09	4059,86	7100,5	1,748952	1330,846	254,7712	1314,99
38	2014	1302553	Manaquiri	45970825	16294483	35911501	27480	27480	2,07E+08	4059,86	7100,5	1,748952	1672,883	592,9579	1306,82
39	2014	1302603	Manaus	4,37E+09	4,34E+08	3,24E+09	2020301	2020301	6,74E+10	4059,86	7100,5	1,748952	2160,779	214,7199	1604,63
40	2014	1302702	Manicoré	69508576	17374651	65133825	52200	52200	4,6E+08	4059,86	7100,5	1,748952	1331,582	332,8477	1247,77
41	2014	1302801	Maraá	38075037	9477083	37708137	18367	18367	1,18E+08	4059,86	7100,5	1,748952	2073,013	515,9842	2053,03
42	2014	1302900	Maués	1,18E+08	25605617	96283953	58834	58834	3,89E+08	4059,86	7100,5	1,748952	2012,949	435,218	1636,53

7. ADAPTAÇÃO E REUTILIZAÇÃO DO PIPELINE

Este capítulo apresenta orientações para adaptação, expansão e reutilização do pipeline descrito ao longo deste manual. A estrutura modular adotada no projeto permite que as rotinas sejam ajustadas para diferentes contextos de pesquisa, unidades geográficas, períodos temporais e conjuntos de variáveis.

O objetivo desta etapa é demonstrar como o pipeline pode ser reaproveitado em novos projetos de construção de bases de dados aplicadas, preservando a lógica de organização, integração e validação apresentada nos capítulos anteriores.

As adaptações podem envolver:

- Alteração da unidade geográfica de análise;
- Inclusão de novas variáveis climáticas;
- Inclusão de novos indicadores econômicos;
- Expansão do período temporal;
- Utilização de outras bases públicas;
- Reorganização da estrutura de diretórios.

A modularização dos scripts e a centralização dos parâmetros globais no `00_config.R` facilitam essas adaptações, reduzindo a necessidade de alterações extensas ao longo do pipeline.

7.1 ADAPTAÇÃO PARA OUTROS RECORTES ESPACIAIS

O pipeline apresentado neste manual foi estruturado para os municípios do estado do Amazonas. Entretanto, sua lógica pode ser adaptada para outros estados, regiões ou conjuntos de municípios mediante alterações pontuais na configuração do projeto.

As principais adaptações envolvem:

- Substituição do shapefile municipal;
- Alteração da unidade federativa;
- Atualização das bases fiscais;
- Revisão das consultas utilizadas nas APIs públicas.

Principais ajustes para adaptação espacial do pipeline

Elemento	Adaptação necessária
Shapefile	Substituir pelo recorte espacial desejado
econ_uf	Alterar a unidade federativa utilizada
Bases fiscais	Utilizar arquivos correspondentes ao novo recorte
Consultas SIDRA	Revisar filtros geográficos
Logs e outputs	Atualizar nomenclaturas do projeto



Uso de IA

Ferramentas de IA podem auxiliar na adaptação dos scripts para outros estados, municípios ou recortes regionais

Exemplos de prompts:

- “Como adaptar este pipeline para outro estado brasileiro?”
- “Como substituir o shapefile utilizado no projeto?”
- “Como alterar os filtros geográficos das consultas SIDRA?”

7.2 INCLUSÃO DE NOVAS VARIÁVEIS CLIMÁTICAS

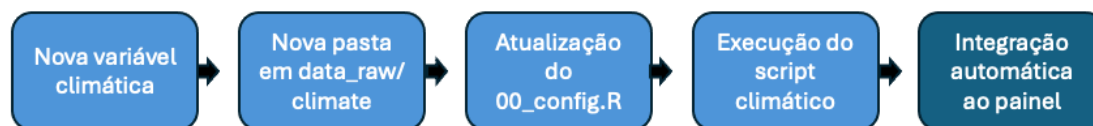
O pipeline climático pode ser expandido para inclusão de novas variáveis ambientais provenientes do TerraClimate ou de outras bases raster compatíveis com a estrutura do projeto.

A inclusão de novas variáveis normalmente exige:

- Criação de uma nova subpasta em `data_raw/climate`;
- Atualização da lista de variáveis climáticas no `00_config.R`;
- Execução do script de extração climática;
- Consolidação automática do novo painel.

A Figura 5 apresenta um exemplo simplificado do fluxo de inclusão de novas variáveis climáticas.

Figura 5 – Fluxo simplificado para inclusão de novas variáveis climáticas



Uso de IA

Ferramentas de IA podem auxiliar na adaptação das rotinas raster e interpretação de erros relacionados à leitura de novas variáveis climáticas.

Exemplos de prompts:

- “Como adaptar um script raster para uma nova variável climática?”
- “Como verificar compatibilidade temporal em arquivos NetCDF?”
- “Como identificar valores ausentes em séries climáticas no R?”

7.3 INCLUSÃO DE NOVOS INDICADORES ECONÔMICOS

O pipeline econômico também pode ser expandido para incorporar novos indicadores fiscais, econômicos ou sociais provenientes de bases públicas nacionais.

A inclusão de novos indicadores geralmente envolve:

- Identificação da fonte de dados;
- Adaptação dos filtros de leitura;
- Padronização das chaves municipais;
- Integração ao painel econômico final.

Entre os indicadores que podem ser incorporados, destacam-se:

- Indicadores educacionais;
- Indicadores de saúde;
- Transferências governamentais;

- Indicadores de emprego;
- Dados agropecuários;
- Informações ambientais complementares.

Procedimentos recomendados

Para inclusão de novos indicadores econômicos, recomenda-se:

1. Verificar a periodicidade dos dados;
2. Validar a cobertura municipal da base;
3. Padronizar as chaves geográficas;
4. Revisar inconsistências temporais;
5. Documentar as novas variáveis no dicionário final.

7.4 ALTERAÇÃO DO PERÍODO TEMPORAL DO PAINEL

O período temporal do painel pode ser alterado diretamente no script de configuração inicial do projeto. Essa adaptação permite expandir ou restringir o horizonte temporal utilizado na construção da base.

As alterações são realizadas principalmente por meio da atualização do parâmetro `panel_years` no script `00_config.R`.

Após a modificação do período de análise, recomenda-se:

- Revisar a disponibilidade dos arquivos climáticos;
- Verificar a cobertura das bases fiscais;
- Validar a existência de informações econômicas para todos os anos;
- Executar novamente as rotinas de auditoria.

Exemplo de alteração do período temporal

```
R/  
panel_years <- 2010:2025
```



Uso de IA

Ferramentas de IA podem auxiliar na adaptação automática do pipeline para novos períodos temporais.

Exemplos de prompts:

- “Como adaptar um painel municipal para outro período temporal no R?”
- “Como verificar anos ausentes em uma base anual no R?”
- “Como a cobertura temporal de um painel no R?”

7.5 RECOMENDAÇÕES PARA REUTILIZAÇÃO DO PIPELINE

A reutilização do pipeline em novos projetos requer atenção à organização do ambiente de trabalho, padronização dos arquivos e documentação das alterações realizadas.

Para garantir maior estabilidade operacional, recomenda-se:

- Manter a estrutura de diretórios do projeto;
- Utilizar nomenclaturas padronizadas;
- Documentar alterações realizadas nos scripts;
- Executar auditorias após modificações relevantes;
- Preservar logs e arquivos intermediários;
- Utilizar backups das bases originais.

Também é recomendável que as adaptações sejam realizadas de forma incremental, validando cada etapa antes da execução completa do pipeline.

Boas práticas recomendadas

Prática	Objetivo
Manter estrutura de pastas	Facilitar localização dos arquivos
Utilizar logs	Melhorar rastreabilidade das etapas
Validar merges	Evitar inconsistências estruturais
Preservar arquivos brutos	Garantir reprodutibilidade
Atualizar dicionário	Documentar alterações realizadas

 Uso de IA

Ferramentas de IA podem auxiliar na revisão de scripts, documentação das adaptações e interpretação de inconsistências operacionais

Exemplos de prompts:

- “Como documentar alterações em um pipeline de dados no R?”
- “Como validar a reutilização de um script em outro projeto?”
- “Como criar logs automáticos no R?”

CONSIDERAÇÕES FINAIS

Este manual técnico apresentou um pipeline aplicado para construção, organização, integração e validação de bases de dados econômicas e climáticas em estrutura de painel municipal anual. A proposta foi desenvolvida com foco na reprodutibilidade, modularidade e adaptação do processo, permitindo que diferentes usuários possam replicar ou expandir as rotinas descritas em novos projetos de pesquisa e aplicações práticas.

Ao longo do documento, foram apresentadas as principais etapas necessárias para operacionalização do pipeline, incluindo:

- Organização do ambiente de trabalho;
- Estruturação dos diretórios do projeto;
- Extração e tratamento de dados climáticos;
- Construção do painel econômico;
- Integração das bases;
- Geração do painel consolidado;
- Auditoria e validação estrutural dos dados.

A utilização de scripts modulares em linguagem R permitiu automatizar diferentes etapas do processo, reduzindo atividades repetitivas e ampliando a rastreabilidade das rotinas executadas. Além disso, a centralização dos parâmetros do projeto no script 00_config.R contribuiu para maior flexibilidade operacional e facilidade de adaptação do pipeline.

O manual também buscou incorporar uma abordagem prática e acessível, conciliando procedimentos técnicos de construção de bases de dados com estratégias de documentação e organização do fluxo operacional. Nesse contexto, a utilização complementar de ferramentas de Inteligência Artificial foi apresentada como suporte auxiliar para interpretação de erros, adaptação de scripts, validação de procedimentos e automatização de tarefas específicas do pipeline.

O pipeline apresentado neste manual foi originalmente desenvolvido no contexto da dissertação de mestrado intitulada “IMPACTOS ECONÔMICOS DAS MUDANÇAS CLIMÁTICAS NOS MUNICÍPIOS DO ESTADO DO AMAZONAS: UMA PERSPECTIVA LONGITUDINAL”, vinculada ao Programa de Pós-Graduação em Economia da Universidade Federal do Amazonas (PPGECO/UFAM). A adaptação das

rotinas para este Produto Técnico-Tecnológico buscou ampliar a aplicabilidade e a reprodutibilidade do processo de construção da base de dados, permitindo sua reutilização em diferentes contextos de pesquisa e aplicações práticas.

A estrutura proposta permite adaptações para diferentes contextos territoriais, períodos temporais e conjuntos de variáveis, possibilitando sua utilização em pesquisas aplicadas, estudos econômicos, análises ambientais, desenvolvimento de indicadores e projetos institucionais baseados em integração de dados públicos.

Por fim, espera-se que este manual contribua para fortalecimento de práticas voltadas à reprodutibilidade, organização e padronização de pipelines de dados aplicados, auxiliando pesquisadores, estudantes e profissionais interessados na construção de bases integradas para análise quantitativa.

REFERÊNCIAS

ABATZOGLOU, John T. *et al.* TerraClimate, a high-resolution global dataset of monthly climate and climatic water balance from 1958–2015. **Scientific Data**, [s. l.], v. 5, n. 1, p. 170191, 2018.

GOOGLE. **Gemini**. [S. l.], 2026. Plataforma de inteligência artificial. Disponível em: <https://gemini.google.com>. Acesso em: 3 maio 2026.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. Produto Interno Bruto dos Municípios 2010-2021: base de dados. , 2023. Disponível em: <https://sidra.ibge.gov.br/tabela/5938>. Acesso em: 20 maio 2025.

OPENAI. **ChatGPT**. [S. l.], 2026. Plataforma de inteligência artificial. Disponível em: <https://chatgpt.com/>. Acesso em: 1 maio 2026.

PENG, Roger D. Reproducible Research in Computational Science. **Science**, [s. l.], v. 334, n. 6060, p. 1226–1227, 2011.

POSIT SOFTWARE, PBC. **RStudio**. versão 2025.05.1. [S. l.]: [s. d.], 2025.

R CORE TEAM. **R: A language and environment for statistical computing**. versão 4.4.1. Viena, Áustria: R Foundation for Statistical Computing, 2024. Disponível em: <https://www.R-project.org/>. Acesso em: 3 abr. 2026.

SECRETARIA DO TESOURO NACIONAL. Relatórios Contábeis e Fiscais de Estados, DF e Municípios. Siconfi: Sistema de Informações Contábeis e Fiscais do Setor Público Brasileiro., 2026. Disponível em: <https://www.tesourotransparente.gov.br/visualizacao/meu-siconfi-extrator-de-dados-do-siconfi>. Acesso em: 10 dez. 2025.

WILKINSON, Mark D. *et al.* The FAIR Guiding Principles for scientific data management and stewardship. **Scientific Data**, [s. l.], v. 3, n. 1, p. 160018, 2016.

APÊNDICE A – Script de configuração inicial do projeto (00_config.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
# ARQUIVO: 00_config.R
#
# FUNÇÃO:
# Configurar o ambiente do projeto,
# carregar pacotes,
# definir diretórios, parâmetros globais
# e validar arquivos-base.
#####
#####

#
=====
=====
# 1. PACOTES NECESSÁRIOS
#
=====
=====

required_packages <- c(
  "terra", "sf", "exactextractr",
  "dplyr", "tidyr", "purrr", "readr",
  "stringr",
  "janitor", "openxlsx", "rstudioapi",
  "tibble",
  "sidrar", "writexl"
)

to_install <-
setdiff(required_packages,
rownames(installed.packages()))

if (length(to_install) > 0) {
  install.packages(to_install,
dependencies = TRUE)
}

invisible(lapply(required_packages,
library, character.only = TRUE))

cat("Pacotes verificados e carregados
com sucesso.\n\n")

#
=====
=====
# 2. DIRETÓRIO RAIZ DO PROJETO
#
=====
=====

if (!exists("root_dir")) {
  if (rstudioapi::isAvailable()) {
    root_dir <-
rstudioapi::selectDirectory(
caption = "Selecione a pasta raiz
do projeto: Manual Técnico (PTT)"
  )
  } else {
    root_dir <- choose.dir()
  }
}

root_dir <- normalizePath(root_dir,
winslash = "/", mustWork = FALSE)

cat("Diretório raiz selecionado:\n",
root_dir, "\n\n")

#
=====
=====
# 3. DIRETÓRIOS PRINCIPAIS
#
=====
=====

data_raw_dir <- file.path(root_dir,
"data_raw")
data_processed_dir <-
file.path(root_dir, "data_processed")
scripts_dir <- file.path(root_dir,
"scripts")
shapefiles_dir <-
file.path(root_dir, "shapefiles")

complete_outputs_dir <-
file.path(data_processed_dir,
"complete_outputs")
complete_logs_dir <-
file.path(complete_outputs_dir, "logs")

dirs_base <- c(
  data_raw_dir,
  data_processed_dir,
  scripts_dir,
  shapefiles_dir,
  complete_outputs_dir,
  complete_logs_dir
)

invisible(lapply(dirs_base, dir.create,
recursive = TRUE, showWarnings = FALSE))

cat("Estrutura principal de pastas
verificada/criada.\n\n")

#
=====
=====
# 4. DIRETÓRIOS DOS DADOS CLIMÁTICOS
#
=====
=====

raw_climate_dir <-
file.path(data_raw_dir, "climate")
```

```

processed_climate_dir      <-      econ_raw_dir,
file.path(data_processed_dir,
"climate_outputs")

monthly_output_dir        <-      econ_processed_dir,
file.path(processed_climate_dir,
"extracted_monthly")
annual_output_dir         <-      econ_receita_dir,
file.path(processed_climate_dir,
"annual")
final_output_dir          <-      econ_desp_orc_dir,
file.path(processed_climate_dir,
"annual")
climate_logs_dir          <-      econ_fpm_dir,
file.path(processed_climate_dir,
"logs")

dirs_climate <- c(
  raw_climate_dir,
  processed_climate_dir,
  monthly_output_dir,
  annual_output_dir,
  final_output_dir,
  climate_logs_dir
)

invisible(lapply(dirs_climate,
  dir.create, recursive = TRUE,
  showWarnings = FALSE))

cat("Estrutura de pastas climáticas
verificada/criada.\n\n")

#
=====
# 5. DIRETÓRIOS DOS DADOS ECONÔMICOS
#
=====

econ_raw_dir      <-      econ_raw_dir, "econ")
file.path(data_raw_dir,
econ_processed_dir <-      econ_processed_dir,
file.path(data_processed_dir,
"econ_outputs")

econ_receita_dir  <-      econ_receita_dir,
file.path(econ_raw_dir,
"receita_orcamentaria")
econ_desp_orc_dir <-      econ_desp_orc_dir,
file.path(econ_raw_dir,
"despesa_orcamentaria")
econ_fpm_dir      <-      econ_fpm_dir,
file.path(econ_raw_dir, "fpm")

econ_annual_dir   <-      econ_processed_dir, "annual")
econ_logs_dir     <-      econ_processed_dir, "logs")
econ_panel_dir    <-      econ_processed_dir, "panel")
econ_final_dir    <-      econ_processed_dir, "final")

file.path(econ_processed_dir, "final")

dirs_econ <- c(
  econ_raw_dir,
  econ_processed_dir,
  econ_receita_dir,
  econ_desp_orc_dir,
  econ_fpm_dir,
  econ_annual_dir,
  econ_logs_dir,
  econ_panel_dir,
  econ_final_dir
)

invisible(lapply(dirs_econ, dir.create,
  recursive = TRUE, showWarnings = FALSE))

cat("Estrutura de pastas econômicas
verificada/criada.\n\n")

#
=====
# 6. PARÂMETROS GLOBAIS
#
=====

panel_years <- 2014:2024

econ_years <- panel_years
econ_uf    <- "AM"
muni_key   <- "ibge7"

target_crs_epsg <- 4326

cat("Parâmetros globais definidos:\n")
cat("Painel:", min(panel_years), "-",
max(panel_years), "\n")
cat("UF econômica:", econ_uf, "\n")
cat("Chave municipal:", muni_key,
"\n\n")

#
=====
# 7. VARIÁVEIS ECONÔMICAS-ALVO
#
=====

econ_vars_target <- c(
  "RC_LIQ",
  "FPM_LIQ",
  "DC",
  "PIB",
  "POP",
  "PIB_PC"
)

econ_files_raw <- list(
  receita_orc =
"REC_ORC_AM_2014_2024.csv",
  desp_orc =
"DESP_ORC_AM_2014_2024.csv"
)

econ_uf_ibge <- 13

```

```

sidra_tables <- list(
  pib_table = 5938,
  pib_var   = 37,
  pop_table = 6579
)

econ_units <- tibble::tibble(
  var = c("RC_LIQ", "FPM_LIQ", "DC",
"PIB", "POP", "PIB_PC"),
  unidade = c(
    "R$ correntes",
    "R$ correntes",
    "R$ correntes",
    "R$ correntes",
    "habitantes",
    "R$/hab"
  )
)

cat("Parâmetros                econômicos
carregados.\n\n")

#
=====
# 8. VARIÁVEIS CLIMÁTICAS
#
=====

climate_variables <- tibble::tibble(
  var_name = c(
    "pdsi",
    "ppt",
    "tmax",
    "tmin",
    "aet",
    "pet",
    "def",
    "soil",
    "ws",
    "q"
  ),
  var_type = c(
    "stock",
    "flow",
    "stock",
    "stock",
    "stock",
    "stock",
    "stock",
    "stock",
    "stock",
    "flow"
  )
)

cat("Variáveis                climáticas
carregadas.\n\n")

#
=====
# 9. SHAPEFILE DOS MUNICÍPIOS
#
=====

shapefile_path <-
file.path(shapefiles_dir,
"am_municipios.shp")

if (file.exists(shapefile_path)) {

  am_muni <-
sf::st_read(shapefile_path, quiet =
TRUE)

  am_muni <-
janitor::clean_names(am_muni)

  if (sf::st_crs(am_muni)$epsg !=
target_crs_epsg) {
    am_muni <-
sf::st_transform(am_muni,
target_crs_epsg)
    cat("Shapefile transformado para
EPSG:", target_crs_epsg, "\n")
  }

  n_municipios <- nrow(am_muni)

  if (n_municipios != 62) {
    warning("Número de municípios
diferente de 62: ", n_municipios)
  } else {
    cat("Shapefile validado: 62
municípios confirmados.\n")
  }

} else {

  warning(
    "Shapefile ainda não encontrado em:
",
    shapefile_path,
    "\nInclua o arquivo
am_municipios.shp na pasta shapefiles
antes de executar os scripts
climáticos."
  )

  am_muni <- NULL
}

cat("\nArquivo 00_config.R carregado com
sucesso.\n")

```

APÊNDICE B – Script de extração das variáveis climáticas (01_extract_one_variable.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
# SCRIPT: 01_extract_one_variable.R
#
# FUNÇÃO:
#   Extrair variáveis climáticas do
#   TerraClimate para os
#   municípios do Amazonas e agregá-las
#   para frequência anual.
#
# OBJETIVO:
#   - Ler arquivos NetCDF (.nc);
#   - Realizar extração espacial
#     municipal;
#   - Organizar os dados em estrutura
#     tabular;
#   - Agregar os valores mensais para
#     frequência anual;
#   - Exportar painéis climáticos anuais.
#
# ENTRADAS:
#   - data_raw/climate/<variavel>/*.nc
#   - shapefiles/am_municipios.shp
#
# SAÍDAS:
#   - climate_outputs/extracted_monthly/
#   - climate_outputs/annual/
#   - climate_outputs/final_panel_2014_2024/
#   - climate_outputs/logs/
#
# OBSERVAÇÃO:
#   O script foi estruturado para
#   reutilização em diferentes
#   variáveis climáticas do
#   TerraClimate.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
    quietly = TRUE)) {
    root_dir <-
      rstudioapi::selectDirectory(
        caption = "Selecione a pasta raiz
do projeto (pasta 'Dados')"
      )
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
  "scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(climate_logs_dir, recursive
= TRUE, showWarnings = FALSE)

log_file <- file.path(
  climate_logs_dir,
  "extract_climate_log.txt"
)

if (file.exists(log_file))
  file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "] ",
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início do Script
01_extract_one_variable.R")
log_line("root_dir =", root_dir)
log_line("Período do painel =",
min(panel_years), "-",
max(panel_years))
```

```

#
=====
# 1. VALIDAR SHAPEFILE
#
=====

if (!exists("am_muni")) {
  stop("Objeto am_muni não encontrado.")
}

if (!inherits(am_muni, "sf")) {
  stop("am_muni não é um objeto sf.")
}

log_line("Shapefile carregado com sucesso.")
log_line("Municípios =", nrow(am_muni))

#
=====
# 2. IDENTIFICAR CHAVE MUNICIPAL
#
=====

ibge_col <- {

  cand <- c(
    "ibge7",
    "cd_mun",
    "cd_mun7",
    "code_muni",
    "geocodigo",
    "cd_geocmu"
  )

  cand[cand %in% names(am_muni)][1]
}

if (is.na(ibge_col) || length(ibge_col) == 0) {
  stop(
    "Não encontrei coluna IBGE no shapefile.\n",
    paste(names(am_muni), collapse = ",")
  )
}

am_muni <- am_muni |>
  dplyr::mutate(
    ibge7 = stringr::str_pad(
      as.character(.data[[ibge_col]]),
      width = 7,
      side = "left",
      pad = "0"
    )
  )

```

```

log_line("Coluna IBGE identificada:",
  ibge_col)

```

```

#
=====
# 3. FUNÇÃO DE EXTRAÇÃO
#
=====

process_terraclimate_variable <-
function(var_name,

  var_type = c("flow", "stock")) {

  var_type <- match.arg(var_type)

  log_line("-----")
  log_line("Processando variável:",
    var_name)
  log_line("Tipo de agregação:",
    var_type)

  var_dir <- file.path(raw_climate_dir,
    var_name)

  if (!dir.exists(var_dir)) {
    stop("Diretório não encontrado: ",
      var_dir)
  }

  #
  =====
  # IDENTIFICAR ARQUIVOS .NC
  #
  =====

  files_all <- list.files(
    var_dir,
    pattern = "\\\\.nc$",
    full.names = TRUE
  )

  if (length(files_all) == 0) {
    stop("Nenhum arquivo .nc encontrado em: ", var_dir)
  }

  pattern_expected <- paste0(
    "^TerraClimate_",
    var_name,
    "_((\\d{4}))\\.nc$"
  )

  yrs <- stringr::str_match(
    basename(files_all),
    pattern_expected
  )[,2]

```

```

files_invalid <-
files_all[is.na(yrs)]

if (length(files_invalid) > 0) {
  warning(
    "Arquivos fora do padrão serão
    ignorados:\n",
    paste(basename(files_invalid),
    collapse = "\n")
  )

  log_line(
    "Arquivos ignorados:",
    paste(basename(files_invalid),
    collapse = " | ")
  )
}

files <- files_all[!is.na(yrs)]
yrs <- as.integer(yrs[!is.na(yrs)])

ord <- order(yrs)

files <- files[ord]
yrs <- yrs[ord]

log_line("Arquivos válidos =",
length(files))

#
=====
# EXTRAÇÃO MENSAL
#
=====

monthly_list <- vector("list",
length(files))

for (i in seq_along(files)) {
  file_i <- files[i]
  year_i <- yrs[i]

  log_line("Lendo:",
  basename(file_i))

  r <- terra::rast(file_i)

  n_layers <- terra::nlyr(r)

  dates_i <- seq.Date(
    from = as.Date(paste0(year_i, "-
    01-01")),
    by = "month",
    length.out = n_layers
  )

  ext_i <- exactextractr::exact_extract(
    r,
    am_muni,
    fun = "mean",

    progress = FALSE
  )

  ext_i <- as.data.frame(ext_i)

  names(ext_i) <- paste0("m_",
  seq_len(ncol(ext_i)))

  ext_i <- dplyr::bind_cols(
    am_muni |>
    sf::st_drop_geometry() |>
    dplyr::select(ibge7),
    ext_i
  )

  ext_long <- ext_i |>
  tidyr::pivot_longer(
    cols = starts_with("m_"),
    names_to = "layer",
    values_to = var_name
  ) |>
  dplyr::mutate(
    month_id =
    readr::parse_number(layer),
    data = dates_i[month_id],
    ano = lubridate::year(data),
    mes = lubridate::month(data)
  ) |>
  dplyr::select(
    ibge7,
    data,
    ano,
    mes,
    !!var_name
  )

  monthly_list[[i]] <- ext_long
}

#
=====
# CONSOLIDAR BASE MENSAL
#
=====

monthly_df <-
dplyr::bind_rows(monthly_list) |>
dplyr::arrange(ibge7, data)

monthly_out <- file.path(
  monthly_output_dir,
  paste0(var_name,
  "_monthly_2014_2024.csv")
)

readr::write_csv(monthly_df,
monthly_out)

log_line("Base mensal exportada:",
monthly_out)

```

```

#
=====
# AGREGAÇÃO ANUAL
#
=====

annual_df <- monthly_df |>
  dplyr::group_by(ibge7, ano) |>
  dplyr::summarise(

    value = if (var_type == "flow") {
      sum(.data[[var_name]], na.rm =
TRUE)
    } else {
      mean(.data[[var_name]], na.rm =
TRUE)
    },

    n_meses = dplyr::n(),
    .groups = "drop"
  )

names(annual_df)[names(annual_df) ==
"value"] <- paste0(var_name, "_anual")

annual_df <- annual_df |>
  dplyr::arrange(ibge7, ano)

#
=====
# VALIDAR DUPLICIDADES
#
=====

dup_panel <- annual_df |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

if (nrow(dup_panel) > 0) {

  readr::write_csv(
    dup_panel,
    file.path(
      climate_logs_dir,
      paste0(var_name,
"_duplicates.csv")
    )
  )

  stop("Duplicidades encontradas em ",
var_name)
}

#
=====
# EXPORTAR BASE ANUAL
#
=====

annual_out <- file.path(
  annual_output_dir,
  paste0(var_name,
"_annual_2014_2024.csv")
)

readr::write_csv(annual_df,
annual_out)

log_line("Base anual exportada:",
annual_out)

#
=====
# EXPORTAR PAINEL FINAL
#
=====

panel_out <- file.path(
  final_output_dir,
  paste0(var_name,
"_panel_2014_2024.csv")
)

readr::write_csv(annual_df,
panel_out)

log_line("Painel final exportado:",
panel_out)

return(invisible(annual_df))
}

#
=====
# 4. EXECUTAR VARIÁVEIS CLIMÁTICAS
#
=====

for (i in
seq_len(nrow(climate_variables))) {

  var_i
climate_variables$var_name[i]
type_i
climate_variables$var_type[i]

  tryCatch(
    {
      process_terraclimate_variable(
        var_name = var_i,
        var_type = type_i
      )
    },
    error = function(e) {

```

```
        log_line("ERRO em", var_i, ":",
e$message)
```

```
        message("\nErro ao processar
variável: ", var_i)
        message(e$message)
    }
}
```

```
#
=====
=====
# 5. FINALIZAÇÃO
```

```
#
=====
=====
```

```
log_line("Script 01 concluído.")
log_line("Log salvo em:", log_file)
```

```
cat("\n=====
=====\\n")
cat("SCRIPT 01 CONCLUÍDO\\n")
cat("Log salvo em:\\n")
cat(log_file, "\\n")
cat("=====
=====\\n")
```

APÊNDICE C – Script de construção da temperatura média anual (02_build_tmean.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
# SCRIPT: 02_build_tmean.R
#
# FUNÇÃO:
#   Construir a variável de temperatura
#   média anual
#   (tmean) a partir das variáveis tmax
#   e tmin.
#
# OBJETIVO:
#   - Ler os painéis anuais de tmax e
#   tmin;
#   - Calcular tmean anual;
#   - Validar a estrutura do painel;
#   - Exportar o painel final de
#   temperatura média.
#
# ENTRADAS:
#   - tmax_panel_2014_2024.csv
#   - tmin_panel_2014_2024.csv
#
# SAÍDAS:
#   - tmean_panel_2014_2024.csv
#   - tmean_build_log.txt
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
    quietly = TRUE)) {
    root_dir <-
rstudioapi::selectDirectory(
  caption = "Selecione a pasta raiz
do projeto"
)
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
"scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(climate_logs_dir, recursive
= TRUE, showWarnings = FALSE)

log_file <- file.path(
  climate_logs_dir,
  "tmean_build_log.txt"
)

if (file.exists(log_file))
file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "]",
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início do Script
02_build_tmean.R")

#
=====
=====
# 1. DEFINIR CAMINHOS
#
=====
=====

tmax_path <- file.path(
  final_output_dir,
  "tmax_panel_2014_2024.csv"
)

tmin_path <- file.path(
```

```

    final_output_dir,
    "tmin_panel_2014_2024.csv"
  )

  if (!file.exists(tmax_path)) {
    stop("Arquivo tmax não encontrado.")
  }

  if (!file.exists(tmin_path)) {
    stop("Arquivo tmin não encontrado.")
  }

  #
  =====
  # 2. LER PAINÉIS
  #
  =====

  tmax_df <- readr::read_csv(
    tmax_path,
    show_col_types = FALSE
  )

  tmin_df <- readr::read_csv(
    tmin_path,
    show_col_types = FALSE
  )

  log_line("tmax linhas =", nrow(tmax_df))
  log_line("tmin linhas =", nrow(tmin_df))

  #
  =====
  # 3. PADRONIZAR CHAVES
  #
  =====

  tmax_df <- tmax_df |>
    dplyr::mutate(
      ibge7 = stringr::str_pad(
        as.character(ibge7),
        width = 7,
        side = "left",
        pad = "0"
      ),
      ano = as.integer(ano)
    )

  tmin_df <- tmin_df |>
    dplyr::mutate(
      ibge7 = stringr::str_pad(
        as.character(ibge7),
        width = 7,
        side = "left",
        pad = "0"
      ),
      ano = as.integer(ano)
    )

```

```

#
=====
# 4. VALIDAR COBERTURA
#
=====

keys_tmax <- tmax_df |>
  dplyr::distinct(ibge7, ano)

keys_tmin <- tmin_df |>
  dplyr::distinct(ibge7, ano)

missing_keys <- dplyr::anti_join(
  keys_tmax,
  keys_tmin,
  by = c("ibge7", "ano")
)

if (nrow(missing_keys) > 0) {

  readr::write_csv(
    missing_keys,
    file.path(
      climate_logs_dir,
      "tmean_missing_keys.csv"
    )
  )

  stop("Cobertura incompatível entre
tmax e tmin.")
}

log_line("Cobertura      validada      com
sucesso.")

#
=====
# 5. CONSTRUIR TMEAN
#
=====

tmean_df <- tmax_df |>
  dplyr::select(
    ibge7,
    ano,
    tmax_anual
  ) |>
  dplyr::inner_join(

    tmin_df |>
      dplyr::select(
        ibge7,
        ano,
        tmin_anual
      ),

    by = c("ibge7", "ano")
  ) |>
  dplyr::mutate(
    tmean_anual = (tmax_anual +
tmin_anual) / 2

```

```

) |>
dplyr::select(
  ibge7,
  ano,
  tmean_anual
) |>
dplyr::arrange(
  ano,
  ibge7
)

log_line("tmean      construída      com
sucesso.")

#
=====
# 6. VALIDAR DUPLICIDADES
#
=====

dup_panel <- tmean_df |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

if (nrow(dup_panel) > 0) {

  readr::write_csv(
    dup_panel,
    file.path(
      climate_logs_dir,
      "tmean_duplicates.csv"
    )
  )

  stop("Duplicidades encontradas em
tmean.")
}

#
=====
# 7. EXPORTAR
#
=====

tmean_out <- file.path(
  final_output_dir,
  "tmean_panel_2014_2024.csv"
)

readr::write_csv(
  tmean_df,
  tmean_out
)

log_line("Painel      exportado:",
tmean_out)

#
=====
# 8. FINALIZAÇÃO
#
=====

log_line("Script 02 concluído.")

cat("\n=====\\n")
cat("SCRIPT 02 CONCLUÍDO\\n")
cat("Arquivo gerado:\\n")
cat(tmean_out, "\\n")
cat("=====\\n")

```

APÊNDICE D – Script de consolidação do painel climático (03_merge_climate_all.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
# SCRIPT: 03_merge_climate_all.R
#
# FUNÇÃO:
#   Consolidar os painéis climáticos
#   anuais em uma única base.
#
# OBJETIVO:
#   - Ler os painéis climáticos
#   individuais gerados pelos Scripts 01 e
#   02;
#   - Validar chaves municipais e
#   temporais;
#   - Integrar as variáveis climáticas
#   por ibge7 + ano;
#   - Exportar o painel climático
#   consolidado.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
    quietly = TRUE)) {
    root_dir <-
      rstudioapi::selectDirectory(
        caption = "Selecione a pasta raiz
do projeto"
      )
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
  "scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
  encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(climate_logs_dir, recursive
  = TRUE, showWarnings = FALSE)

log_file <- file.path(climate_logs_dir,
  "climate_merge_log.txt")

if (file.exists(log_file))
  file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "]",
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
    = TRUE)
}

log_line("Início do Script
03_merge_climate_all.R")
log_line("root_dir =", root_dir)
log_line("Período do painel =",
  min(panel_years), "-",
  max(panel_years))

#
=====
=====
# 1. DEFINIR VARIÁVEIS CLIMÁTICAS
# ESPERADAS
#
=====
=====

vars_expected <-
  climate_variables$var_name

# A variável tmean é construída
separadamente no Script 02.
vars_all <- c(vars_expected, "tmean")

panel_files <- file.path(
  final_output_dir,
  paste0(vars_all,
    "_panel_2014_2024.csv")
)

names(panel_files) <- vars_all
```

```

missing_files <-
panel_files[!file.exists(panel_files)]

if (length(missing_files) > 0) {
  stop(
    "Arquivos climáticos ausentes:\n",
    paste(missing_files, collapse =
"\n")
  )
}

log_line("Todos os arquivos climáticos
foram encontrados.")
log_line("Variáveis:", paste(vars_all,
collapse = ", "))

#
=====
# 2. FUNÇÃO PARA LER E VALIDAR CADA
PAINEL
#
=====

read_and_check_panel <- function(path,
var_name) {

  df <- readr::read_csv(path,
show_col_types = FALSE) |>
  dplyr::mutate(
    ibge7 = stringr::str_pad(
      as.character(ibge7),
      width = 7,
      side = "left",
      pad = "0"
    ),
    ano = as.integer(ano)
  )

  required_cols <- c("ibge7", "ano",
paste0(var_name, "_anual"))

  missing_cols <-
setdiff(required_cols, names(df))

  if (length(missing_cols) > 0) {
    stop(
      "O arquivo ",
      basename(path),
      " não possui as colunas esperadas:
",
      paste(missing_cols, collapse = ",
")
    )
  }

  df <- df |>
  dplyr::select(
    ibge7,
    ano,
    dplyr::all_of(paste0(var_name,
"_anual"))
  )

  dup <- df |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

  if (nrow(dup) > 0) {
    readr::write_csv(
      dup,
      file.path(climate_logs_dir,
paste0(var_name,
"_merge_duplicates.csv"))
    )
    stop("Duplicidades encontradas em ",
basename(path))
  }

  years_missing <- setdiff(panel_years,
sort(unique(df$ano)))

  if (length(years_missing) > 0) {
    warning(
      "Anos ausentes em ",
      basename(path),
      ": ",
      paste(years_missing, collapse = ",
")
    )
  }

  expected_rows <- 62 *
length(panel_years)

  if (nrow(df) != expected_rows) {
    warning(
      "Número de linhas diferente do
esperado em ",
      basename(path),
      ": ",
      nrow(df),
      " linhas; esperado = ",
      expected_rows
    )
  }

  log_line(
    basename(path),
    "| linhas =",
    nrow(df),
    "| colunas =",
    ncol(df)
  )

  return(df)
}

#
=====
# 3. LER PAINÉIS INDIVIDUAIS
#
=====

panels_list <- lapply(
  vars_all,

```

```

function(v)
read_and_check_panel(panel_files[[v]],
v)
)

names(panels_list) <- vars_all

#
=====
# 4. INTEGRAR PAINÉIS CLIMÁTICOS
#
=====

climate_full <- Reduce(
  function(x, y) {
    dplyr::full_join(x, y, by =
c("ibge7", "ano"))
  },
  panels_list
) |>
  dplyr::arrange(ano, ibge7)

log_line("Painel climático consolidado
criado.")
log_line("Linhas          =",
nrow(climate_full))
log_line("Colunas        =",
ncol(climate_full))

#
=====
# 5. VALIDAÇÕES DO PAINEL CONSOLIDADO
#
=====

expected_rows <- 62 *
length(panel_years)

if (nrow(climate_full) != expected_rows)
{
  warning(
    "Painel climático consolidado possui
número de linhas diferente do esperado:
",
    nrow(climate_full),
    " linhas; esperado = ",
    expected_rows
  )
}

dup_full <- climate_full |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

if (nrow(dup_full) > 0) {
  readr::write_csv(
    dup_full,
    file.path(climate_logs_dir,
"climate_full_duplicates.csv")
  )
  stop("Duplicidades encontradas no
painel climático consolidado.")
}

count_by_year <- climate_full |>
  dplyr::count(ano) |>
  dplyr::arrange(ano)

log_line("Contagem por ano:")
capture.output(count_by_year) |>
  paste(collapse = "\n") |>
  log_line()

na_counts <-
colSums(is.na(climate_full))

log_line("NAs por variável:")
log_line(paste(names(na_counts),
na_counts, sep = "=", collapse = " | "))

#
=====
# 6. EXPORTAR PAINEL CLIMÁTICO
CONSOLIDADO
#
=====

out_path <- file.path(
  final_output_dir,
  "climate_panel_full_2014_2024.csv"
)

readr::write_csv(climate_full,
out_path)

log_line("Painel climático consolidado
exportado:", out_path)

#
=====
# 7. FINALIZAÇÃO
#
=====

log_line("Script 03 concluído.")
log_line("Log salvo em:", log_file)

cat("\n=====
=====
\n")
cat("SCRIPT 03 CONCLUÍDO\n")
cat("Arquivo gerado:\n")
cat(out_path, "\n")
cat("=====
=====
\n")

```

APÊNDICE E – Script de construção do painel fiscal municipal (04_econ_build_fiscal_panel.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
# SCRIPT: 04_econ_build_fiscal_panel.R
#
# FUNÇÃO:
#   Construir o painel fiscal municipal
#   anual a partir dos
#   arquivos do SICONFI.
#
# OBJETIVO:
#   - Ler bases fiscais de receita e
#   despesa;
#   - Padronizar nomes de colunas e
#   chaves municipais;
#   - Construir Receita Corrente Líquida,
#   FPM líquido e Despesa Corrente;
#   - Montar painel municipal anual
#   balanceado;
#   - Exportar o painel fiscal para uso
#   no Script 05.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (lexists("root_dir")) {
  if (requireNamespace("rstudioapi",
quietly = TRUE)) {
    root_dir <-
rstudioapi::selectDirectory(
  caption = "Selecione a pasta raiz
do projeto"
)
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
"scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(econ_logs_dir, recursive =
TRUE, showWarnings = FALSE)

log_file <- file.path(econ_logs_dir,
"fiscal_build_log.txt")

if (file.exists(log_file))
file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "]" ,
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início do Script
04_econ_build_fiscal_panel.R")
log_line("root_dir =", root_dir)
log_line("UF =", econ_uf)
log_line("Período =", min(econ_years),
"-", max(econ_years))

#
=====
=====
# 1. FUNÇÕES AUXILIARES
#
=====
=====

normalize_ibge7 <- function(x) {
  x <- as.character(x)
  x <- gsub("\\D", "", x)
  stringr::str_pad(x, width = 7, side =
"left", pad = "0")
}

detect_delim_rsiconfi <- function(path,
n = 8) {
  lines <- readLines(path, n = n, warn =
FALSE, encoding = "UTF-8")
  lines <- lines[nzchar(lines)]

  if (!length(lines)) return(";")
}
```

```

    semis                                     <-
sum(stringr::str_count(lines,
stringr::fixed(";")))
    comms                                     <-
sum(stringr::str_count(lines,
stringr::fixed(",")))

    if (comms > semis) "," else ";"
  }

to_numeric_rsiconfi <- function(x) {
  if (is.numeric(x))
    return(as.numeric(x))

  s <- as.character(x)
  s <- gsub("\\\"", "", s)
  s <- gsub("\\\\s", "", s)
  s[s %in% c("", "-", "NA", "N/A")] <-
NA_character_

  s <- ifelse(
    !is.na(s) & grepl("^\\((.*\\))$", s),
    paste0("-", gsub("\\(", "()", s)),
    s
  )

  v1 <- suppressWarnings(
    readr::parse_number(
      s,
      locale =
readr::locale(decimal_mark = ",")
    )
  )

  v2 <- suppressWarnings(as.numeric(s))

  bad1 <- mean(is.na(v1))
  bad2 <- mean(is.na(v2))

  if (!is.finite(bad1)) bad1 <- 1
  if (!is.finite(bad2)) bad2 <- 1

  if (bad1 <= bad2) as.numeric(v1) else
as.numeric(v2)
}

read_rsiconfi_csv <- function(path) {
  if (!file.exists(path)) {
    stop("Arquivo não encontrado: ",
path)
  }

  delim <- detect_delim_rsiconfi(path)

  readr::read_delim(
    path,
    delim = delim,
    show_col_types = FALSE,
    guess_max = 200000,
    progress = FALSE
  ) |>
  janitor::clean_names()
}

pick_col <- function(df, candidates) {
  hit <- candidates[candidates %in%
names(df)]

  if (!length(hit))
    return(NA_character_)

  hit[1]
}

extract_codigo <- function(x) {
  x <- as.character(x)
  x <- gsub("\\\"", "", x)
  x <- sub("\\s*-*$", "", x)
  trimws(x)
}

is_leaf_codes <- function(codes) {
  codes <- as.character(codes)

  has_child <- vapply(
    seq_along(codes),
    function(i) {
      any(
        startsWith(codes,
paste0(codes[i], ".")) &
        codes != codes[i]
      )
    },
    FUN.VALUE = logical(1)
  )

  !has_child
}

build_full_grid <- function(all_ibge7,
years) {
  tidyr::expand_grid(
    ano = as.integer(years),
    ibge7 = as.character(all_ibge7)
  )
}

#
=====
# 2. DEFINIR CAMINHOS DOS ARQUIVOS
BRUTOS
#
=====

rec_path <- file.path(
  econ_receita_dir,
  econ_files_raw$receita_orc
)

desp_path <- file.path(
  econ_desp_orc_dir,
  econ_files_raw$desp_orc
)

if (!file.exists(rec_path)) {
  stop("Arquivo de receita não
encontrado: ", rec_path)
}

```

```

if (!file.exists(desp_path)) {
  stop("Arquivo de despesa não
  encontrado: ", desp_path)
}

```

```

log_line("Arquivo receita =", rec_path)
log_line("Arquivo despesa =", desp_path)

```

```

#
=====
# 3. LER E PADRONIZAR RECEITA
ORÇAMENTÁRIA
#
=====

```

```

rec_raw <- read_rsiconfi_csv(rec_path)

```

```

col_an      <- pick_col(rec_raw,
c("an_exercicio"))
col_id      <- pick_col(rec_raw,
c("id_ente", "cd_ente", "codigo_ente"))
col_no      <- pick_col(rec_raw,
c("no_ente", "nome_ente"))
col_sg      <- pick_col(rec_raw,
c("sg_ente", "uf"))
col_val     <- pick_col(rec_raw, c("value",
"valor", "vl_valor"))
col_el      <- pick_col(rec_raw,
c("elementlabel", "element_label"))
col_tip     <- pick_col(rec_raw,
c("tipo_receita", "tipo_de_receita",
"tp_receita"))
col_rub     <- pick_col(rec_raw,
c("rubrica_receita", "rubrica",
"rubrica_da_receita"))
col_pop     <- pick_col(rec_raw,
c("qt_habitante", "populacao", "pop"))

```

```

needed_rec <- c(
  col_an, col_id, col_no, col_sg,
  col_val, col_el, col_tip, col_rub
)

```

```

if (any(is.na(needed_rec))) {
  stop("Não foi possível identificar
  todas as colunas necessárias da
  receita.")
}

```

```

rec_orc <- rec_raw |>
  dplyr::rename(
    an_exercicio =
dplyr::all_of(col_an),
    id_ente      =
dplyr::all_of(col_id),
    no_ente      =
dplyr::all_of(col_no),
    sg_ente      =
dplyr::all_of(col_sg),
    value        =
dplyr::all_of(col_val),

```

```

    elementlabel =
dplyr::all_of(col_el),
    tipo_receita =
dplyr::all_of(col_tip),
    rubrica_receita =
dplyr::all_of(col_rub)
  ) |>
  dplyr::mutate(
    ano = as.integer(an_exercicio),
    ibge7 = normalize_ibge7(id_ente),
    nm_mun = as.character(no_ente),
    value = to_numeric_rsiconfi(value),
    codigo =
extract_codigo(elementlabel),
    POP = if (!is.na(col_pop))
to_numeric_rsiconfi(.data[[col_pop]])
else NA_real_
  ) |>
  dplyr::filter(
    sg_ente == econ_uf,
    ano %in% econ_years
  ) |>
  dplyr::select(
    ano,
    ibge7,
    nm_mun,
    tipo_receita,
    rubrica_receita,
    elementlabel,
    codigo,
    value,
    POP
  )

```

```

log_line("Receita lida: linhas =",
nrow(rec_orc))

```

```

#
=====
# 4. EXTRAIR POPULAÇÃO INFORMADA NA BASE
FISCAL
#
=====

```

```

pop_panel <- rec_orc |>
  dplyr::select(ano, ibge7, POP) |>
  dplyr::filter(!is.na(POP)) |>
  dplyr::distinct(ano, ibge7, .keep_all
= TRUE)

```

```

log_line("POP fiscal: linhas =",
nrow(pop_panel))

```

```

#
=====
# 5. CONSTRUIR RECEITA CORRENTE LÍQUIDA
#
=====

```

```

rc_candidates <- rec_orc |>

```



```

#
=====
# 7. LER E PADRONIZAR DESPESA
ORÇAMENTÁRIA
#
=====

desp_raw <- read_rsiconfi_csv(desp_path)

needed_desp <- c(
  "an_exercicio",
  "id_ente",
  "no_ente",
  "sg_ente",
  "fase_despesa",
  "rubrica_despesa",
  "value",
  "elementlabel"
)

missing_desp <- setdiff(needed_desp,
names(desp_raw))

if (length(missing_desp) > 0) {
  stop(
    "Faltam colunas na base de despesa: ",
    paste(missing_desp, collapse = ", ")
  )
}

desp_orc <- desp_raw |>
  dplyr::mutate(
    ano = as.integer(an_exercicio),
    ibge7 = normalize_ibge7(id_ente),
    nm_mun = as.character(no_ente),
    value = to_numeric_rsiconfi(value),
    codigo = extract_codigo(elementlabel)
  ) |>
  dplyr::filter(
    sg_ente == econ_uf,
    ano %in% econ_years
  ) |>
  dplyr::select(
    ano,
    ibge7,
    nm_mun,
    fase_despesa,
    rubrica_despesa,
    elementlabel,
    codigo,
    value
  )

log_line("Despesa lida: linhas =",
nrow(desp_orc))

```

```

#
=====
# 8. CONSTRUIR DESPESA CORRENTE
#
=====

dc_panel <- desp_orc |>
  dplyr::filter(
    fase_despesa == "Despesas
Liquidadas",
    stringr::str_detect(
      elementlabel,
      stringr::regex("^3\\.0\\.\"",
ignore_case = TRUE)
    ),
    stringr::str_detect(
      elementlabel,
      stringr::regex("Despesas\\s+Correntes$"
, ignore_case = TRUE)
    )
  ) |>
  dplyr::group_by(ano, ibge7) |>
  dplyr::summarise(
    dc = sum(value, na.rm = TRUE),
    .groups = "drop"
  )

log_line("DC construída: linhas =",
nrow(dc_panel))

#
=====
# 9. CONSTRUIR GRADE MUNICIPAL ANUAL
#
=====

ibge_col <- {
  cand <- c(
    "ibge7",
    "cd_mun",
    "cd_mun7",
    "code_muni",
    "geocodigo",
    "cd_geocmu"
  )

  cand[cand %in% names(am_muni)][1]
}

name_col <- {
  cand <- c(
    "name_muni",
    "nm_mun",
    "nome_municipio",
    "municipio",
    "nm_municipio"
  )

  cand[cand %in% names(am_muni)][1]
}

```

```

if (is.na(ibge_col) || length(ibge_col)
== 0) {
  stop("Não encontrei coluna IBGE no
shapefile.")
}

```

```

mun_base <- am_muni |>
  sf::st_drop_geometry() |>
  dplyr::transmute(
    ibge7
  normalize_ibge7(.data[[ibge_col]]), =
    nm_mun_shp = if (!is.na(name_col)) {
      as.character(.data[[name_col]])
    } else {
      NA_character_
    }
  ) |>
  dplyr::distinct()

```

```

all_ibge7 <- sort(unique(mun_base$ibge7))

```

```

grid <- build_full_grid(all_ibge7,
econ_years) |>
  dplyr::left_join(mun_base, by =
"ibge7")

```

```

#
=====
# 10. INTEGRAR PAINEL FISCAL
#
=====

```

```

econ_panel <- grid |>
  dplyr::left_join(rc_liq, by = c("ano",
"ibge7")) |>
  dplyr::left_join(fpm_liq, by =
c("ano", "ibge7")) |>
  dplyr::left_join(dc_panel, by =
c("ano", "ibge7")) |>
  dplyr::left_join(pop_panel, by =
c("ano", "ibge7")) |>
  dplyr::mutate(
    nm_mun = dplyr::coalesce(nm_mun,
nm_mun_shp)
  ) |>
  dplyr::select(
    ano,
    ibge7,
    nm_mun,
    rc_liq,
    fpm_liq,
    dc,
    POP
  ) |>
  dplyr::arrange(ano, ibge7)

```

```

#
=====
# 11. VALIDAR PAINEL FISCAL

```

```

#
=====

```

```

expected_rows <- 62 * length(econ_years)

```

```

log_line("Painel fiscal: linhas =",
nrow(econ_panel), "| esperado =",
expected_rows)

```

```

if (nrow(econ_panel) != expected_rows) {
  warning(
    "Painel fiscal possui número de
linhas diferente do esperado: ",
    nrow(econ_panel)
  )
}

```

```

dup_panel <- econ_panel |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

```

```

if (nrow(dup_panel) > 0) {
  readr::write_csv(
    dup_panel,
    file.path(econ_logs_dir,
"fiscal_panel_duplicates.csv")
  )
  stop("Duplicidades encontradas no
painel fiscal.")
}

```

```

count_by_year <- econ_panel |>
  dplyr::count(ano) |>
  dplyr::arrange(ano)

```

```

log_line("Contagem por ano:")
capture.output(count_by_year) |>
  paste(collapse = "\n") |>
  log_line()

```

```

na_counts <- colSums(is.na(econ_panel))

```

```

log_line("NAs por coluna:")
log_line(paste(names(na_counts),
na_counts, sep = "=", collapse = " | "))

```

```

#
=====
# 12. EXPORTAR PAINEL FISCAL
#
=====

```

```

out_csv <- file.path(
  econ_panel_dir,
  "fiscal_panel_2014_2024.csv"
)

```

```

out_xlsx <- file.path(
  econ_panel_dir,
  "fiscal_panel_2014_2024.xlsx"
)

```

```

readr::write_csv(econ_panel, out_csv)

wb <- openxlsx::createWorkbook()

openxlsx::addWorksheet(wb,
"fiscal_panel")
openxlsx::writeDataTable(wb,
"fiscal_panel", econ_panel)

meta <- tibble::tibble(
  item = c(
    "gerado_em",
    "uf",
    "periodo",
    "arquivo_receita",
    "arquivo_despesa",
    "observacao"
  ),
  valor = c(
    format(Sys.time(),          "%Y-%m-%d
%H:%M:%S"),
    econ_uf,
    paste(range(econ_years), collapse =
"-"),
    basename(rec_path),
    basename(desp_path),
    "Painel fiscal municipal construído
a partir de bases do SICONFI."
  )
)

openxlsx::addWorksheet(wb, "metadados")
openxlsx::writeDataTable(wb,
"metadados", meta)

openxlsx::saveWorkbook(wb,    out_xlsx,
overwrite = TRUE)

log_line("CSV exportado:", out_csv)
log_line("XLSX exportado:", out_xlsx)

#
=====
=====
# 13. FINALIZAÇÃO
#
=====
=====

log_line("Script 04 concluído.")
log_line("Log salvo em:", log_file)

cat("\n=====
=====\\n")
cat("SCRIPT 04 CONCLUÍDO\\n")
cat("Arquivo gerado:\\n")
cat(out_csv, "\\n")
cat("=====
=====\\n")

```

APÊNDICE F – Script de integração econômica, deflacionamento e cálculo de variáveis per capita (05_econ_add_pib_pop_deflate_and_finalize.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
#
# SCRIPT:
05_econ_add_pib_pop_deflate_and_finaliz
e.R
#
# FUNÇÃO:
# Complementar o painel fiscal com PIB,
# população, IPCA,
# deflacionamento e variáveis per
# capita.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
quietly = TRUE)) {
    root_dir <-
rstudioapi::selectDirectory(
  caption = "Selecione a pasta raiz
do projeto"
)
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
"scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

ANO_BASE <- 2024

#
=====
=====
# 0.1. LOG

#
=====
=====
dir.create(econ_logs_dir, recursive =
TRUE, showWarnings = FALSE)

log_file <- file.path(econ_logs_dir,
"econ_final_build_log.txt")

if (file.exists(log_file))
file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "] ",
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início do Script
05_econ_add_pib_pop_deflate_and_finaliz
e.R")
log_line("root_dir =", root_dir)
log_line("Ano-base do deflator =",
ANO_BASE)
log_line("Período =", min(econ_years),
"-", max(econ_years))

#
=====
=====
# 1. LER PAINEL FISCAL
#
=====
=====

fiscal_path <-
file.path(econ_panel_dir,
"fiscal_panel_2014_2024.csv")

if (!file.exists(fiscal_path)) {
  stop("Painel fiscal não encontrado: ",
fiscal_path)
}

econ_panel <-
readr::read_csv(fiscal_path,
show_col_types = FALSE) |>
janitor::clean_names() |>
dplyr::mutate(
  ano = as.integer(ano),
  ibge7 =
stringr::str_pad(as.character(ibge7),
7, "left", "0")
)
```

```

)

needed_cols <- c("ano", "ibge7",
"nm_mun", "rc_liq", "fpm_liq", "dc",
"pop")
missing_cols <- setdiff(needed_cols,
names(econ_panel))

if (length(missing_cols) > 0) {
  stop(
    "O painel fiscal não possui as
colunas esperadas: ",
    paste(missing_cols, collapse = ", ")
  )
}

log_line("Painel fiscal lido.")
log_line("Linhas =", nrow(econ_panel))
log_line("Colunas =", ncol(econ_panel))

#
=====
# 2. LISTA DOS MUNICÍPIOS DO SHAPEFILE
#
=====

if (is.null(am_muni)) {
  stop("Objeto am_muni não disponível.
Verifique o shapefile no 00_config.R.")
}

ibge_col <- {
  cand <- c("ibge7", "cd_mun",
"cd_mun7", "code_muni", "geocodigo",
"cd_geocmu")
  cand[cand %in% names(am_muni)][1]
}

if (is.na(ibge_col) || length(ibge_col)
== 0) {
  stop("Não encontrei coluna IBGE no
shapefile.")
}

am_codes <- am_muni |>
  sf::st_drop_geometry() |>
  dplyr::transmute(
    ibge7 =
stringr::str_pad(as.character(.data[[ib
ge_col]]), 7, "left", "0")
  ) |>
  dplyr::distinct() |>
  dplyr::pull(ibge7) |>
  sort()

log_line("Municípios identificados no
shapefile =", length(am_codes))

#
=====
# 3. FUNÇÕES AUXILIARES SIDRA

```

```

#
=====

to_num <- function(x) {
  if (is.numeric(x))
return(as.numeric(x))
  readr::parse_number(
    as.character(x),
    locale = readr::locale(decimal_mark
= ",")
  )
}

sidra_standardize <- function(df,
value_name) {
  df <- janitor::clean_names(df)

  mun_col <- names(df)[
    stringr::str_detect(names(df),
"municip" ) &
    stringr::str_detect(names(df),
"codigo")
  ][1]

  ano_col <- dplyr::coalesce(
    names(df)[names(df) == "ano"][1],

names(df)[stringr::str_detect(names(df)
, "periodo")][1]
  )

  val_col <- names(df)[names(df) %in%
c("valor", "value")][1]

  if (is.na(mun_col) || is.na(ano_col)
|| is.na(val_col)) {
    stop(
      "SIDRA: não foi possível detectar
município, ano ou valor. Colunas: ",
      paste(names(df), collapse = ", ")
    )
  }

  df |>
    dplyr::transmute(
      ibge7 =
stringr::str_pad(as.character(.data[[mu
n_col]]), 7, "left", "0"),
      ano =
as.integer(as.character(.data[[ano_col]
]])),
      !!value_name :=
to_num(.data[[val_col]])
    )
}

sidra_fetch_n6_chunks <-
function(table_id, var_id, years,
n6_codes,

value_name, chunk_size = 50) {

  n6_codes <-
unique(as.character(n6_codes))

```

```

    chunks      <-      split(n6_codes,
ceiling(seq_along(n6_codes) /
chunk_size))

    out_list     <-      vector("list",
length(chunks))

    for (i in seq_along(chunks)) {

        codes_i   <-      paste(chunks[[i]],
collapse = ",")

        api_i <- paste0(
            "/t/", table_id,
            "/n6/", codes_i,
            "/v/", var_id,
            "/p/", min(years), "-", max(years)
        )

        log_line("SIDRA chunk", i, "/",
length(chunks), "| api =", api_i)

        df_i <- sidrar::get_sidra(api =
api_i)

        out_list[[i]] <-
sidra_standardize(df_i, value_name =
value_name)
    }

    dplyr::bind_rows(out_list) |>
    dplyr::filter(
        ano %in% years,
        ibge7 %in% n6_codes
    )
}

#
=====
# 4. BAIXAR IPCA E CALCULAR DEFLATOR
#
=====

fetch_ipca_dec_by_year <-
function(years, base_year) {

    p_ini <- paste0(min(years), "01")
    p_fim <- paste0(max(years), "12")

    api_ipca <- paste0(
        "/t/1737/n1/all/v/2266/p/",
        p_ini,
        "-",
        p_fim
    )

    log_line("Baixando IPCA via SIDRA:",
api_ipca)

    ipca_raw <- sidrar::get_sidra(api =
api_ipca) |>
    janitor::clean_names()

    per_col <- dplyr::coalesce(
        names(ipca_raw)[names(ipca_raw)
%in%
c("mes_codigo",
"periodo_codigo")][1],

names(ipca_raw)[stringr::str_detect(nam
es(ipca_raw),
"mes_.*codigo|periodo_.*codigo")][1]
)

    val_col <-
names(ipca_raw)[names(ipca_raw) %in%
c("valor", "value")][1]

    if (is.na(per_col) || is.na(val_col))
    {
        stop("IPCA: não foi possível
identificar colunas de período ou
valor.")
    }

    ipca_m <- ipca_raw |>
    dplyr::transmute(
        periodo =
as.integer(as.character(.data[[per_col]
])),
        ipca_index =
to_num(.data[[val_col]])
    ) |>
    dplyr::mutate(
        ano = periodo %/% 100L,
        mes = periodo %% 100L
    ) |>
    dplyr::filter(
        ano %in% years,
        mes == 12L
    ) |>
    dplyr::select(ano, ipca_dec =
ipca_index) |>
    dplyr::distinct()

    base_ipca <- ipca_m |>
    dplyr::filter(ano == base_year) |>
    dplyr::pull(ipca_dec)

    if (length(base_ipca) != 1 ||
is.na(base_ipca)) {
        stop("IPCA do ano-base não
encontrado.")
    }

    ipca_m |>
    dplyr::mutate(
        ano_base = base_year,
        ipca_dec_base = base_ipca,
        deflator = ipca_dec_base /
ipca_dec
    )
}

ipca_defl <-
fetch_ipca_dec_by_year(econ_years,
ANO_BASE)

log_line("IPCA/deflator carregado.")

```

```

log_line("Anos =",
paste(range(ipca_defl$ano), collapse =
"-"))

#
=====
=====
# 5. BAIXAR PIB MUNICIPAL
#
=====
=====

log_line("Baixando PIB municipal via
SIDRA.")

pib_df <- sidra_fetch_n6_chunks(
  table_id = sidra_tables$pib_table,
  var_id   = sidra_tables$pib_var,
  years    = econ_years,
  n6_codes = am_codes,
  value_name = "PIB",
  chunk_size = 50
) |>
  dplyr::mutate(
    PIB = PIB * 1000
  )

log_line("PIB baixado: linhas =",
nrow(pib_df))

#
=====
=====
# 6. BAIXAR POPULAÇÃO RESIDENTE
#
=====
=====

pop_var_id <- 9324

log_line("Baixando população municipal
via SIDRA.")

pop_sidra <- sidra_fetch_n6_chunks(
  table_id = sidra_tables$pop_table,
  var_id   = pop_var_id,
  years    = econ_years,
  n6_codes = am_codes,
  value_name = "POP_SIDRA",
  chunk_size = 50
)

log_line("POP SIDRA baixada: linhas =",
nrow(pop_sidra))

#
=====
=====
# 7. INTEGRAR, DEFLACIONAR E CALCULAR
PER CAPITA
#
=====
=====

```

```

econ_final <- econ_panel |>
  dplyr::left_join(pop_sidra, by =
c("ano", "ibge7")) |>
  dplyr::left_join(pib_df, by = c("ano",
"ibge7")) |>
  dplyr::left_join(
    ipca_defl |>
      dplyr::select(ano, ipca_dec,
ipca_dec_base, deflator),
    by = "ano"
  ) |>
  dplyr::mutate(
    pop = dplyr::coalesce(pop,
POP_SIDRA),

    rc_pc = dplyr::if_else(!is.na(pop)
& pop > 0, rc_liq / pop, NA_real_),
    fpm_pc = dplyr::if_else(!is.na(pop)
& pop > 0, fpm_liq / pop, NA_real_),
    dc_pc = dplyr::if_else(!is.na(pop)
& pop > 0, dc / pop, NA_real_),
    pib_pc = dplyr::if_else(!is.na(pop)
& pop > 0, PIB / pop, NA_real_),

    rc_liq_real = rc_liq * deflator,
    fpm_liq_real = fpm_liq * deflator,
    dc_real = dc * deflator,
    pib_real = PIB * deflator,

    rc_pc_real =
dplyr::if_else(!is.na(pop) & pop > 0,
rc_liq_real / pop, NA_real_),
    fpm_pc_real =
dplyr::if_else(!is.na(pop) & pop > 0,
fpm_liq_real / pop, NA_real_),
    dc_pc_real =
dplyr::if_else(!is.na(pop) & pop > 0,
dc_real / pop, NA_real_),
    pib_pc_real =
dplyr::if_else(!is.na(pop) & pop > 0,
pib_real / pop, NA_real_)
  ) |>
  dplyr::arrange(ano, ibge7)

log_line("Painel econômico final
construído.")
log_line("Linhas =", nrow(econ_final))
log_line("Colunas =", ncol(econ_final))

#
=====
=====
# 8. VALIDAÇÕES
#
=====
=====

expected_rows <- 62 * length(econ_years)

if (nrow(econ_final) != expected_rows) {
  warning("Número de linhas diferente do
esperado no painel econômico final.")
}

```

```

dup <- econ_final |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

if (nrow(dup) > 0) {
  readr::write_csv(
    dup,
    file.path(econ_logs_dir,
      "econ_final_duplicates.csv")
  )
  stop("Duplicidades encontradas no
painel econômico final.")
}

na_counts <- colSums(is.na(econ_final))

log_line("NAs por coluna:")
log_line(paste(names(na_counts),
na_counts, sep = "=", collapse = " | "))

#
=====
# 9. EXPORTAR
#
=====

out_csv <- file.path(econ_final_dir,
"econ_panel_final_2014_2024.csv")
out_xlsx <- file.path(econ_final_dir,
"econ_panel_final_2014_2024.xlsx")

readr::write_csv(econ_final, out_csv)

wb <- openxlsx::createWorkbook()

openxlsx::addWorksheet(wb,
"econ_panel_final")
openxlsx::writeDataTable(wb,
"econ_panel_final", econ_final)

meta <- tibble::tibble(
  item = c(
    "gerado_em",
    "periodo",
    "ano_base_deflator",
    "fonte_pib",
    "fonte_pop",
    "fonte_ipca",
    "observacao"
  ),
  valor = c(
    format(Sys.time(), "%Y-%m-%d
%H:%M:%S"),
    paste(range(econ_years), collapse =
"-"),
    as.character(ANO_BASE),
    paste0("SIDRA tabela ",
sidra_tables$pib_table, ", variável ",
sidra_tables$pib_var),
    paste0("SIDRA tabela ",
sidra_tables$pop_table, ", variável ",
pop_var_id),
    "SIDRA tabela 1737, variável 2266,
IPCA dezembro",
    "Painel econômico final com PIB,
população, deflator, variáveis reais e
per capita."
  )
)

openxlsx::addWorksheet(wb, "metadados")
openxlsx::writeDataTable(wb,
"metadados", meta)

openxlsx::saveWorkbook(wb, out_xlsx,
overwrite = TRUE)

log_line("CSV exportado:", out_csv)
log_line("XLSX exportado:", out_xlsx)

#
=====
# 10. FINALIZAÇÃO
#
=====

log_line("Script 05 concluído.")
log_line("Log salvo em:", log_file)

cat("\n=====
=====\\n")
cat("SCRIPT 05 CONCLUÍDO\\n")
cat("Arquivo gerado:\\n")
cat(out_csv, "\\n")
cat("=====
=====\\n")

```

APÊNDICE G – Script de integração final das bases (06_build_panel_master_2014_2024.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
# de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
#                                SCRIPT:
06_build_panel_master_2014_2024.R
#
# FUNÇÃO:
#   Integrar o painel climático
#   consolidado e o painel econômico
#   final, gerando a base master do
#   projeto.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
    quietly = TRUE)) {
    root_dir <-
rstudioapi::selectDirectory(
  caption = "Selecione a pasta raiz
do projeto"
)
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
"scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(complete_logs_dir, recursive
= TRUE, showWarnings = FALSE)
dir.create(complete_outputs_dir,
recursive = TRUE, showWarnings = FALSE)

log_file <-
file.path(complete_logs_dir,
"panel_master_build_log.txt")

if (file.exists(log_file))
file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "]",
    paste(..., collapse = " ")
  )
  cat(msg, "\n")
  cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início do Script
06_build_panel_master_2014_2024.R")
log_line("root_dir =", root_dir)
log_line("Período =", min(panel_years),
"-", max(panel_years))

#
=====
=====
# 1. DEFINIR ARQUIVOS DE ENTRADA
#
=====
=====

climate_path <- file.path(
  final_output_dir,
  "climate_panel_full_2014_2024.csv"
)

econ_path <- file.path(
  econ_final_dir,
  "econ_panel_final_2014_2024.csv"
)

if (!file.exists(climate_path)) {
  stop("Painel climático não encontrado:
", climate_path)
}

if (!file.exists(econ_path)) {
  stop("Painel econômico não encontrado:
", econ_path)
}

log_line("Arquivo climático =",
climate_path)
```

```

log_line("Arquivo      econômico      =",
econ_path)

#
=====
# 2. LER E PADRONIZAR BASES
#
=====

pad_ibge7 <- function(x) {
  stringr::str_pad(as.character(x), 7,
"left", "0")
}

climate <- readr::read_csv(climate_path,
show_col_types = FALSE) |>
janitor::clean_names() |>
dplyr::mutate(
  ibge7 = pad_ibge7(ibge7),
  ano = as.integer(ano)
)

econ <- readr::read_csv(econ_path,
show_col_types = FALSE) |>
janitor::clean_names() |>
dplyr::mutate(
  ibge7 = pad_ibge7(ibge7),
  ano = as.integer(ano)
)

log_line("Clima:      linhas      =",
nrow(climate),      "|      colunas      =",
ncol(climate))
log_line("Econ: linhas =", nrow(econ),
"| colunas =", ncol(econ))

#
=====
# 3. FUNÇÕES DE VALIDAÇÃO
#
=====

check_panel_shape <- function(df,
df_name, years_expected) {

  required_cols <- c("ibge7", "ano")
  missing_cols <-
setdiff(required_cols, names(df))

  if (length(missing_cols) > 0) {
    stop(df_name, " não possui colunas
obrigatórias: ", paste(missing_cols,
collapse = ", "))
  }

  dup <- df |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

  if (nrow(dup) > 0) {
    stop(df_name, " possui duplicidades
em ibge7 + ano.")
  }

  expected_rows <- 62 *
length(years_expected)

  if (nrow(df) != expected_rows) {
    stop(
      df_name,
      " possui ",
      nrow(df),
      " linhas; esperado = ",
      expected_rows
    )
  }

  missing_years <-
setdiff(years_expected,
sort(unique(df$ano)))

  if (length(missing_years) > 0) {
    stop(df_name, " está sem os anos: ",
paste(missing_years, collapse = ", "))
  }

  count_by_year <- df |>
  dplyr::count(ano) |>
  dplyr::filter(n != 62)

  if (nrow(count_by_year) > 0) {
    stop(df_name, " não possui 62
municípios em todos os anos.")
  }

  TRUE
}

check_panel_shape(climate,
"climate_panel_full", panel_years)
check_panel_shape(econ,
"econ_panel_final", panel_years)

log_line("Validação      estrutural
concluída em clima e economia.")

#
=====
# 4. VERIFICAR COMPATIBILIDADE DAS
CHAVES
#
=====

keys_climate <- climate |>
dplyr::distinct(ibge7, ano)

keys_econ <- econ |>
dplyr::distinct(ibge7, ano)

missing_in_econ <- dplyr::anti_join(
  keys_climate,
  keys_econ,

```

```

    by = c("ibge7", "ano")
  )

missing_in_climate <- dplyr::anti_join(
  keys_econ,
  keys_climate,
  by = c("ibge7", "ano")
)

if (nrow(missing_in_econ) > 0 ||
    nrow(missing_in_climate) > 0) {

  if (nrow(missing_in_econ) > 0) {
    readr::write_csv(
      missing_in_econ,
      file.path(complete_outputs_dir,
        "missing_keys_in_econ.csv")
    )
  }

  if (nrow(missing_in_climate) > 0) {
    readr::write_csv(
      missing_in_climate,
      file.path(complete_outputs_dir,
        "missing_keys_in_climate.csv")
    )
  }

  stop("As chaves entre clima e economia
  não são compatíveis.")
}

log_line("Cobertura de chaves 1:1
confirmada.")

#
=====
# 5. INTEGRAR BASE MASTER
#
=====

panel_master <- econ |>
  dplyr::inner_join(
    climate,
    by = c("ibge7", "ano")
  ) |>
  dplyr::arrange(ano, ibge7)

expected_rows <- 62 *
length(panel_years)

if (nrow(panel_master) != expected_rows) {
  stop("panel_master possui número de
  linhas diferente do esperado.")
}

if
(any(duplicated(names(panel_master))))
{
  stop("Há nomes de colunas duplicados
  após o merge.")
}

```

```

log_line("Panel master criado.")
log_line("Linhas" =,
nrow(panel_master))
log_line("Colunas" =,
ncol(panel_master))

#
=====
# 6. GERAR DICIONÁRIO DE VARIÁVEIS
#
=====

infer_origem <- function(v) {

  if (v %in% c("ano", "ibge7",
    "nm_mun")) return("identificacao")

  if (stringr::str_detect(v,
    "ipca|deflator"))
    return("economica_precos")
  if (stringr::str_detect(v, "pib"))
    return("economica_produto")
  if (stringr::str_detect(v,
    "rc|dc|fpm"))
    return("economica_fiscal")
  if (v %in% c("pop", "pop_sidra"))
    return("economica_demografia")

  if (stringr::str_detect(v,
    "_anual$")) return("climatica_anual")

  return("derivada")
}

infer_fonte <- function(origem) {
  dplyr::case_when(
    origem == "identificacao" ~ "IBGE /
    integração municipal",
    origem == "economica_precos" ~
    "SIDRA/IBGE - IPCA",
    origem == "economica_produto" ~
    "SIDRA/IBGE - PIB Municipal",
    origem == "economica_fiscal" ~
    "SICONFI/STN",
    origem == "economica_demografia" ~
    "SICONFI/STN e SIDRA/IBGE",
    origem == "climatica_anual" ~
    "TerraClimate",
    TRUE ~ "Derivação própria"
  )
}

infer_unidade <- function(v) {

  if (v == "ano") return("ano")
  if (v == "ibge7") return("código
  IBGE")
  if (v == "nm_mun") return("nome do
  município")

  if (stringr::str_detect(v, "ipca"))
    return("número-índice")
}

```

```

if (v == "deflator")
return("adimensional")

if (stringr::str_detect(v,
"_pc_real")) return("R$/hab, preços
reais")
if (stringr::str_detect(v, "_pc$"))
return("R$/hab")
if (stringr::str_detect(v, "_real"))
return("R$ reais")
if (stringr::str_detect(v,
"rc_liq|fpm_liq|dc|pib")) return("R$
correntes")

if (v %in% c("pop", "pop_sidra"))
return("habitantes")

if (stringr::str_detect(v,
"^ppt|^aet|^pet|^def|^soil|^q"))
return("mm")
if (stringr::str_detect(v,
"^tmax|^tmin|^tmean")) return("°C")
if (stringr::str_detect(v, "^ws"))
return("m/s")
if (stringr::str_detect(v, "^pdsi"))
return("índice adimensional")

NA_character_
}

infer_descricao <- function(v) {

if (v == "ibge7") return("Código IBGE
municipal padronizado com sete
dígitos.")
if (v == "ano") return("Ano de
referência da observação.")
if (v == "nm_mun") return("Nome do
município.")

if (v == "rc_liq") return("Receita
Corrente Líquida municipal em valores
correntes.")
if (v == "fpm_liq") return("Fundo de
Participação dos Municípios líquido em
valores correntes.")
if (v == "dc") return("Despesa
Corrente municipal liquidada em valores
correntes.")
if (v == "PIB" || v == "pib")
return("Produto Interno Bruto municipal
em valores correntes.")
if (v == "pop") return("População
residente utilizada no painel.")
if (v == "pop_sidra")
return("População residente obtida pelo
SIDRA/IBGE.")

if (v == "ipca_dec") return("IPCA de
dezembro do ano de referência.")
if (v == "ipca_dec_base") return("IPCA
de dezembro do ano-base.")
if (v == "deflator") return("Fator de
deflacionamento utilizado para converter
valores nominais em reais.")

```

```

if (stringr::str_detect(v, "_real"))
return("Variável monetária convertida
para valores reais pelo deflator.")
if (stringr::str_detect(v, "_pc"))
return("Variável per capita calculada
pela divisão do valor correspondente
pela população.")

base_clima <- dplyr::case_when(
stringr::str_detect(v, "^pdsi") ~
"índice de severidade de seca de
Palmer",
stringr::str_detect(v, "^ppt") ~
"precipitação anual",
stringr::str_detect(v, "^tmax") ~
"temperatura máxima anual",
stringr::str_detect(v, "^tmin") ~
"temperatura mínima anual",
stringr::str_detect(v, "^tmean") ~
"temperatura média anual",
stringr::str_detect(v, "^aet") ~
"evapotranspiração real anual",
stringr::str_detect(v, "^pet") ~
"evapotranspiração potencial anual",
stringr::str_detect(v, "^def") ~
"déficit hídrico anual",
stringr::str_detect(v, "^soil") ~
"umidade do solo anual",
stringr::str_detect(v, "^ws") ~
"velocidade do vento anual",
stringr::str_detect(v, "^q") ~
"escoamento superficial anual",
TRUE ~ NA_character_
)

if (!is.na(base_clima) &&
stringr::str_detect(v, "_anual$")) {
return(paste0(
"Variável climática municipal
referente a ",
base_clima,
", extraída do TerraClimate e
agregada para frequência anual."
))
}

return("Variável integrante do painel
master municipal anual.")
}

num_stats <- function(x) {
if (!is.numeric(x)) {
return(c(min = NA, p50 = NA, p99 =
NA, max = NA))
}

c(
min = suppressWarnings(min(x, na.rm
= TRUE)),
p50 =
suppressWarnings(stats::quantile(x,
0.50, na.rm = TRUE, names = FALSE)),
p99 =
suppressWarnings(stats::quantile(x,
0.99, na.rm = TRUE, names = FALSE)),

```

```

    max = suppressWarnings(max(x, na.rm
= TRUE))
  )
}

```

```

vars <- names(panel_master)
origem <- vapply(vars, infer_origem,
character(1))
fonte <- vapply(origem, infer_fonte,
character(1))
unidade <- vapply(vars, infer_unidade,
character(1))
descricao <- vapply(vars,
infer_descricao, character(1))

```

```

stats_mat <- t(vapply(panel_master,
num_stats, numeric(4)))
stats_tbl <-
tibble::as_tibble(stats_mat)
names(stats_tbl) <- c("min", "p50",
"p99", "max")

```

```

dictionary <- tibble::tibble(
  var = vars,
  origem = origem,
  fonte = fonte,
  unidade = unidade,
  descricao = descricao,
  classe = vapply(panel_master,
function(x) paste(class(x), collapse =
", "), character(1)),
  n_unique = vapply(panel_master,
dplyr::n_distinct, integer(1)),
  na_count = vapply(panel_master,
function(x) sum(is.na(x)), integer(1)),
  na_rate = round(vapply(panel_master,
function(x) mean(is.na(x)), numeric(1))
* 100, 2)
) |>
  dplyr::bind_cols(stats_tbl) |>
  dplyr::arrange(origem, var)

```

```

log_line("Dicionário de variáveis
gerado.")

```

```

#
=====
=====
# 7. EXPORTAR BASE FINAL E DICIONÁRIO
#
=====
=====

```

```

out_csv <-
file.path(complete_outputs_dir,
"panel_master_2014_2024.csv")
out_xlsx <-
file.path(complete_outputs_dir,
"panel_master_2014_2024.xlsx")
dict_csv <-
file.path(complete_outputs_dir,

```

```

"panel_master_dictionary_2014_2024.csv"
)

```

```

readr::write_csv(panel_master, out_csv)
readr::write_csv(dictionary, dict_csv)

```

```

writexl::write_xlsx(
  list(
    panel_master = panel_master,
    dictionary = dictionary
  ),
  out_xlsx
)

```

```

log_line("CSV exportado:", out_csv)
log_line("XLSX exportado:", out_xlsx)
log_line("Dicionário exportado:",
dict_csv)

```

```

#
=====
=====
# 8. FINALIZAÇÃO
#
=====
=====

```

```

log_line("Script 06 concluído.")
log_line("Log salvo em:", log_file)

```

```

cat("\n=====
=====
\n")
cat("SCRIPT 06 CONCLUÍDO\n")
cat("Arquivo gerado:\n")
cat(out_csv, "\n")
cat("=====
=====
\n")

```

APÊNDICE H – Script de auditoria e validação estrutural da base consolidada (07_audit_panel_master_2014_2024.R)

```
#####
#####
# PROJETO: Manual Técnico de Construção
de Bases de Dados Aplicadas
# AUTOR: Felipe Soares Esteves
# PROGRAMA: PPG-ECO/UFAM
#
#                                SCRIPT:
07_audit_panel_master_2014_2024.R
#
# FUNÇÃO:
#   Auditar e validar a base final
consolidada do projeto.
#
# OBJETIVO:
#   - Ler a base
panel_master_2014_2024.csv;
#   - Verificar estrutura do painel
municipal anual;
#   - Checar duplicidades, lacunas e
cobertura temporal;
#   - Identificar valores ausentes;
#   - Realizar verificações básicas em
variáveis econômicas;
#   - Realizar verificações básicas em
variáveis climáticas;
#   - Gerar relatórios de auditoria para
documentação do pipeline.
#
# ENTRADAS:
#
data_processed/complete_outputs/panel_m
aster_2014_2024.csv
#
# SAÍDAS:
#
data_processed/complete_outputs/logs/au
dit_panel_master_2014_2024.txt
#
data_processed/complete_outputs/audit_s
ummary.csv
#
data_processed/complete_outputs/audit_n
a_by_var.csv
#
data_processed/complete_outputs/audit_n
a_by_year.csv
#
data_processed/complete_outputs/audit_k
ey_holes.csv
#
data_processed/complete_outputs/audit_o
utliers_top10.csv
#
# OBSERVAÇÃO:
#   Este script não altera a base final.
Ele apenas executa
#   verificações estruturais e gera
relatórios de auditoria.
#####
#####

#
=====
=====
# 0. CARREGAR CONFIGURAÇÃO
#
=====
=====

if (!exists("root_dir")) {
  if (requireNamespace("rstudioapi",
quietly = TRUE)) {
    root_dir <-
rstudioapi::selectDirectory(
      caption = "Selecione a pasta raiz
do projeto"
    )
  } else {
    root_dir <- choose.dir()
  }
}

config_path <- file.path(root_dir,
"scripts", "00_config.R")

if (!file.exists(config_path)) {
  stop("Arquivo 00_config.R não
encontrado.")
}

source(config_path)

#
=====
=====
# 0.1. LOG
#
=====
=====

dir.create(complete_outputs_dir,
recursive = TRUE, showWarnings = FALSE)
dir.create(complete_logs_dir, recursive
= TRUE, showWarnings = FALSE)

log_file <- file.path(
  complete_logs_dir,
  "audit_panel_master_2014_2024.txt"
)

if (file.exists(log_file))
  file.remove(log_file)

log_line <- function(...) {
  msg <- paste0(
    "[", format(Sys.time(), "%Y-%m-%d
%H:%M:%S"), "]",
    paste(..., collapse = " ")
  )
}
```

```

    cat(msg, "\n")
    cat(msg, "\n", file = log_file, append
= TRUE)
}

log_line("Início      do      Script
07_audit_panel_master_2014_2024.R")
log_line("root_dir =", root_dir)
log_line("Período =", min(panel_years),
"-", max(panel_years))

#
=====
# 1. LER BASE FINAL
#
=====

panel_path <- file.path(
  complete_outputs_dir,
  "panel_master_2014_2024.csv"
)

if (!file.exists(panel_path)) {
  stop("Base final não encontrada: ",
panel_path)
}

panel_master <- readr::read_csv(
  panel_path,
  show_col_types = FALSE
) |>
  janitor::clean_names() |>
  dplyr::mutate(
    ibge7 =
stringr::str_pad(as.character(ibge7),
7, "left", "0"),
    ano = as.integer(ano)
  )

log_line("Base final lida.")
log_line("Linhas      =",
nrow(panel_master))
log_line("Colunas      =",
ncol(panel_master))

#
=====
# 2. DEFINIR PARÂMETROS ESPERADOS
#
=====

years_expected <- panel_years
expected_rows <- 62 *
length(years_expected)

log_line("Linhas      esperadas      =",
expected_rows)

#
=====
# 3. CHECAGENS ESTRUTURAIS PRINCIPAIS
#
=====

required_cols <- c("ibge7", "ano")

missing_cols <- setdiff(required_cols,
names(panel_master))

if (length(missing_cols) > 0) {
  stop(
    "A base final não possui colunas
obrigatórias: ",
    paste(missing_cols, collapse = ", ")
  )
}

if (nrow(panel_master) != expected_rows)
{
  stop(
    "Número de linhas inesperado: ",
    nrow(panel_master),
    " linhas; esperado = ",
    expected_rows
  )
}

n_muni <-
dplyr::n_distinct(panel_master$ibge7)

if (n_muni != 62) {
  stop(
    "Número de municípios diferente de
62: ",
    n_muni
  )
}

years_present <-
sort(unique(panel_master$ano))

missing_years <-
setdiff(years_expected, years_present)

if (length(missing_years) > 0) {
  stop(
    "Anos ausentes na base final: ",
    paste(missing_years, collapse = ", ")
  )
}

dup_keys <- panel_master |>
  dplyr::count(ibge7, ano) |>
  dplyr::filter(n > 1)

if (nrow(dup_keys) > 0) {
  readr::write_csv(
    dup_keys,
    file.path(complete_outputs_dir,
"audit_duplicate_keys.csv")
  )
}

```

```

    stop("Foram encontradas duplicidades
em ibge7 + ano.")
}

count_by_year <- panel_master |>
  dplyr::count(ano) |>
  dplyr::arrange(ano)

bad_years <- count_by_year |>
  dplyr::filter(n != 62)

if (nrow(bad_years) > 0) {
  stop("Há anos com número de municípios
diferente de 62.")
}

log_line("OK: estrutura do painel
validada.")

#
=====
# 4. VERIFICAR LACUNAS NA GRADE IBGE7 +
ANO
#
=====

full_grid <- tidyr::expand_grid(
  ibge7 =
sort(unique(panel_master$ibge7)),
  ano = years_expected
)

key_holes <- dplyr::anti_join(
  full_grid,
  panel_master |>
    dplyr::distinct(ibge7, ano),
  by = c("ibge7", "ano")
)

if (nrow(key_holes) > 0) {
  readr::write_csv(
    key_holes,
    file.path(complete_outputs_dir,
"audit_key_holes.csv")
  )

  stop(
    "Foram encontradas lacunas na grade
ibge7 + ano. ",
    "Verifique audit_key_holes.csv."
  )
}

log_line("OK: não há lacunas na grade
ibge7 + ano.")

#
=====
# 5. VERIFICAR NOMES DE COLUNAS

```

```

#
=====

if
(any(duplicated(names(panel_master))))
{
  stop("Há nomes de colunas duplicados
na base final.")
}

suspicious_cols <- names(panel_master)[
  stringr::str_detect(names(panel_master)
, "\\..x$|\\.y$")
]

if (length(suspicious_cols) > 0) {
  log_line(
    "WARNING: colunas com sufixo .x/.y
detectadas:",
    paste(suspicious_cols, collapse = ",
")
  )
} else {
  log_line("OK: sem colunas com sufixos
.x/.y.")
}

#
=====
# 6. RELATÓRIO DE VALORES AUSENTES POR
VARIÁVEL
#
=====

na_by_var <- tibble::tibble(
  var = names(panel_master),
  classe = vapply(
    panel_master,
    function(x) paste(class(x),
collapse = ", "),
    character(1)
  ),
  na_count = vapply(
    panel_master,
    function(x) sum(is.na(x)),
    integer(1)
  ),
  na_rate = round(
    vapply(
      panel_master,
      function(x) mean(is.na(x)),
      numeric(1)
    ) * 100,
    2
  )
) |>
  dplyr::arrange(dplyr::desc(na_rate),
var)

na_by_var_path <- file.path(
  complete_outputs_dir,

```

```

"audit_na_by_var.csv"
)

readr::write_csv(na_by_var,
na_by_var_path)

log_line("Relatório de NA por variável
exportado:", na_by_var_path)

#
=====
# 7. RELATÓRIO DE VALORES AUSENTES POR
ANO
#
=====

na_by_year <- panel_master |>
  dplyr::group_by(ano) |>
  dplyr::summarise(
    dplyr::across(
      dplyr::everything(),
      ~ sum(is.na(.x))
    ),
    .groups = "drop"
  ) |>
  dplyr::arrange(ano)

na_by_year_path <- file.path(
  complete_outputs_dir,
  "audit_na_by_year.csv"
)

readr::write_csv(na_by_year,
na_by_year_path)

log_line("Relatório de NA por ano
exportado:", na_by_year_path)

#
=====
# 8. VERIFICAÇÕES ECONÔMICAS BÁSICAS
#
=====

econ_nonnegative_vars <- intersect(
  names(panel_master),
  c(
    "rc_liq",
    "fpm_liq",
    "dc",
    "pib",
    "rc_liq_real",
    "fpm_liq_real",
    "dc_real",
    "pib_real",
    "rc_pc",
    "fpm_pc",
    "dc_pc",
    "pib_pc",
    "rc_pc_real",

```

```

    "fpm_pc_real",
    "dc_pc_real",
    "pib_pc_real",
    "pop",
    "pop_sidra"
  )
)

if (length(econ_nonnegative_vars) > 0) {

  neg_econ <- tibble::tibble(
    var = econ_nonnegative_vars,
    n_negativos = vapply(
      econ_nonnegative_vars,
      function(v) sum(panel_master[[v]]
< 0, na.rm = TRUE),
      integer(1)
    )
  ) |>
  dplyr::filter(n_negativos > 0)

  if (nrow(neg_econ) > 0) {
    readr::write_csv(
      neg_econ,
      file.path(complete_outputs_dir,
"audit_negative_econ_values.csv")
    )

    log_line("WARNING:          valores
negativos encontrados em variáveis
econômicas.")
  } else {
    log_line("OK: variáveis econômicas
sem valores negativos.")
  }
}

#
=====
# 9. VERIFICAR DEFLATOR
#
=====

if ("deflator" %in% names(panel_master))
{

  defl_tbl <- panel_master |>
    dplyr::distinct(ano, deflator) |>
    dplyr::arrange(ano)

  defl_path <- file.path(
    complete_outputs_dir,
    "audit_deflator_by_year.csv"
  )

  readr::write_csv(defl_tbl, defl_path)

  last_year <- max(panel_years)

  defl_last <- defl_tbl |>
    dplyr::filter(ano == last_year) |>
    dplyr::pull(deflator)

```

```

    if (length(defl_last) == 1 &&
abs(defl_last - 1) < 1e-8) {
      log_line("OK: deflator do último ano
é igual a 1.")
    } else {
      log_line("WARNING: deflator do
último ano não é igual a 1.")
    }

    log_line("Tabela do deflator
exportada:", defl_path)
  }

#
=====
# 10. VERIFICAÇÕES CLIMÁTICAS BÁSICAS
#
=====

climate_vars <- names(panel_master)[
stringr::str_detect(names(panel_master)
, "_anual$")
]

log_line("Variáveis climáticas anuais
identificadas =", length(climate_vars))

climate_nonnegative_vars <-
climate_vars[
  stringr::str_detect(
    climate_vars,
    "^(ppt|aet|pet|def|soil|q)_"
  )
]

if (length(climate_nonnegative_vars) >
0) {

  neg_climate <- tibble::tibble(
    var = climate_nonnegative_vars,
    n_negativos = vapply(
      climate_nonnegative_vars,
      function(v) sum(panel_master[[v]]
< 0, na.rm = TRUE),
      integer(1)
    )
  ) |>
  dplyr::filter(n_negativos > 0)

  if (nrow(neg_climate) > 0) {
    readr::write_csv(
      neg_climate,
      file.path(complete_outputs_dir,
"audit_negative_climate_values.csv")
    )

    log_line("WARNING: valores
negativos encontrados em variáveis
climáticas não negativas.")
  } else {
    log_line("OK: variáveis climáticas
não negativas sem valores negativos.")
  }
}

}
}

#
=====
# 11. CHECAR COERÊNCIA DA TEMPERATURA
MÉDIA
#
=====

if (all(c("tmax_anual", "tmin_anual",
"tmean_anual") %in%
names(panel_master))) {

  tmean_check <- panel_master |>
  dplyr::mutate(
    tmean_calc = (tmax_anual +
tmin_anual) / 2,
    diff_tmean = tmean_anual -
tmean_calc
  ) |>
  dplyr::summarise(
    media_diferenca =
mean(diff_tmean, na.rm = TRUE),
    p95_abs_diferenca =
stats::quantile(abs(diff_tmean), 0.95,
na.rm = TRUE),
    max_abs_diferenca =
max(abs(diff_tmean), na.rm = TRUE)
  )

  tmean_check_path <- file.path(
    complete_outputs_dir,
    "audit_tmean_check.csv"
  )

  readr::write_csv(tmean_check,
tmean_check_path)

  log_line("Checagem tmean exportada:",
tmean_check_path)
}

#
=====
# 12. IDENTIFICAR OUTLIERS SIMPLES
#
=====

outlier_vars <- intersect(
  names(panel_master),
  c(
    "rc_liq",
    "fpm_liq",
    "dc",
    "pib",
    "rc_pc_real",
    "fpm_pc_real",
    "dc_pc_real",
    "pib_pc_real",

```

```

    "ppt_anual",
    "tmean_anual",
    "pdsi_anual"
  )
)

top_outliers <- function(df, var, n =
10) {

  df |>
    dplyr::select(
      ibge7,
      dplyr::any_of("nm_mun"),
      ano,
      dplyr::all_of(var)
    ) |>
    dplyr::filter(!is.na(.data[[var]]))
|>

dplyr::arrange(dplyr::desc(abs(.data[[v
ar]]))) |>
  dplyr::slice_head(n = n) |>
  dplyr::mutate(var = var) |>
  dplyr::relocate(var)
}

if (length(outlier_vars) > 0) {

  outlier_tbl <- dplyr::bind_rows(
    lapply(outlier_vars, function(v)
top_outliers(panel_master, v, 10))
  )

  outlier_path <- file.path(
    complete_outputs_dir,
    "audit_outliers_top10.csv"
  )

  readr::write_csv(outlier_tbl,
outlier_path)

  log_line("Relatório de outliers
exportado:", outlier_path)
}

#
=====
# 13. GERAR RESUMO FINAL DA AUDITORIA
#
=====

summary_tbl <- tibble::tibble(
  item = c(
    "linhas",
    "colunas",
    "municipios",
    "ano_inicial",
    "ano_final",
    "colunas_com_na",
    "colunas_com_na_maior_20pct"
  ),
  valor = c(
    nrow(panel_master),
    ncol(panel_master),

dplyr::n_distinct(panel_master$ibge7),
    min(panel_master$ano),
    max(panel_master$ano),
    sum(na_by_var$na_count > 0),
    sum(na_by_var$na_rate > 20)
  )
)

summary_path <- file.path(
  complete_outputs_dir,
  "audit_summary.csv"
)

readr::write_csv(summary_tbl,
summary_path)

log_line("Resumo da auditoria
exportado:", summary_path)

#
=====
# 14. FINALIZAÇÃO
#
=====

log_line("Auditoria concluída com
sucesso.")
log_line("Log salvo em:", log_file)

cat("\n=====
=====\\n")
cat("SCRIPT 07 CONCLUÍDO\\n")
cat("Auditoria finalizada.\\n")
cat("Log gerado em:\\n")
cat(log_file, "\\n")
cat("=====
=====\\n")

```

APÊNDICE I – Evidências dos produtos gerados pelo pipeline

Figura I.1 – Arquivos finais produzidos pelo pipeline

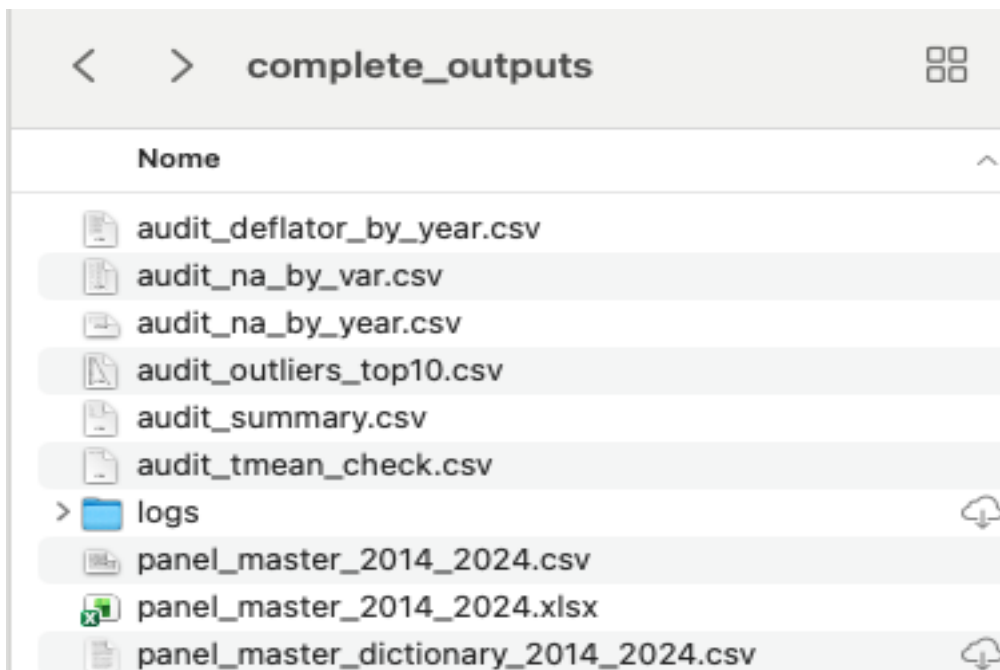


Figura I.2 – Base consolidada gerada pelo pipeline

panel_master_2014_2024 — Salvo													
Página Inicial Inserir Desenhar Layout da Página Fórmulas Dados Revisão Exibir Automatizar													
Calibri (Corpo) 11 A ⁺ A ⁻ Geral Formatação Formatar cor Estilos de Célula													
A1 ano													
	A	B	C	D	E	F	G	H	I	J	K	L	
	ano	ibge7	nm_mun	rc_liq	fpm_liq	dc	pop	pop_sic	piib	ipca_d	ipca_dec	deflato	
1	2014	1300029	Alvarães	29510765,87	8098595,73	21610450	15357	15357	1,08E+08	4059,86	7100,5	1,748952	
2	2014	1300060	Amatutã	26083749,75	9311133,17	17778489	10644	10644	63736000	4059,86	7100,5	1,748952	
3	2014	1300086	Anamá	26394887,36	9311133,17	15913419	11981	11981	86080000	4059,86	7100,5	1,748952	
4	2014	1300102	Anori	34140648,79	13966699,97	26172060	18826	18826	1,69E+08	4059,86	7100,5	1,748952	
5	2014	1300144	Apuí	38305283,33	13966699,97	30849408	20258	20258	1,93E+08	4059,86	7100,5	1,748952	
6	2014	1300201	Atalaia do Norte	41129162,58	13966699,97	30277954	17658	17658	1,25E+08	4059,86	7100,5	1,748952	
7	2014	1300300	Autazes	66189354,15	19645302,24	46993840	36301	36301	2,39E+08	4059,86	7100,5	1,748952	
8	2014	1300409	Barcelos	47769121,14	18622266,74	37241494	27273	27273	1,48E+08	4059,86	7100,5	1,748952	
9	2014	1300508	Barreirinha	61405126,94	16294483,31	48601269	30202	30202	2,11E+08	4059,86	7100,5	1,748952	
10	2014	1300607	Benjamin Constant	77465916,91	20950050,05	58424876	38533	38533	2,71E+08	4059,86	7100,5	1,748952	
11	2014	1300631	Beruri	44600659,22	13966699,97	34984726	17755	17755	1,21E+08	4059,86	7100,5	1,748952	
12	2014	1300680	Boa Vista do Ramos	32214352,15	11638916,52	25603317	17248	17248	98384000	4059,86	7100,5	1,748952	
13	2014	1300706	Boca do Acre	57561458,1	18622266,73	46369727	33148	33148	2,78E+08	4059,86	7100,5	1,748952	
14	2014	1300805	Borba	73716244,79	20950050,05	57802601	38688	38688	2,55E+08	4059,86	7100,5	1,748952	
15	2014	1300839	Caapiranga	31163771,28	9311133,17	24354823	12214	12214	1,31E+08	4059,86	7100,5	1,748952	
16	2014	1300904	Canutama	26184272,82	8082211,89	24707413	14944	14944	88213000	4059,86	7100,5	1,748952	
17	2014	1301001	Carauari	52614599,71	16294483,31	42374934	27645	27645	2,36E+08	4059,86	7100,5	1,748952	
18	2014	1301100	Careiro	61745217,74	18622266,73	45674911	35938	35938	2,18E+08	4059,86	7100,5	1,748952	
19	2014	1301159	Careiro da Várzea	45414235,89	16294482,8	35167758	27357	27357	2,41E+08	4059,86	7100,5	1,748952	
20	2014	1301209	Coari	254938424,3	20533678,69	2,74E+08	82209	82209	3,11E+09	4059,86	7100,5	1,748952	
21	2014	1301308	Codajás	41665009,53	16294483,31	31508193	26242	26242	6,52E+08	4059,86	7100,5	1,748952	
22	2014	1301407	Eirunepé	63714873,17	18622266,73	47278514	33580	33580	3,83E+08	4059,86	7100,5	1,748952	
23	2014	1301506	Envira	40864481,57	13966699,97	28641187	18422	18422	1,25E+08	4059,86	7100,5	1,748952	
24	2014	1301605	Fonte Boa	46177192,4	9477082,65	43767594	21295	21295	1,52E+08	4059,86	7100,5	1,748952	
25	2014	1301654	Guajará	34155487,75	11638916,52	26638373	15561	15561	1E+08	4059,86	7100,5	1,748952	
26	2014	1301704	Humaitá	76913862,66	15795137,55	74775152	50230	50230	4,23E+08	4059,86	7100,5	1,748952	
27	2014	1301803	IPIXUNA	37312316,2	16294483,31	27050756	26118	26118	1,12E+08	4059,86	7100,5	1,748952	
28	2014	1301852	Irlanduba	89668143,62	15837514,03	80762983	45250	45250	5,3E+08	4059,86	7100,5	1,748952	
29	2014	1301902	Itacoatiara	194602740,6	34916750,32	1,66E+08	95714	95714	1,59E+09	4059,86	7100,5	1,748952	
30	2014	1301951	Itamarati		6983349,8	19840671	8205	8205	75976000	4059,86	7100,5	1,748952	
31	2014	1302009	Itapiranga	24632221,76	6983349,8	17308582	8864	8864	80542000	4059,86	7100,5	1,748952	

Figura I.3 – Dicionário de variáveis gerado automaticamente

panel_master_dictionary_2014_2024

var	origem	fonte	unidade	descricao	classe	n_unique	na
aet_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a evapotranspiração real anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
def_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a déficit hídrico anual, extraída do TerraClimate e agregada para frequência anual.	numeric	675	
pdsi_anual	climatica_anual	TerraClimate	índice adimensional	Variável climática municipal referente a índice de severidade de seca de Palmer, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
pet_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a evapotranspiração potencial anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
ppt_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a precipitação anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
q_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a escoamento superficial anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
soil_anual	climatica_anual	TerraClimate	mm	Variável climática municipal referente a umidade do solo anual, extraída do TerraClimate e agregada para frequência anual.	numeric	678	
tmax_anual	climatica_anual	TerraClimate	°C	Variável climática municipal referente a temperatura máxima anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
tmean_anual	climatica_anual	TerraClimate	°C	Variável climática municipal referente a temperatura média anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
tmin_anual	climatica_anual	TerraClimate	°C	Variável climática municipal referente a temperatura mínima anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
ws_anual	climatica_anual	TerraClimate	m/s	Variável climática municipal referente a velocidade do vento anual, extraída do TerraClimate e agregada para frequência anual.	numeric	682	
pop	economica_demografia	SICONFI/STN e SIDRA/IBGE	habitantes	População residente utilizada no painel.	numeric	495	
pop_sidra	economica_demografia	SICONFI/STN e SIDRA/IBGE	habitantes	População residente obtida pelo SIDRA/IBGE.	numeric	556	
dc	economica_fiscal	SICONFI/STN	R\$ correntes	Despesa Corrente municipal liquidada em valores correntes.	numeric	671	
dc_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variável per capita calculada pela divisão do valor correspondente pela população.	numeric	674	
dc_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, preços reais	Variável monetária convertida para valores reais pelo deflator.	numeric	674	
dc_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para valores reais pelo deflator.	numeric	672	
fpm_liq	economica_fiscal	SICONFI/STN	R\$ correntes	Fundo de Participação dos Municípios líquido em valores correntes.	numeric	351	
fpm_liq_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para valores reais pelo deflator.	numeric	350	
fpm_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variável per capita calculada pela divisão do valor correspondente pela população.	numeric	673	
fpm_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, preços reais	Variável monetária convertida para valores reais pelo deflator.	numeric	673	
rl_liq	economica_fiscal	SICONFI/STN	R\$ correntes	Receita Corrente Líquida municipal em valores correntes.	numeric	670	
re_liq_real	economica_fiscal	SICONFI/STN	R\$ reais	Variável monetária convertida para valores reais pelo deflator.	numeric	671	
rc_pc	economica_fiscal	SICONFI/STN	R\$/hab	Variável per capita calculada pela divisão do valor correspondente pela população.	numeric	673	
rc_pc_real	economica_fiscal	SICONFI/STN	R\$/hab, preços reais	Variável monetária convertida para valores reais pelo deflator.	numeric	673	
deflator	economica_precos	SIDRA/IBGE - IPCA	adimensional	Fator de deflacionamento utilizado para converter valores nominais em reais.	numeric	11	
ipca_dec	economica_precos	SIDRA/IBGE - IPCA	número-índice	IPCA de dezembro do ano de referência.	numeric	11	
ipca_dec_base	economica_precos	SIDRA/IBGE - IPCA	número-índice	IPCA de dezembro do ano-base.	numeric	1	
piB	economica_produto	SIDRA/IBGE - PIB Municipal	R\$ correntes	Produto Interno Bruto municipal em valores correntes.	numeric	620	
piB_pc	economica_produto	SIDRA/IBGE - PIB Municipal	R\$/hab	Variável per capita calculada pela divisão do valor correspondente pela população.	numeric	621	
piB_pc_real	economica_produto	SIDRA/IBGE - PIB Municipal	R\$/hab, preços reais	Variável monetária convertida para valores reais pelo deflator.	numeric	621	
piB_real	economica_produto	SIDRA/IBGE - PIB Municipal	R\$ reais	Variável monetária convertida para valores reais pelo deflator.	numeric	621	
ano	identificacao	IBGE / integração municipal	ano	Ano de referência da observação.	integer	11	
ibge7	identificacao	IBGE / integração municipal	código IBGE	Código IBGE municipal padronizado com sete dígitos.	character	62	
nm_mun	identificacao	IBGE / integração municipal	nome do município	Nome do município.	character	62	

Figura I.4 – Relatórios de auditoria gerados automaticamente

Nome
 audit_deflator_by_year.csv
 audit_na_by_var.csv
 audit_na_by_year.csv
 audit_outliers_top10.csv
 audit_summary.csv
 audit_tmean_check.csv

Figura I.5 – Registros de execução (logs)

```

panel_master_build_log.txt

[2026-05-26 16:52:50] Início do Script 06_build_panel_master_2014_2024.R
[2026-05-26 16:52:50] root_dir = /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)
[2026-05-26 16:52:50] Período = 2014 - 2024
[2026-05-26 16:52:50] Arquivo climático = /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/climate_outputs/final_panel_2014_2024/climate_panel_full_2014_2024.csv
[2026-05-26 16:52:50] Arquivo econômico = /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/econ_outputs/final/econ_panel_final_2014_2024.csv
[2026-05-26 16:52:51] Clima: linhas = 682 | colunas = 13
[2026-05-26 16:52:51] Econ: linhas = 682 | colunas = 24
[2026-05-26 16:52:51] Validação estrutural concluída em clima e economia.
[2026-05-26 16:52:51] Cobertura de chaves 1:1 confirmada.
[2026-05-26 16:52:51] Panel master criado.
[2026-05-26 16:52:51] Linhas = 682
[2026-05-26 16:52:51] Colunas = 35
[2026-05-26 16:52:51] Dicionário de variáveis gerado.
[2026-05-26 16:52:51] CSV exportado: /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/complete_outputs/panel_master_2014_2024.csv
[2026-05-26 16:52:51] XLSX exportado: /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/complete_outputs/panel_master_2014_2024.xlsx
[2026-05-26 16:52:51] Dicionário exportado: /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/complete_outputs/panel_master_dictionary_2014_2024.csv
[2026-05-26 16:52:51] Script 06 concluído.
[2026-05-26 16:52:51] Log salvo em: /Users/felipeesteves/Library/CloudStorage/OneDrive-Pessoal/MESTRADO/Manual Técnico (PTT)/data_processed/complete_outputs/logs/panel_master_build_log.txt

```