INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS - *CAMPUS* SABARÁ
ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Guilherme Lopes Pereira

MODELAGEM E ANÁLISE DE DADOS DE MANUTENÇÃO DE CORREIAS TRANSPORTADORAS

Belo Horizonte - MG
2026

GUILHERME LOPES PEREIRA

MODELAGEM E ANÁLISE DE DADOS DE MANUTENÇÃO DE CORREIAS TRANSPORTADORAS

Trabalho de conclusão de curso apresentado ao Curso de Engenharia de Controle e Automação do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - *Campus* Sabará para a obtenção do título de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Luiz Guilherme Hilel Drumond Silveira

Belo Horizonte - MG
2026

Pereira, Guilherme Lopes

P436m Modelagem e análise de dados de manutenção de correias transportadoras [manuscrito]. / Guilherme Lopes Pereira. - 2026.

76 f. : il.

Orientador: Prof. Me. Luiz Guilherme Hilel Drumond Silveira.

Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Controle e Automação) – Instituto Federal de Minas Gerais, *Campus* Sabará.

1. Banco de dados - Modelagem. – Monografia. 2. Banco de dados - Gerência. – Monografia. 3. Sistemas de informação gerencial. – Monografia. 4. MySQL (Recurso eletrônico). – Monografia. 5. Correias transportadoras - Manutenção. – Monografia. I. Silveira, Luiz Guilherme Hilel Drumond. II. Instituto Federal de Minas Gerais, *Campus* Sabará. III. Bacharelado em Engenharia de Controle e Automação. IV. Título.

CDU 004.65

César dos Santos Moreira / CRB6-2229
Biblioteca do IFMG *Campus* Sabará

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
Campus Sabará
Diretoria de Ensino, Pesquisa e Extensão
Conselho de Área - Informática e Comunicação
Rodovia MGC 262, Km 10 - Bairro Sobradinho - CEP 34590-390 - Sabará - MG
- www.ifmg.edu.br

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Aos seis dias do mês de fevereiro do ano de dois mil e vinte e seis, às dezenove horas, sob a presidência do professor Luiz Guilherme Hilel Drumond, reuniu-se a banca examinadora composta pelos professores abaixo relacionados, para a defesa do Trabalho de Conclusão de Curso (TCC) do discente Guilherme Lopes Pereira, matrícula nº 0049908, do curso de Engenharia de Controle e Automação, do IFMG campus Sabará.

O trabalho intitulado ANÁLISE E MODELAGEM DE DADOS NA MANUTENÇÃO PREDITIVA DE UMA CORREIA TRANSPORTADORA foi apresentado e submetido à apreciação da banca. Após exposição, arguição e deliberação, a banca atribuiu a nota final de 62 pontos (de um total de 80). Somados aos 16 pontos atribuídos ao aluno pelo docente responsável pela disciplina de Projeto II, o aluno ficou com um total de 78 pontos, resultando em aprovado, condicionado ao cumprimento das orientações e prazos estabelecidos pelas normas acadêmicas institucionais.

Compuseram a Banca Examinadora:

- Membro 1 (Presidente): Luiz Guilherme Hilel Drumond
- Membro 2: Erick Fonseca Boaventura
- Membro 3: Rodrigo Hiroshi Murofushi

O discente deverá apresentar a versão final do trabalho em formato PDF e depositá-la no repositório institucional até o dia vinte e sete de fevereiro de dois mil e vinte e seis. O não cumprimento dessas exigências implicará na não contabilização das horas referentes aos componentes curriculares de TCC I e TCC II no sistema acadêmico.

A sessão foi encerrada às vinte horas e quarenta minutos. Para constar, eu, Luiz Guilherme Hilel Drumond, redigi a presente ata que após lida publicamente, foi aprovada e assinada pelos membros da banca examinadora.
Sabará, 06 de fevereiro de 2026.

Sabará, 07 de fevereiro de 2026.

Documento assinado eletronicamente por **Luiz Guilherme Hilel Drumond Silveira**, **Professor**, em 07/02/2026, às 13:19, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Rodrigo Hiroshi Murofushi**, **Professor EBTT**, em 07/02/2026, às 17:50, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Erick Fonseca Boaventura**, **Professor EBTT**, em 07/02/2026, às 18:46, conforme Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade do documento pode ser conferida no site https://sei.ifmg.edu.br/consultadocs informando o código verificador **2614168** e o código CRC **D3F395CB**.

23714.001466/2025-48	2614168v1

Dedico esta monografia aos futuros analistas, cientistas e engenheiros de dados, e ao Instituto Federal de Minas Gerais, realizada em prol do conhecimento.

AGRADECIMENTOS

Agradeço ao meu círculo íntimo familiar — meus amados pais, amado irmão mais velho e minha amada avó por lado de mãe por prover as condições e o apoio essencial que me trouxeram até o momento de realização deste trabalho.

Também agradeço ao meu orientador, Luiz Guilherme H. D. Silveira, e ao professor Rodrigo H. Murofushi, que me deram todo o apoio acadêmico e institucional para a conclusão deste trabalho e para minha graduação.

“Education is not preparation for life; education is life it self."

John Dewey

RESUMO

A gestão da manutenção aplicada a correias transportadoras desempenha papel fundamental na confiabilidade operacional e na otimização de recursos em ambientes industriais. Nesse contexto, este trabalho propõe a informatização do registro, armazenamento e análise de dados de manutenção de uma correia transportadora de industrial, por meio da modelagem de uma base de dados estruturada e do desenvolvimento de um relatório analítico em Microsoft Power BI. Os dados utilizados no projeto são fictícios, porém elaborados com base em um cenário real, de modo a preservar a coerência operacional e a aplicabilidade prática da solução proposta. A arquitetura do sistema contempla a coleta de parâmetros operacionais simulados de sensores instalados em componentes críticos da correia transportadora, com armazenamento em bancos de dados relacionais e plataformas em nuvem, utilizando MySQL e Google Planilhas. A extração, transformação e carregamento dos dados (ETL) são realizados por meio do Power Query, com o apoio das linguagens M e DAX para modelagem analítica, enquanto a linguagem Python é empregada na geração de dados, automação de processos e desenvolvimento de interfaces de inserção. Como resultado, é desenvolvido um relatório dinâmico que possibilita a consulta histórica, a correlação entre diagnósticos, processos e ordens de serviço, além da visualização de indicadores relevantes para a tomada de decisão. A solução demonstra potencial para reduzir o tempo de diagnóstico de falhas, aumentar a taxa de amostragem das informações e promover a padronização dos procedimentos de manutenção, contribuindo para ganhos operacionais no curto e médio prazo.

Palavras-chave: Correia Transportadora. Manutenção. Base de dados. MySQL. Power BI.

ABSTRACT

Maintenance management applied to conveyor belts plays a fundamental role in operational reliability and resource optimization in industrial environments. In this context, this work proposes the computerization of the recording, storage, and analysis of maintenance data for an industrial conveyor belt through the modeling of a structured database and the development of an analytical report in Microsoft Power BI. The data used in the project are fictitious; however, they are based on a real-world scenario in order to preserve operational coherence and the practical applicability of the proposed solution. The system architecture includes the collection of simulated operational parameters from sensors installed on critical components of the conveyor belt, with storage in relational databases and cloud-based platforms using MySQL and Google Sheets. The data extraction, transformation, and loading (ETL) process is carried out using Power Query, supported by the M and DAX languages for analytical modeling, while the Python programming language is employed for data generation, process automation, and the development of data entry interfaces. As a result, a dynamic report is developed, enabling historical data consultation, correlation between diagnostics, processes, and work orders, as well as the visualization of relevant indicators to support decision-making. The proposed solution demonstrates potential to reduce fault diagnosis time, increase the data sampling rate, and promote the standardization of maintenance procedures, contributing to operational gains in the short and medium term.

Keywords: Conveyor Belt. Maintenance. Database. MySQL. Power BI.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3NF	Third Normal Form
BI	Business Inteligence
CMMS	Computerized Maintenance Management System
CSV	Comma-separated Values
CT	Correia Transportadora
DAX	Data Analysis Expressions
DER	Diagrama Entidade-relacionamento
EbM	Evidence-based Maintenance
ETL	Extract, Transform, Load
FK	Foreign Key
GUI	Graphic User Interface
IA	Inteligência Artificial
IFMG	Instituto Federal de Minas Gerais
KPI	Key Process Indicator
ML	Machine Learning
MTTA	Medium Time to Attend
MTTR	Medium Time Between Failures
OS	Ordem de Serviço
PK	Primary Key
SGBD	Sistema Gerenciador de Banco de Dados
SLA	Service Level Agreement
UM	Unidade de medida
VSC	Visual Studio Code

SUMÁRIO

1 INTRODUÇÃO

O registro físico (papel, fichas, pastas e planilhas descentralizadas) tende a transformar a manutenção em um processo pouco rastreável e pouco auditável: informações se perdem, chegam incompletas, não seguem um padrão de preenchimento e ficam retidas no local onde foram arquivadas. O efeito prático é que a organização passa a ter histórico fragmentado de falhas, intervenções e custos — o que dificulta análises de recorrência, priorização de ativos críticos, comprovação de conformidade e até a comparação de desempenho entre períodos/equipes. Do ponto de vista gerencial, isso mina a construção de indicadores e o aprendizado institucional. Tsang, Jardine e Kolodny (1999) destacam que acompanhar o desempenho das operações de manutenção é uma questão-chave de gestão em organizações com forte investimento em ativos, justamente porque os dispêndios são relevantes e precisam ser governados com informação confiável.

Na ausência de um sistema informatizado de gestão de manutenção, a operação também perde a capacidade de realizar o ciclo entre ocorrência, registro, análise e decisão, ficando mais reativa e dependente de memória tácita. Isso é particularmente crítico quando se busca evoluir para abordagens mais analíticas (como manutenção baseada em condição), pois tais estratégias dependem de dados coletados e organizados — seja de inspeções, seja de monitoramento — para sustentar decisões oportunas.

Em ambientes industriais, a dependência de registros físicos (ordens de serviço em papel, apontamentos manuais e arquivos locais) agrava problemas que já são estruturalmente críticos: alta cadência de produção, múltiplos turnos, grande volume de intervenções, requisitos de segurança e conformidade, além da pressão por disponibilidade (tempo de parada — *downtime* — com custo elevado). Nesse cenário, a informação "nasce" no chão de fábrica, mas frequentemente chega ao nível gerencial atrasada, inconsistente ou incomparável entre linhas/áreas — o que prejudica o planejamento, a gestão de sobressalentes, a priorização por criticidade e a construção de indicadores de forma confiável. Muchiri *et al.* (2011) ressaltam justamente que a gestão de desempenho da manutenção, no contexto industrial, precisa estar alinhada e integrada às demais funções organizacionais — sobretudo à produção — o que pressupõe dados consistentes e fluxos informacionais capazes de sustentar essa interação; sem um sistema informatizado, essa integração tende a ser frágil e reativa.

A informatização dos dados é uma tendência em pauta no planeta desde os anos 1980, possibilitando uma análise mais ágil e veraz de grandes quantidades de dados que, em diversas empresas de grande porte, se encontravam um dia segmentadas e espalhadas por diversas partes do globo. O desenvolvimento de sistemas e hospedagem em plataformas gera benefícios como a melhora na gestão e análise de dados, possibilidade de acesso em tempo real, armazenamento seguro, redução de erro humano e a possibilidade da criação de estratégias e soluções para a resolução de problemas para gestão de riscos (STüRMER; NUSSBAUMER; STöCKLI, 2021).

Nesse contexto, é proposto um projeto de informatização do registro e análise de dados de

manutenção de correias transportadoras (CTs — ver figura 1) industriais. A hipótese estabelecida aqui é uma migração do ambiente físico para o digital, como discutido anteriormente, em que dados fictícios baseados em um cenário real — configurando o conceito de *mock data* — representariam informações de quebras obtidas por aferições dos componentes dessas CTs. No entanto, essa migração acompanhará diversos desafios, uma vez que é necessário definir dados pertinentes ou não a serem hospedados e soluções confiáveis a longo prazo de hospedagem, modelagem e análise de dados.

Figura 1 – Correira transportadora industrial.

Fonte: https://globalbelt.com.br/correia-transportadora-de-graos-eficiencia-no-setor-agricola/. Acesso em: 10 jan. 2026.

Para a conversão de dados brutos em informações úteis de negócio, é necessário que a base de dados a ser utilizada possua informações de data, prazo, localidade, diagnóstico técnico e identificação única. Esses dados permitirão a criação de análises de *lead-time*, — delta de tempo entre o início e o fim de um processo — possibilitando a medição de performance de processos e serviços. Também será possível criar análises históricas, para a identificação de padrões, tendências e relações de causa e efeito, e análises comparativas por período, unidade e tipo de ocorrência, evidenciando gargalos operacionais e recorrências. Adicionalmente, a presença de chaves únicas e registros padronizados viabiliza a rastreabilidade ponta a ponta (da abertura ao encerramento), a segmentação por criticidade e categoria de falha, e a construção de indicadores como tempo médio de atendimento e reparo (MTTA e MTTR, na sigla em inglês),

taxa de retrabalho, cumprimento de SLA (*Service Level Agreement*, no inglês, que define os padrões, prazos, métricas e expectativas de qualidade entre um prestador de serviço e um cliente), além da confiabilidade de ativos quando aplicável (por exemplo, por meio de séries temporais e distribuição de falhas).

1.1 Objetivos

1.1.1 *Objetivo geral*

Informatizar o registro histórico e a gestão da manutenção dos componentes de uma correia transportadora através do desenvolvimento de um relatório dinâmico em Power BI alimentado por uma base de dados em nuvem fundamentada em um cenário real.

1.1.2 *Objetivos específicos*

Para alcançar o objetivo geral apresentado, é necessário:

1. Estabelecer quais dados devem ser utilizados para se obter uma base de dados íntegra e fiel ao processo;
2. Projetar a estrutura de dados que irá compor a base a ser hospedada *online*;
3. Hospedar essa base de dados em plataformas modulares e eficazes;
4. Elaborar um relatório dinâmico em Power BI completo e que provenha boas análises de negócio.

1.2 Justificativa

O armazenamento físico ou local de dados representa uma potencial perda competitiva perante outros modelos digitais de gestão, podendo gerar atrasos de diagnóstico e tomada de decisão estratégica, além de alta vulnerabilidade a problemas de fator humano e ambiental, como ausência de pessoal e incêndios. Ademais, o planejamento e gestão informatizada da manutenção permite a redução de circulação de pessoal e menor exposição a riscos de trabalho, direcionando os operadores para onde eles são necessários quando eles são necessários, o que também resulta em ganho econômico para a companhia. Ao centralizar e padronizar o histórico de intervenções, falhas, inspeções e recursos empregados, a organização aumenta a rastreabilidade e a auditabilidade do processo, reduzindo retrabalho e inconsistências de registro, além de facilitar a conformidade com requisitos internos e externos (qualidade, segurança e indicadores contratuais). Essa centralização também fortalece o planejamento ao melhorar a previsibilidade de paradas, a coordenação entre manutenção e operação, e a gestão de sobressalentes, mitigando indisponibilidades causadas por falta de peças ou alocação inadequada de mão de obra.

A implementação de uma solução de BI apresenta um elevado ganho na gestão da manutenção das CTs, uma vez que proporciona maior controle sobre a mesma e com dados facilmente atualizáveis, juntamente da criação de análises e estratégias de negócio mais eficientes. Com painéis e métricas consolidadas, torna-se viável acompanhar desempenho por ativo e por unidade, identificar recorrências e perdas crônicas, medir *lead-time* e cumprimento de prazos, e sustentar decisões baseadas em evidências, como priorização por criticidade, balanceamento do *backlog* e avaliação de custo por tipo de falha. Adicionalmente, a transparência operacional viabilizada pelo BI reduz a dependência de conhecimento tácito, favorece a continuidade do processo entre turnos e equipes e acelera a resposta a eventos críticos. Por fim, quando integrada a rotinas de registro consistentes, a camada analítica abre espaço para evolução gradual de maturidade — de uma manutenção predominantemente reativa para práticas mais planejadas e orientadas por dados, com potencial de reduzir indisponibilidade, custos indiretos e riscos operacionais.

1.3 Organização do Texto

Este trabalho está organizado da seguinte forma:

1. **Introdução:** Apresenta o panorama geral do estudo, a problemática investigada e os objetivos propostos, contemplando o objetivo geral e os objetivos específicos do trabalho.
2. **Referencial teórico:** Desenvolve a base conceitual relacionada ao tema, abordando teorias, conceitos e estudos anteriores que sustentam e fundamentam a pesquisa.
3. **Metodologia:** Descreve os procedimentos metodológicos adotados para a coleta e análise dos dados, detalhando as etapas do desenvolvimento, bem como as ferramentas e técnicas utilizadas.
4. **Resultados:** Expõe os principais resultados obtidos, fazendo uso de figuras resultantes do desenvolvimento e quadros de síntese aliados ao texto, a fim de evidenciar os produto final do processo de desenvolvimento.
5. **Conclusão e trabalhos futuros:** Sintetiza os resultados alcançados e as conclusões do estudo, discutindo suas implicações e sugerindo possíveis caminhos para pesquisas e desenvolvimentos futuros.
6. **Referências bibliográficas:** Apresenta a relação completa das fontes consultadas e citadas ao longo do trabalho, de acordo com as normas acadêmicas de formatação vigentes.
7. **Apêndice**: Apresenta os principais códigos, sejam por relevância para o processo ou complexidade, em todas as linguagens utilizadas para o desenvolvimento do projeto, juntamente com a página completa do relatório para a solução desenvolvida em Power BI.

2 REVISÃO BIBLIOGRÁFICA

2.1 Fundamentação Teórica

A gestão da manutenção industrial tem passado por uma profunda transformação, impulsionada pela necessidade de aumentar a eficiência operacional e reduzir custos. Ela é vista com uma perspectiva estratégica que a posiciona como um contribuinte direto para a lucratividade ((SHERWIN, 2000)). Essa mudança de paradigma é sustentada pela aplicação de tecnologias de informação que permitem uma análise de dados mais sofisticada, viabilizando a otimização de todo o ciclo de vida dos ativos. Este capítulo explora os conceitos teóricos que formam a base para uma gestão da manutenção orientada por dados, abrangendo as estratégias de intervenção, a modelagem de dados e as ferramentas de *Business Intelligence* (BI).

As estratégias de manutenção evoluíram de abordagens puramente reativas para modelos proativos e analíticos. A literatura classifica as estratégias em um espectro que vai da manutenção corretiva (reação à falha) à manutenção preventiva (baseada em tempo ou uso) e, finalmente, à manutenção preditiva (baseada na condição do ativo) ((GARG; DESHMUKH, 2006)). O avanço em direção a estratégias mais proativas depende fundamentalmente da capacidade de coletar e analisar dados de desempenho dos equipamentos. A análise sistemática de dados históricos de falhas, reparos e intervenções permite identificar padrões de comportamento dos ativos, otimizar os intervalos de manutenção preventiva e alocar recursos de forma mais eficiente. Essa abordagem orientada por dados transforma a manutenção de uma atividade meramente reativa em um processo estratégico de tomada de decisão, onde cada intervenção é justificada por evidências concretas extraídas do histórico operacional dos equipamentos ((SHERWIN, 2000)).

A eficácia da análise de dados de manutenção está diretamente ligada à qualidade de sua estruturação. Um banco de dados bem projetado é o alicerce para qualquer sistema de informação de manutenção ((DUARTE; CUNHA; CRAVEIRO, 2013)). O processo de projeto de um banco de dados relacional inicia-se com a modelagem conceitual, utilizando o Diagrama Entidade-Relacionamento (DER). O DER é uma representação gráfica que define as entidades principais de um domínio, seus atributos e os relacionamentos entre elas. Essa modelagem conceitual é então traduzida em um modelo lógico, onde se aplica o processo de normalização. A normalização, conforme introduzida por Codd (1970) em seu trabalho seminal sobre o modelo relacional, é um processo formal para organizar as tabelas de forma a minimizar a redundância de dados e evitar anomalias de inserção, atualização e exclusão. Atingir a Terceira Forma Normal (3NF — regra de normalização de bancos de dados relacionais que visa eliminar a redundância e garantir a integridade dos dados, removendo dependências transitivas) é um objetivo comum, pois garante um alto grau de integridade dos dados sem comprometer excessivamente o desempenho das consultas.

Com os dados devidamente estruturados, o próximo passo é transformá-los em inteligência acionável, um processo conhecido como *Business Intelligence*. O BI engloba um conjunto de

tecnologias e processos que permitem às organizações coletar dados de múltiplas fontes, prepará-los para análise e criar visualizações que facilitem a compreensão e a tomada de decisão ((LLAVE, 2017)). A plataforma Microsoft Power BI é uma ferramenta de BI líder de mercado que se destaca por sua capacidade de se conectar a diversas fontes de dados, sua interface de usuário intuitiva e suas poderosas capacidades de visualização. No contexto da manutenção, o Power BI permite que os gestores criem *dashboards* dinâmicos para monitorar em tempo real os indicadores-chave de desempenho (*Key Process Indicators*, no inglês — KPI) de manutenção, explorar as causas-raiz das falhas e comunicar os resultados de forma clara e impactante. A linguagem DAX (*Data Analysis Expressions*) é um dos principais diferenciais do Power BI, oferecendo uma vasta biblioteca de funções para a criação de medidas e cálculos personalizados, permitindo análises complexas que seriam difíceis de realizar com ferramentas de planilhas tradicionais ((RUSSO; FERRARI, 2020)). A combinação de um banco de dados bem modelado com as capacidades analíticas do Power BI cria um ecossistema poderoso para a gestão da manutenção baseada em evidências.

2.2 Trabalhos Relacionados

A transição de paradigmas na gestão da manutenção, de um modelo puramente reativo para estratégias informadas por dados, representa um dos pilares da moderna engenharia de produção e automação. A capacidade de coletar, analisar e interpretar dados históricos de operação e falha é o que permite a implementação de táticas de manutenção de forma eficiente e otimizada. Esta abordagem busca um equilíbrio entre o custo da intervenção e o risco da falha, distanciando-se tanto da manutenção de emergência, que é dispendiosa, economicamente ineficaz e gera riscos humanos, quanto da manutenção preditiva que, embora avançada, pode demandar investimentos significativos em sensores e algoritmos complexos.

A fundação de qualquer estratégia de manutenção orientada a dados reside na qualidade e na estrutura das informações coletadas. Stürmer, Nussbaumer e Stöckli (2021) argumentam que a digitalização dos processos de manutenção é um passo indispensável, pois não apenas facilita o acesso à informação em tempo real, mas também constrói um repositório histórico robusto, essencial para análises de tendências e para a tomada de decisões estratégicas. Contudo, a simples digitalização gera um novo desafio: o gerenciamento de grandes volumes de dados. Ojha, Singh e Singh (2017) abordam essa questão no contexto de *big data*, enfatizando que a utilidade de um vasto conjunto de dados está intrinsecamente ligada à sua correta classificação e estruturação. Em um trabalho seminal que antecede a popularização do termo *big data*, Labib (1998) já oferecia uma visão estratégica sobre o papel dos Sistemas de Gestão de Manutenção Computadorizados (sigla CMMS no inglês), posicionando-os não como meros arquivos digitais, mas como plataformas dinâmicas para a análise de decisão, capazes de transformar dados brutos de falhas em *insights* para a melhoria contínua dos processos de manutenção.

Aprofundando a análise sobre o valor contido nos dados históricos, Salonen, Bengtsson

e Fridholm (2020) investigam as possibilidades práticas de melhoria da manutenção através da análise de dados de CMMS. Em seu estudo de caso, eles demonstram como a análise aprofundada de ordens de serviço de manutenção corretiva, quando bem descritas, permite à indústria compreender a natureza das quebras e, consequentemente, refinar os programas de manutenção preventiva para aumentar a confiabilidade dos sistemas de produção. O trabalho, no entanto, também expõe um obstáculo crítico e recorrente na indústria: a baixa qualidade dos dados inseridos, seja pela descrição vaga das falhas por parte dos operadores, seja pelo preenchimento incompleto dos relatórios pelos técnicos de manutenção. Esse achado reforça a necessidade de um processo de coleta de dados padronizado e rigoroso, como o proposto neste projeto, para garantir que a análise subsequente seja precisa e relevante.

A transformação de dados estruturados em inteligência utilizável é o domínio da BI. De forma geral, Aishwarya, Lahari e Saheb (2024) e Khan (2024) concordam que as ferramentas de análise de dados são o elo que converte informações brutas em conhecimento, viabilizando uma tomada de decisão mais informada e estratégica, com impactos diretos na otimização de operações e no gerenciamento de riscos. De maneira mais específica, Barbieri, Laserna e Mateus (2024) propõem a aplicação de BI para apoiar a Manutenção Baseada em Evidências (sigla EbM no inglês), uma metodologia que visa alcançar resultados operacionais comparáveis aos de estratégias mais complexas, como a Manutenção Centrada em Confiabilidade, mas com menor investimento inicial. A EbM se baseia na análise de dados concretos e evidências históricas para formular e ajustar os planos de manutenção, tornando-a particularmente adequada para a manutenção não preditiva e para a realidade de pequenas e médias empresas.

Levando a aplicação de BI um passo adiante, Picozzi *et al.* (2024) apresentam um caso prático de desenvolvimento de um *dashboard* interativo em Power BI para o monitoramento de 18 KPIs em um departamento de engenharia clínica. O estudo categoriza os KPIs em áreas como logística, técnica e gerenciamento de equipamentos, e demonstra como a visualização dinâmica dos dados permite uma análise aprofundada da eficiência da manutenção e da obsolescência dos ativos. Este trabalho é um exemplo claro do potencial do Power BI como ferramenta para traduzir dados complexos de CMMS em uma interface visual e intuitiva, que, conforme apontado por Shinde e Shivthare (2024), é fundamental para facilitar a identificação de padrões e melhorar a comunicação entre as equipes técnica e gerencial. Branco (2023), por sua vez, utiliza o Power BI para criar um *dashboard* de supervisão interativo e o conecta a um servidor MySQL.

No contexto específico de correias transportadoras, o monitoramento de condições é um tema central. Trabalhos como o de (BULL *et al.*, 2022), que utiliza sensores para monitorar parâmetros como vibração e temperatura em roletes, e os estudos sobre gêmeos digitais de Pulcini e Modoni (2024),Mafia *et al.* (2024) e Al-Kahwati *et al.* (2022), embora primariamente focados na manutenção preditiva, geram um tipo de dado que é extremamente valioso para qualquer estratégia de manutenção. A análise desses dados de falha e operação é a base para estudos de confiabilidade, como o conduzido por Ahmadi *et al.* (2019). Eles realizam uma análise completa

de Confiabilidade, Disponibilidade e Manutenibilidade em um sistema de correia transportadora, utilizando distribuições estatísticas para modelar o tempo até a falha e o tempo de reparo dos componentes. Com base nessa análise, eles são capazes de determinar intervalos ótimos para a manutenção preventiva, de forma a garantir um nível de confiabilidade predefinido, ilustrando uma aplicação direta de dados históricos na otimização da manutenção.

A escolha da estratégia de manutenção mais apropriada para cada sistema ou componente é, em si, uma decisão estratégica complexa. Shafiee (2015) aborda essa questão ao enquadrar o problema da seleção da estratégia de manutenção como um problema de múltiplos critérios de decisão. A revisão apresentada pelo autor mostra que a decisão entre adotar uma abordagem reativa, preventiva, preditiva ou uma combinação delas deve ser baseada em uma avaliação ponderada de diversos fatores, como criticidade do ativo, custo da falha, custo da intervenção, segurança e impacto ambiental. Essa perspectiva teórica reforça a ideia de que não existe uma única estratégia de manutenção que seja universalmente ótima; em vez disso, a escolha deve ser customizada ao contexto operacional, à criticidade do equipamento e aos objetivos de negócio da organização. A análise de dados históricos, nesse cenário, transcende a mera execução de uma estratégia predefinida, tornando-se o alicerce que informa a própria seleção e o ajuste dinâmico dessa estratégia, estabelecendo um ciclo virtuoso de melhoria contínua e alinhamento estratégico.

Em síntese, a literatura examinada constrói um panorama robusto sobre os pilares da manutenção moderna, porém, frequentemente de forma fragmentada. O projeto a ser desenvolvido aqui, contudo, se diferencia ao não apenas reconhecer, mas ativamente integrar essas múltiplas facetas em uma solução coesa e aplicada, focada especificamente no domínio da manutenção de correias transportadoras. A principal contribuição se apresenta na forma da criação de um fluxo de trabalho completo e unificado: desde a concepção e modelagem de uma base de dados em nuvem, otimizada para a padronização e integridade dos registros de manutenção, até a implementação final de um relatório dinâmico e interativo. Esta solução transforma o dado bruto em uma ferramenta de gestão visual e analítica, capacitando os gestores a analisar tendências históricas, calcular e monitorar indicadores de desempenho chave e identificar os principais ofensores de manutenção de forma intuitiva e ágil. Dessa forma, o projeto preenche uma lacuna prática, construindo a ponte entre a coleta de dados operacionais e a tomada de decisão gerencial estratégica no contexto da manutenção não preditiva, um campo que, apesar de sua vasta aplicação, carece de soluções integradas e acessíveis como a aqui desenvolvida.

3 METODOLOGIA

Este capítulo descreverá a metodologia detalhada para a execução do projeto de informatização do registro e análise de dados de manutenção de correias transportadoras, estruturando o fluxo de dados desde a sua concepção inicial até a visualização analítica final.

A implementação deste projeto buscará, no *back-end*, criar uma base de dados robusta e de modificação prática, e no *front-end*, gerar um relatório dinâmico unificado que seja intuitivo e facilmente atualizável, otimizando a velocidade de identificação de falhas e de padrões.

3.1 Concepção e Modelagem da Base com Dados Fictícios

Para simular um ambiente corporativo e validar o sistema, serão elaboradas quatro tabelas principais: Reportes, Sensores, Processos e Ordens de Serviço (OS), buscando demonstrar como uma informatização de análises e registros poderia ser executada de forma aplicável. A ferramenta primária para a estruturação inicial destas tabelas será o Microsoft Excel, devido à sua ágil e relativa simplicidade na edição de registros — essencial para essa estruturação tendo em vista as diversas correções, formatações e mesclagens necessárias nas tabelas para que elas obtenham um formato semelhante ao encontrado em aplicações reais. Sendo uma ferramenta consolidada no mercado, o Excel possui uma grande biblioteca de funções e atalhos de preenchimento de linha, além de fácil integração com o Google Planilhas, softwares de banco de dados e a opção de gravação em formato de Valores Separados por Vírgulas (*Comma-separated Values* no inglês — CSV), o que posteriormente poderá facilitar sua importação para Sistemas Gerenciadores de Bancos de Dados (SGBD).

Entretanto, para garantir que a base possua a variabilidade e um volume de dados necessário para uma simulação fiel (como um *data lake* corporativo), serão utilizados ferramentas de Inteligência Artificial consolidadas no mercado parar gerar registros aleatórios — especificamente, o ChatGPT e o Deepseek. Isso se demonstrou necessário pelo fato do acesso a um banco de dados real não ter sido possível para este projeto, que resultaria na exposição de dados sensíveis. Ademais, o uso de duas IAs diferentes se demonstra necessário devido ao limite de uso diário estabelecido por cada um dos serviços, utilizadas sempre na versão gratuita e de forma alternada para evitar o atingimento desse limite. Uma ordem de utilização foi definida de acordo com grau de fidelidade de resposta observada de forma empírica por uso prévio e complexidade da solução pedida, priorizando o uso do ChatGPT para solicitações mais complexas e o Deepseek para aquelas mais simples, visto que o último é a ferramenta mais recente entre as duas.

A tabela de Reportes funcionará como a tabela central do projeto, armazenando fichas de diagnóstico técnico. Serão gerados 300 registros aleatórios cobrindo o período de 2024 a 2025, utilizando funções como ALEATÓRIOENTRE e lógicas de programação para definir o status de cada ficha (Pendente ou Realizada). Para determinar campos não-calculados como os de "Técnica" e "Local", serão desenvolvidos e executados *scripts* em Python garantindo que o

número de registros obedeça a contadores. Além disso, a granularidade dos componentes será alcançada através de funções de pesquisa (PROCV) vinculadas a tabelas-fato de equipamentos, como motores, redutores e rolamentos. O número de 300 registros foi escolhido pois, ao planejar a execução deste projeto, foi-se observado que mediante todas as tarefas necessárias para reproduzir os campos das tabelas desse projeto, um número muito maior não permitiria sua realização prática de forma artificial, visto que ferramentas de IA não seriam capazes de demonstrar todas as particularidades contidas nesse projeto, uma vez que essas particularidades são criadas apenas pelo fator humano nos processos. Para isso, uma curação de linha por linha seria exigida, possibilitada apenas pelo preenchimento orgânico e temporal, ao longo de anos, da base de dados aqui estipulada.

A tabela de Sensores simulará o monitoramento de 40 componentes instalados em campo, buscando retratar um cenário em que componentes classificados como críticos — por quaisquer motivos sejam, como por quebra constante ou essencialidade numa planta — tenham a interrupção do seu funcionamento reconhecida de forma mais veloz, enquanto as tabelas de Processos e Ordens de Serviço estabelecerão o vínculo entre o diagnóstico e a execução da manutenção. Um diferencial metodológico será o uso de tabelas dimensão — tabelas que possuem informações em linhas únicas que não se repetem — para a construção de campos de status das tabelas de Processos e OSs. Outra operação relevante será a expansão da tabela de OS no Power Query nativo do Excel (também contido no Power BI), onde a função *List.Numbers* será aplicada para multiplicar registros únicos de OS em múltiplas etapas de operação, permitindo uma análise detalhada da performance dos serviços.

Posteriormente, essas tabelas seriam hospedadas parte via MySQL, parte via Google Planilhas, de modo a tornar o projeto fidedigno com o que foi proposto, finalmente sendo transformadas em consultas no software Power BI.

3.2 Arquitetura de Hospedagem e Interfaces de Inserção

Após a estruturação das tabelas de acordo com a idealização mencionada na subseção anterior, a metodologia contemplará uma hospedagem híbrida em nuvem para refletir a diversidade tecnológica de grandes bancos de dados. A tabela de Reportes será migrada para o Google Planilhas, onde ficará integrada ao Google Formulários, se apresentando como os softwares de escolha para abrigar esses dados por apresentarem rapidez para a inserção de dados através dos formulários e custo zero de implementação. Esta configuração permitirá que analistas e operadores insiram novos diagnósticos de campo através de uma interface gráfica de usuário (*Graphic User Interface* no inglês — GUI) amigável.

No contexto atual, existem algumas opções de SGBDs de SQL, como o MySQL, PostgreSQL, SQL Server e Oracle. A escolha para a aplicação neste projeto se apresenta como o MySQL, devido à grande biblioteca online para aprendizado e número de ferramentas integradas

para com o mesmo. O XAMPP Control Panel, por sua vez, é um pacote gratuito e de código aberto que instala um ambiente de servidor *web* local, permitindo que desenvolvedores criem e testem sites e aplicações web dinâmicas como se estivessem em um servidor de produção, sem a necessidade de *internet*. Ele é necessário para que o MySQL Workbench, um cliente gráfico de gerenciamento de servidores, seja utilizado para o gerenciamento da base deste projeto, permitindo modelar, desenvolver e administrar o bancos de dados.

As tabelas de Sensores, Processos e OSs serão hospedadas em um ambiente SQL. Para isso, será configurado um servidor local através do XAMPP Control Panel, com gerenciamento direto via MySQL Workbench. A definição dos esquemas de banco de dados ocorrerá no editor Visual Studio Code (VSC), que possui funcionalidades de organização e indentação de código, onde os tipos de dados serão especificados (como VARCHAR, DECIMAL e DATE) para assegurar a integridade das informações e a correta formatação de datas e números decimais padrão SQL. A segmentação da hospedagem em múltiplas plataformas realizada aqui é justificada pela rapidez na Prova de Conceito, sendo possível verificar a usabilidade técnica dos métodos escolhidos de forma mais prática.

Então, se tornava necessária a elaboração de uma forma de novos dados serem adicionados à essas tabelas hospedadas em nuvem. A solução idealizada foi a criação de formulários executáveis programados em linguagem Python. Essa linguagem foi escolhida devido à sua rápida curva de aprendizagem, alta modularidade e simplicidade. Para isso, foram necessárias três instalações:

- Do pacote da linguagem em questão no sistema, juntamente com sua extensão respectiva para o VSC, encontrada internamente no aplicativo;
- Da biblioteca tkinter, através do VSC. Essa é a biblioteca padrão do Python para criar GUIs, permitindo desenvolver rapidamente janelas, botões, caixas de texto e outros elementos visuais de forma fácil e eficiente, já incluída no Python e não exigindo instalação adicional;
- De um conector MySQL para Python via terminal (*prompt* de comando do Windows).

Ademais, a instalação desses formulários gráficos permitirão a inserção de novos dados na base de dados sem a necessidade de manipulação direta de códigos SQL, utilizando *placeholders* de variáveis para enviar os *inputs* dos usuários simultaneamente ao servidor local.

A hospedagem nas plataformas escolhidas é justificada pela Prova de Conceito, permitindo verificar a usabilidade técnica dos métodos escolhidos de forma mais ágil, favorecer a reprodutibilidade e reduzir a complexidade inerente à implantação direta em ambientes definitivos, que normalmente demandariam maiores requisitos de governança, segurança e infraestrutura. Dessa forma, torna-se possível realizar ciclos rápidos de teste, ajuste e validação, possibilitando a identificação precoce de limitações arquiteturais e oportunidades de melhoria antes de um eventual aplicação definitiva.

3.3 Processamento de Dados e Visualização Analítica (*Front-end*)

A fase final da metodologia focará na transformação de dados brutos em inteligência de negócio através do Microsoft Power BI. O processo de ETL (Extração, Transformação e Carregamento) será centralizado no Power Query, onde as consultas provenientes do Google e do MySQL serão tratadas, limpas e formatadas. Um passo crítico será a criação do "Sistema Consolidado", uma consulta resultante da mesclagem das tabelas de Processos e OSs, eliminando redundâncias operacionais antes do carregamento para o modelo de dados.

Dentro do Power BI, serão estabelecidos relacionamentos do tipo muitos-para-muitos entre as tabelas — agora denominadas consultas, de Sensores, Reportes e o Sistema Consolidado, utilizando as chaves de "Ficha", "Processo" e "OS". Para a análise de performance, serão implementadas medidas personalizadas em linguagem DAX, possibilitando o cálculo de indicadores de *lead-time* e taxas de consistência entre as consultas.

O *front-end* do relatório será desenhado para oferecer um controle analítico total, incluindo rankings, exibição de falhas, tabelas de consulta e indicadores de status dos sensores. O uso de dicas de ferramenta para melhor clareza do relatório e visualização aprimorada de dados e a filtragem cruzada entre os visuais garantirá que o usuário possa identificar rapidamente gargalos operacionais e priorizar esforços de manutenção, validando o objetivo geral de informatizar e otimizar o processo de manutenção deste projeto.

4 RESULTADOS

Para uma análise dos resultados gerados no desenvolvimento da solução para a digitalização proposta, será necessário dividir esta seção nos seguintes subtópicos:

1. **Concepção da base de dados:** abordará a geração da base com dados fictícios através das ferramentas selecionadas.
2. **Formato de hospedagem:** realização da hospedagem dos dados contidos nas tabelas, juntamente da forma de manutenção das mesmas.
3. ***Front-end* em Microsoft Power BI:** possibilidades e análises geradas.

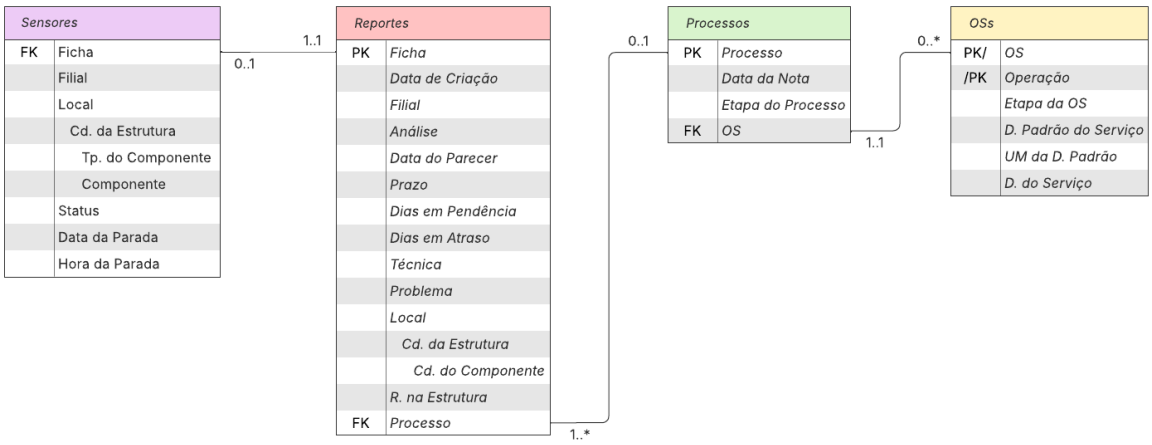
4.1 Concepção da Base de Dados

Baseado numa análise de cenário real, foram elaboradas quatro tabelas com dados fictícios baseadas em um ambiente real, buscando preservar a diversidade de dados presente num ambiente corporativo e os desafios a serem solucionados por um analista de dados. Estas foram denominadas como:

- Reportes;
- Sensores;
- Processos;
- Ordens de Serviço (OS).

Essas tabelas foram montadas utilizando o Microsoft Excel (versão 2019), devido à sua relativa simplicidade e versatilidade para edição de registros. Entretanto, essas tabelas não foram montadas exclusivamente através de fórmulas do software, tendo sido complementada através do uso de IAs generativas mencionadas na seção 3. Na figura 2, é possível observar um diagrama entidade-relacionamento exprimindo a organização definida entre as tabelas do projeto, em que cada uma delas é conectada através das chaves primárias (PK) e estrangeiras (FK) definidas.

Figura 2 – Diagrama entidade-relacionamento das tabelas do projeto.



Fonte: Elaborado pelo autor, 2026.

Com essa estrutura, é possível detalhar mais profundamente cada tabela do projeto, de acordo com as subseções a serem vistas em sequência.

4.1.1 *Tabela de Reportes*

A tabela de Reportes foi criada no intuito de ser a tabela central deste projeto, sendo aquela a ser alimentada pelos operadores em campo após cada rodada de aferição realizada. Ela armazena as Fichas de Diagnóstico relativas às CTs — uma espécie de registro dos resultados das inspeções feita em campo nos componentes relacionados à ela, na positiva de identificação de defeito. Cada linha corresponde a uma ficha, e as colunas, a seus campos, somando um total de 300 nessa amostra. Cada ficha contém dados de localidade exata dentro de todas as filiais de uma empresa, datas e análises de *lead-time* de solucionamento, informações de diagnóstico e vínculo com outras tabelas. Todas essas informações são trazidas num formato que permite seu registro histórico na tabela. Na tabela 1 é possível observar um resumo de todos os campos encontrados nessa tabela.

Várias colunas necessitarão do uso das funções ALEATÓRIO() e ALEATÓRIOENTRE para sua construção. Por conta disso, é necessário se atentar à uma peculiaridade do Microsoft Excel, que é a atualização dessas fórmulas desencadeada por qualquer alteração na pasta de trabalho. O problema disso é que muitos desses dados serão utilizados para a construção das outras tabelas, então, essa atualização automática foi desativada para essas construções. A seguir, é possível observar o detalhamento da geração de cada coluna/campo da tabela.

Tabela 1 – Campos da tabela de Reportes.

Campo	Tipo de dado	Resumo
Ficha	Número	Principal coluna da tabela, funcionando como campo-chave central do projeto. Composta de registros únicos e incrementais.
Data de Criação	Data	Data de criação da ficha na tabela.
Filial	Texto	Filial respectiva à ficha criada. Representa a maior granulação de localização.
Análise	Texto	Pode ser "pendente" ou "realizada". Resultado por extenso do preenchimento do campo "Data do Parecer".
Data do Parecer	Data	Data de verificação do problema identificado e criação de um plano de ação ou da própria resolução do problema.
Prazo	Número	*Deadline*, em semanas, para a verificação do problema.
Dias em Pendência	Número	Delta entre a data de criação da ficha e a data do parecer.
Dias em Atraso	Número	Delta entre o prazo e os dias em pendência. Representa a extrapolação do prazo.
Técnica	Texto	Técnica fictícia utilizada para a identificar um problema na CT, baseada em termos aleatórios.
Problema	Texto	Descoberta resultante do uso da técnica, manifestada como um código.
Local	Texto	Áreas dentro de cada filial que podem conter uma ou mais CTs, representadas por uma letra (A–C). É o primeiro campo em direção a uma granulação menor.
Código da Estrutura	Texto	Concatenação da letra de "Local" com cinco outras letras (A–E), gerando até 15 estruturas possíveis em cada filial. Cada CT é uma estrutura.
Código do Componente	Texto	Concatenação do campo "Código da Estrutura" com algarismos de 1–8, resultando em até 120 combinações diferentes. Representa a menor granulação de local.
Recorrência na Estrutura	Número	Número de aparições de problemas em um mesmo código de estrutura.
Processo	Número	Campo-chave de seis algarismos para ligação com a tabela de "Processos", podendo conter registros repetidos.

Ficha — os números de 1-300 foram gerados aleatoriamente. Para isso, 300 números sequenciais foram gerados através das fórmulas LIN() e ALEATÓRIO(), juntamente com manipulações de colunas na planilha.

Data de Criação — gerada através do seguinte comando no Deepseek: "gere 300 datas aleatórias no período (inclusivo) de 01/01/2024 à 30/09/2025". Entretanto, dois problemas foram

identificados:

- A geração trouxe valores fora do período estipulado, exigindo que as linhas fossem filtradas manualmente via Excel para manter apenas aquelas que estivessem dentro do período solicitado, sendo copiadas para uma célula ao lado para que a coluna anterior pudesse ser removida;
- A geração extrapolou as 300 linhas, tendo que ser limitada manualmente no Excel deletando as linhas que haviam ultrapassado a quantidade solicitada.

Ademais, algumas datas tiveram seu mês alterado manualmente para novembro e dezembro de 2025 devido ao delta de tempo entre o início e fim da construção da parte técnica do projeto, complementando a variabilidade mas de forma reduzida.

Filial — para esse campo, o seguinte comando foi utilizado via Deepseek: "gere 300 linhas aleatórias com os seguintes termos: Alpha; Bravo; Charlie; Delta; Ecko". Estas seriam nomes fictícios de filiais de uma empresa. A geração, contudo, extrapolou o limite definido de linhas, sendo necessário a exclusão manual via Excel das linhas extras.

Análise — gerada através do comando no Deepseek: "gere 300 linhas aleatórias com os termos "Pendente"e ”Realizada”". Novamente, a geração não foi do limite estipulado de linhas, mas sim de um número inferior. Para isso, linhas de uma altura aleatória do código foram copiadas e coladas para completar as células faltantes, visto que, de acordo com a Lei dos Grandes Números, o impacto dessa ação seria desprezível.

Data do Parecer — as 300 datas geradas e modificadas contidas no campo "Data de Criação" foram salvas num arquivo CSV e anexadas ao Deepseek, juntamente do seguinte comando: "agora, se baseando nas datas contidas no arquivo CSV em anexo, estipule datas posteriores a elas que variem de 1-90 dias". A geração dessas datas mais uma vez extrapolou o número estipulado, mas desta vez, tendo de ser interrompida em chat manualmente. No momento dessa pausa, mais de 1000 linhas haviam sido geradas, contudo, correspondentes às suas datas respectivas do campo "Data de Criação".

Dentro do Excel, uma lógica IF foi criada: caso o campo "Análise" contivesse "Realizada" para a célula atual, as novas datas, contidas em uma outra coluna próxima, seriam exibidas naquela coluna em específico; caso contrário, a célula ficaria vazia. Essa lógica foi necessária para que apenas aquelas fichas que possuíssem sua análise realizada, contivessem uma data de parecer.

Ademais, uma nova diretiva foi inserida posterior ao comando utilizado para gerar essa coluna: "um número de linhas aleatório está sendo gerado. Implemente um contador no seu código de geração!", no intuito de corrigir o problema identificado na próxima solicitação.

Prazo — o seguinte comando foi *inputado* no Deepseek: "agora, gere 300 linhas com o X variando 1-9 semanas de acordo com o padrão abaixo entre aspas: ”X semana(s)”". As linhas

resultantes seriam um prazo aleatório estabelecido para que a ficha tivesse sua verificação (campo "Análise") realizada por um analista.

O número de linhas, contudo, permaneceu aleatório, com a diretiva anteriormente estabelecida sendo ignorada pela IA.

Dias em Pendência — essa é uma coluna calculada. Sua lógica foi construída da seguinte forma, utilizando a função IF: se o campo de prazo de intervenção respectivo estiver vazio, uma subtração da célula respectiva do campo "Data de Criação" é feita do dia atual; caso contrário, essa subtração é realizada do campo "Data do Parecer".

Dias em Atraso — por tratar-se também de uma coluna calculada, esta depende da coluna "Dias em Pendência" e é definida a partir da seguinte lógica condicional (IF): se a subtração dos dias de "Prazo" — obtidos por meio da função ESQUERDA, utilizada para extrair o dia da semana, multiplicado por 7 e acrescido de 1 dia para considerar a subtração do dia atual — do valor correspondente em "Dias em Pendência" resultar em um valor menor que zero, o número de dias em atraso é definido como zero; caso contrário, o resultado do cálculo descrito até agora é mantido. Isso ocorre pois, diferentemente do campo "Dias em Pendência", este campo contabiliza apenas os dias contados após o fim do prazo para a solução do problema representado pela ficha atual.

Técnica — para essa geração, uma abordagem diferente foi utilizada, juntamente com o uso do ChatGPT ao invés do Deepseek. o seguinte comando foi inserido: "gere 300 linhas aleatoriamente, implementando um contador ao seu código que encerre sua execução quando este limite for atingido, em que cada linha contenha um dos cinco termos abaixo: Gauss; Stockholm; Taylor; Geiger; Ford". Esses foram termos fictícios baseados em nomes de personalidades relevantes na história no intuito de preencher o registro.

Este comando resultou em um código em Python (ver apêndice A.1.1) que, em sequência, teve sua execução solicitada dentro do ChatGPT juntamente da sua exportação para um arquivo CSV, resultando no número correto de linhas.

Problema — esse campo cria um código que relaciona a técnica fictícia utilizada para identificar o defeito com um dígito de problema fictício, elaborado através da geração aleatória de números inteiros entre um a cinco dentro do Excel, sendo concatenados com as duas primeiras letras de cada técnica, fazendo uso da fórmula ALEATÓRIOENTRE e de fórmulas de modificação e criação de texto.

Local — aqui, o seguinte comando foi inserido no ChatGPT: "gere exatamente 300 linhas aleatórias variando entre as letras A-C. Se necessário, crie um código em Python para a solução, contendo um contador para gerar exatamente este número de linhas, e imprima o resultado para mim". Com isso, um código em Python foi gerado (ver apêndice A.1.2), que por sua vez, teve sua execução solicitada dentro do ChatGPT juntamente da sua exportação para um arquivo CSV, totalizando o número solicitado de linhas. Cada letra se refere a uma área genérica que existira

dentro das filiais da empresa.

Código da Estrutura — seguindo o mesmo procedimento para a geração do campo "Local", as letras agora variaram de A-E. Isso representou um salto para uma menor granulação de dados — ao invés de três alternativas, aqui foram gerados cinco diferentes. A ideia por trás desse campo é que cada local contenha cinco estruturas diferentes que, concatenadas em Excel via função CONCAT, formam o código estipulado para esse campo.

Código do Componente — esse campo representa a menor granulação no que tange a representação de localidade dessa tabela. Para sua elaboração, o campo "Código da Estrutura" foi concatenado (função CONCAT) com um número aleatório (função ALEATÓRIOENTRE) gerado entre 1-8 — esses sendo códigos que representam uma série de componentes encontrados em correias transportadoras.

Recorrência na Estrutura — para esse campo, a função CONT.SES foi utilizada, em que todas as aparições de um mesmo código de estrutura, dentro da mesma filial, foram contadas. Essa coluna tem por função criar uma espécie de ranque das estruturas que mas apresentam problemas e falhas que, dependendo da filial em questão, pode ocorrer por diversas intempéries.

Processo — representa uma chave que relaciona essa tabela com as tabelas de Processos e Ordens de Serviço. Para isso, a mesma solução utilizada para a criação do campo de ficha foi utilizada inicialmente, porém, deixando de utilizar a função LIN() e utilizando a funcionalidade de auto-preenchimento do Excel para ir de 300000 à 309999, criando um código de seis dígitos. O prefixo 30 por si próprio indica processos que se referem à fichas relativas apenas às CTs, considerando que existam fichas de diagnóstico referentes a outros assuntos e problemas dentro de uma companhia de grande porte.

Por fim, alguns poucos números de processo foram alterados manualmente para que possuam múltiplas ocorrências na tabela, visto que um processo pode conter mais de uma ficha (mais a ser visto na seção 4.1.3).

4.1.2 *Tabela de Sensores*

A tabela de Sensores consta com 40 registros de funcionamento de sensores, buscando retratar um cenário em que componentes considerados como críticos — por quaisquer motivos sejam, como por quebra constante ou relevância para o funcionamento geral da planta — tenham a interrupção do seu funcionamento reconhecida de forma mais veloz. Para sua criação, fichas aleatórias foram selecionadas da tabela de Sensores e incluídas em uma nova planilha. Juntamente do campo Ficha, os campos de Filial, Local, Código da Estrutura e Data do Parecer foram trazidos dessas fichas selecionadas aleatoriamente. Diversas alterações foram realizadas na nova tabela e novas colunas foram criadas, entretanto, buscando demonstrar como um registro de funcionamento de sensores funcionaria.

Tabela 2 – Novos campos da tabela de Sensores.

Campo	Tipo de dado	Resumo
Tipo de Componente	Número	Contém exclusivamente dígitos de 1–8, diferentemente do campo “Código do Componente”. Refere-se apenas ao código numérico que representa o componente da CT.
Componente	Texto	Denominação por extenso do código em “Tipo de Componente”.
Status	Texto	Contém apenas "parado"ou "normal", representando o estado de funcionamento da CT.
Hora da Parada	Tempo	Hora de interrupção do funcionamento pelo defeito identificado.

Tipo de Componente — resultante da extração do último dígito referente ao código do componente em questão na CT, utilizando a função RIGHT.

Componente — uma diferente lógica foi aplicada para a construção deste campo. Primeiramente, uma tabela dimensão foi criada (figura 3), relacionando uma série de componentes que uma correia transportadora contém com um código em formato de dígito. Com essa tabela criada numa planilha auxiliar dentro da pasta de trabalho do Excel, a função PROCV foi utilizada de forma a retornar o nome do equipamento relacionado ao dígito contido no campo "Tipo de Componente".

Figura 3 – Tabela dimensão de Tipos de Equipamento.

Tipos de Equipamento	
Código	Equipamento
1	Estrutura
2	Esteira
3	Rolete
4	Mancal
5	Truque
6	Rolamento
7	Motor
8	Redutor

Fonte: Elaborado pelo autor, a partir do Microsoft Excel (2026).

Status — para esse campo, a seguinte lógica IF foi construída: se o campo "Data do Parecer" está preenchido, então o status é "Normal"; caso contrário, "Parado"; que resultou em cerca de 19 campos com o termo "Parado". Essa lógica foi realizada pelo fato de o próprio preenchimento desse campo indicar se a análise (campo "Análise" da tabela de Reportes) da ficha foi realizada, que representaria a correção do problema no componente da CT. Com isso, não é necessário utilizar uma função mais complexa, como a PROCV. Os termos "Normal" e

"Parado" representam se o componente em questão está funcionando normalmente ou paralisado, respectivamente.

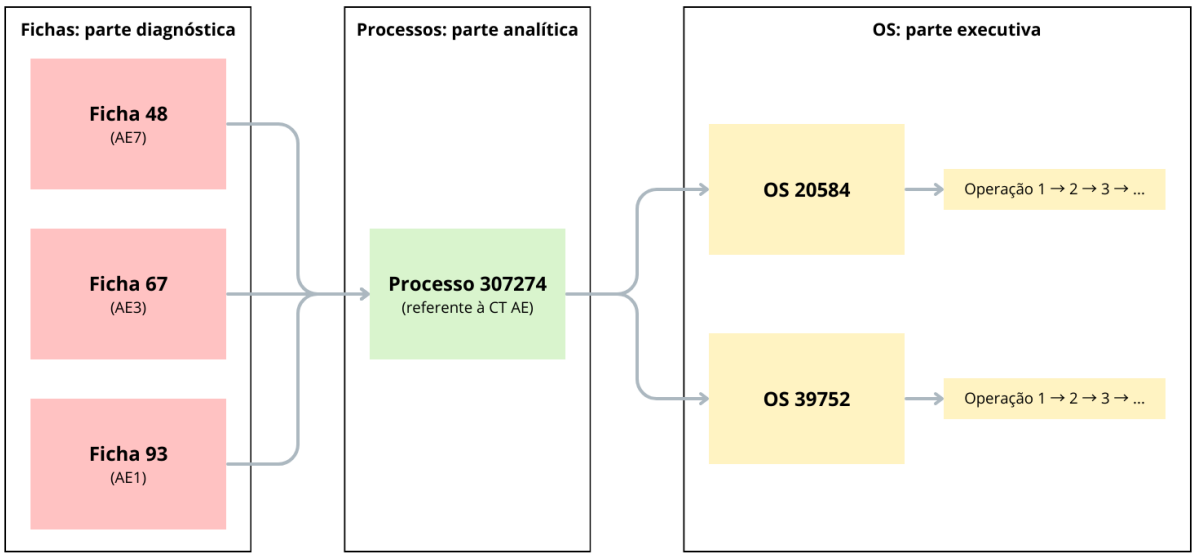
Hora da Parada — para a geração desse campo, 19 horários aleatórios respectivos ao número de fichas com status "Parado" foram gerados através do seguinte comando em ChatGPT: "me gere 19 horários com precisão de minutos aleatórios, de 00:00 horas até 23:59 horas". O intuito dessa geração é fornecer com certa precisão qual foi o momento de identificação do problema no componente, visto que fatores como temperatura, chuva e outros eventos pontuais podem ter influenciado ou levado diretamente àquele componente a apresentar falha, em que cerca de metade da base

Como últimas modificações, as fichas que possuíam status "Normal" tiveram seu número eliminado da tabela, pois CTs em pleno funcionamento correto não possuem uma ficha de diagnóstico de problema associada. Ademais, o campo "Data do Parecer" foi renomeado para "Data de Parada"— mais adequado para a tabela em questão.

4.1.3 *Tabela de Processos*

A tabela de Processos é essencial para o vínculo ficha-OS. A distinção entre ficha e processo se dá pelo fato de um processo ser um registro que pode conter diversos outros grupos de dados associados a ele — neste caso, múltiplas fichas ou múltiplas OSs (ver exemplo na figura 4). Isso ocorre, por exemplo, devido à razões financeiras, em que é mais economicamente viável contratar uma empreiteira para realizar uma série de serviços relacionados de uma única vez, ou por um grande número de problemas semelhantes ou idênticos que podem ser executados pela mesma empreiteira.

Figura 4 – Diagrama de fluxo de processo.



Fonte: Elaborado pelo autor, 2026.

Uma breve descrição de cada campo dessa tabela é exibida na tabela 3. De forma subsequente, a construção dessa tabela é descrita, apresentando três novos campos e possuindo um campo já conhecido previamente.

Tabela 3 – Campos da tabela de Processos.

Campo	Tipo de dado	Resumo
Processo	Número	Modificação do campo "Processo" da tabela de Reportes, buscando mesclar os registros da outra tabela no meio de registros não pertinentes ao projeto.
Data da Nota	Data	Datas em "Data de Criação"+1 dia para números de processo já existentes, e números aleatórios para os novos, justificado pela imagem 4.
Etapa do Processo	Texto	Alternando entre "analisando", "aguardando OS" e "encerrado", representa status próprio do Processo.
OS	Número	Chave referente à parte executiva da manutenção, também utilizada na tabela "Ordens de Serviço".

Processo — existem algumas diferenças deste campo para o encontrado na tabela de Reportes. Ainda na tabela Reportes, das 300 fichas criadas, cerca de 50 tiveram seus processos deletados. Essa ação foi realizada no intuito de simular o fato de que nem todas as fichas, inseridas manualmente pelos operadores em campo, possuirão um número de processo vinculado já de início, visto que preenchimentos parciais são frequentes no cenário real de uma companhia, seja por falta de toda a informação no momento, seja por desatenção. Então, esses números de processo foram copiados para este campo, sendo submetidos a outra modificação similar: cerca de 30 números de processo foram novamente deletados, mas somente nesta tabela. Essa nova exclusão busca representar uma lacuna de preenchimento entre as bases, para, novamente, casos de erro humano ou falta de atualização entre as bases.

Além dos processos restantes, totalizando cerca de 220, mais 1000 linhas foram geradas aleatoriamente e adicionadas abaixo dos números de processos já existentes. Essa inserção foi realizada de modo a tornar a base de dados mais complexa, buscando simular de forma mais fiel possível um *data lake*, comumente encontrado em empresas de grande porte, em que mesmo dentro de um tipo número de processo, podem existir diversos tipos de Fichas de Diagnóstico diferentes sobre diversas áreas de manutenção, como por exemplo, tratando-se do diagnóstico de problemas em veículos e instalações utilizando do mesmo prefixo 30. Para que os números já existentes não interfiram com os novos a serem inseridos na planilha, essa adição de registros foi realizada através da cópia de mais números da planilha auxiliar criada para a primeira geração dessa coluna na tabela Reportes (ver seção 4.1.1).

Data do Processo — representa um campo derivado do "Data de Criação" da tabela de Reportes, com um delta de tempo de um dia adicionado, visto que um processo pode ser criado

apenas após a condição de existência de uma ficha. Isso resultou nas, aproximadamente, 220 datas iniciais, e para os outros números de processo sem correspondência na tabela de Reportes, novas datas aleatórias dentro do mesmo período do campo "Data de Criação" (ver seção 4.1.1 foram geradas via ChatGPT.

Etapa do Processo — para essa geração, mais uma tabela dimensão foi criada numa planilha auxiliar no Excel (figura 5), buscando tornar a geração aleatória nativa. Então, a função PROCV foi utilizada da seguinte forma: procure o valor aleatório gerado de 1 à 3 (função ALEATÓRIOENTRE) na matriz da tabela auxiliar, me retornando a 2ª coluna de índice, em caso de correspondência exata. Com isso, o status do processo correspondente é retornado da avaliação.

Figura 5 – Tabela dimensão de Tipos de Status do Processo.

Tipos de Status do Processo	
Código	Status
1	Analisando
2	Aguardando OS
3	Encerrado

Fonte: Microsoft Excel, 2026.

OS — essa é uma chave que relacionará a tabela atual com a tabela de Ordens de Serviço. Ela é relativa, como o título do campo sugere, às execuções de serviço referentes a um processo. Diferentemente da chave "Processo", cada código de 5 dígitos aqui é único, não podendo se repetir. Isso se justifica pelo fato de cada serviço ser único, por consequência, ser planejado de acordo com àquela solicitação. Para sua geração, o método utilizado para o campo de "Processo" foi utilizado novamente, com apenas o intervalo de algarismos sendo modificado.

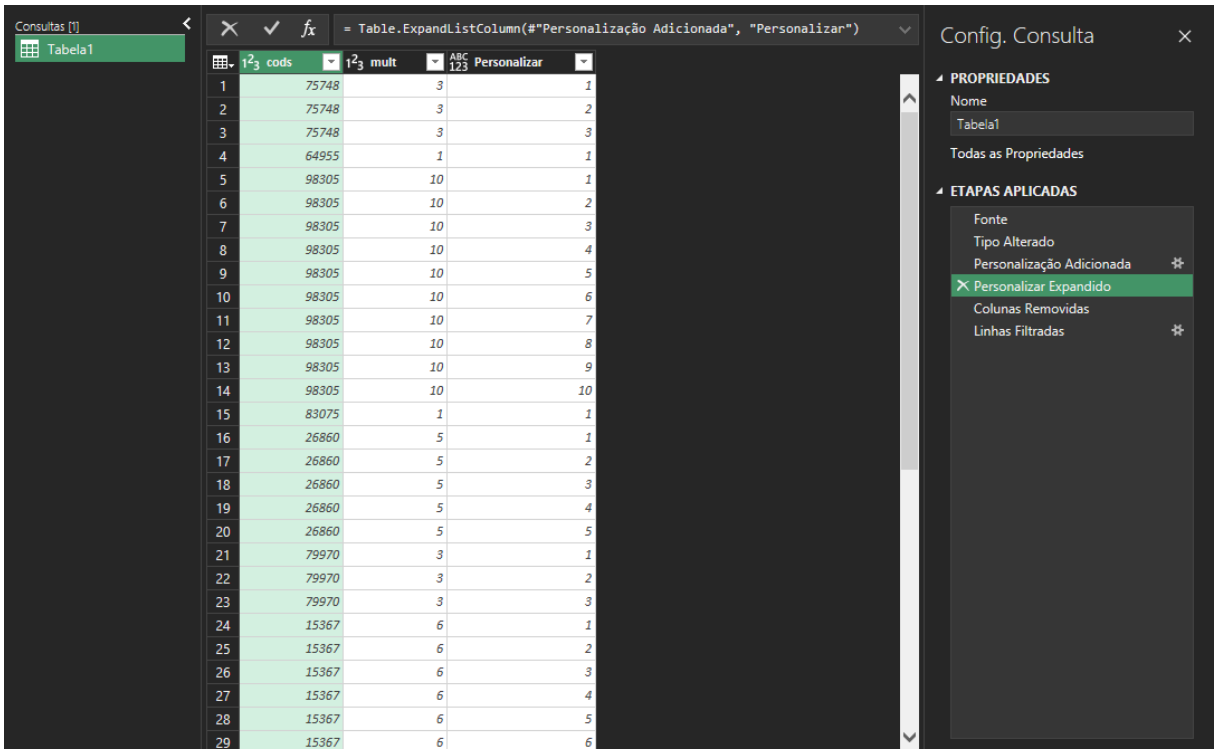
4.1.4 *Tabela de Ordens de Serviço (OS)*

A tabela de Ordens de Serviço é a última tabela contendo dados fictícios criada para este projeto, se destacando por ser a que contém o maior número de registros. Isso acontece pois a lógica é que, por fins de organização, uma OS pode conter múltiplas etapas de execução de um serviço, e que todas essas etapas devem ser registradas, contabilizadas e precificadas mediante as operações necessárias.

Para que as OSs passem a possuir múltiplos registros internos representados nessa tabela, todos os números de OSs foram copiados para uma nova planilha auxiliar, e então, através do uso da função ALEATÓRIOENTRE, números no intervalo de 1 à 10 foram gerados para cada linha — representando uma variação estipulada de operações por OS. Então, no editor do Power Query interno ao Excel, uma série de etapas foi realizada para transformar os registros únicos de OS com um número aleatório de operações internas, em n linhas de mesma OS com n =

essas operações internas (ver apêndice A.1.3). Na figura 6, é possível observar o resultado da multiplicação mencionada.

Figura 6 – Demonstração da transformação realizada em Power Query no Excel.



	cods	mult	Personalizar
1	75748	3	1
2	75748	3	2
3	75748	3	3
4	64955	1	1
5	98305	10	1
6	98305	10	2
7	98305	10	3
8	98305	10	4
9	98305	10	5
10	98305	10	6
11	98305	10	7
12	98305	10	8
13	98305	10	9
14	98305	10	10
15	83075	1	1
16	26860	5	1
17	26860	5	2
18	26860	5	3
19	26860	5	4
20	26860	5	5
21	79970	3	1
22	79970	3	2
23	79970	3	3
24	15367	6	1
25	15367	6	2
26	15367	6	3
27	15367	6	4
28	15367	6	5
29	15367	6	6

Fonte: Elaborado pelo autor, a partir do Microsoft Power Query do Excel (2026).

Como ponto principal desse código, a função *List.Numbers* foi aplicada à coluna de operações através de uma fórmula personalizada, existente apenas no Power Query, que fez com que o número de OS fosse multiplicado pelo número de operações. Com essa tabela modificada, exibida em uma nova planilha no Excel, dois campos da tabela de OSs já haviam sido criados: os campos de "**OS**" e de "**Operação**". A partir deste momento, era possível criar os campos finais para a consolidação da base de dados deste projeto. Na tabela 4, é possível observar um resumo dos campos contidos nessa tabela.

Tabela 4 – Campos da tabela de OSs.

Campo	Tipo de dado	Resumo
OS	Número	Versão expandida do campo na base de "Processos" de acordo com o número de operações de cada OS.
Operação	Número	Numerado de forma sequencial em relação ao número de operações por OS. Possibilitada a demonstração dos dados contidos nas próximas colunas.
Etapa da OS	Texto	Semelhante ao "Etapa do Processo" possuindo três termos únicos, mas voltado para a execução individual de cada operação.
Duração Padrão do Serviço	Decimal	Duração aleatória padrão estabelecida para cada operação.
Unidade de Medida da Duração Padrão	Texto	Pode ser horas ou dias de serviço, contabilizando apenas tempo em turno.
Duração do Serviço	Decimal	Duração aleatória requerida de fato para cada operação.

Etapa da OS — para esse campo, a mesma lógica utilizada para a criação do campo "Etapa do Processo" foi utilizada (ver seção 4.1.3); contudo, as denominações foram diferentes, devido à natureza diferente do registro.

Figura 7 – Tabela dimensão de Tipos de Status da OS.

Tipos de Status da OS	
Código	Status
1	Planejando
2	Agendado
3	Executado

Fonte: Elaborado pelo autor, a partir do Microsoft Excel (2026).

O campo de "**Duração Padrão do Serviço**" representa uma métrica comumente utilizada no setor terciário, possibilitando analisar a performance de serviços prestados. Para sua concepção, um número aleatório foi gerado no intervalo de 1-24 (função ALEATÓRIOENTRE) em uma coluna auxiliar, limitando o tempo máximo de horas que uma operação pode conter dentro de uma OS neste projeto. Numa nova coluna, denominada "**Unidade de Medida (UM) da Duração Padrão**", a seguinte lógica IF foi implementada: se o número contido em "Duração Padrão do Serviço" for maior ou igual ao número 12, o período é dado em dias; caso contrário, é dado em horas. Esse campo foi criado devido ao turno de 12 horas para serviços escolhido para este

projeto. Por fim, o valor contido nas linhas da coluna auxiliar era dividido por 12 caso o campo "UM da Duração Padrão" correspondente fosse "dias", ou permanecia o mesmo em caso negativo, através de outra lógica IF, resultando no campo mencionado no início deste parágrafo.

Duração do Serviço — sendo o último campo desta tabela, ela representa o tempo de execução que os prestadores de serviço levaram para a execução da operação em questão. Sua lógica é observada através da fórmula localizada no apêndice A.1.4 que, traduzida, multiplica o valor correspondente do campo "Duração Padrão do Serviço" por um valor aleatório entre 66% e 150%, buscando retratar uma variabilidade plausível de tempo de execução em um cenário real.

4.2 Formato de Hospedagem

Com a base de dados em formato de tabelas no Excel consolidada, ela deve ser hospedada em nuvem. De forma a demonstrar a variedade de plataformas de *cloud* presentes numa companhia de grande porte, com um gigantesco volume de dados, o armazenamento das tabelas foi dividido em duas plataformas principais: Google Planilhas e um servidor local criado utilizando o XAMPP Control Panel gerido pelo MySQL Workbench. Para isso, algumas adaptações na formatação de dados tiveram de ser efetuadas devido às particularidades de cada plataforma e, posteriormente, à forma de inclusão de novos dados pelos operadores e analistas responsáveis que atuariam no processo. Para abordar essa hospedagem, é necessário fazer a seguinte segmentação, que abordará os conteúdos de acordo:

- Tabela de Reportes: Google Planilhas + Google Formulários;
- Tabelas de Sensores, Processos e OSs: XAMPP Control Panel + MySQL Workbench + Python 3.13 + Visual Studio Code (com extensões).

4.2.1 *Hospedagem da Tabela de Reportes*

A tabela de Reportes foi elaborada de forma a receber os dados de diagnóstico de problemas observados nos componentes de uma CT. Para isso, o Google Planilhas em conjunto com o Google Formulários foram utilizados. A integração com o Google Formulários garantiu a automação de "carimbos de data/hora" e a replicação automática de fórmulas para novos registros via contagem incremental. Portanto, dois formulários foram criados antes da importação da tabela de Reportes para o Google Planilhas, sendo denominados como:

- Reportes — Criação;
- Reportes — Adição.

O que os diferencia é que existem dois dados que podem ser adicionados posteriormente na tabela de Reportes, visto que envolvem análise posterior por pessoal responsável pela manutenção:

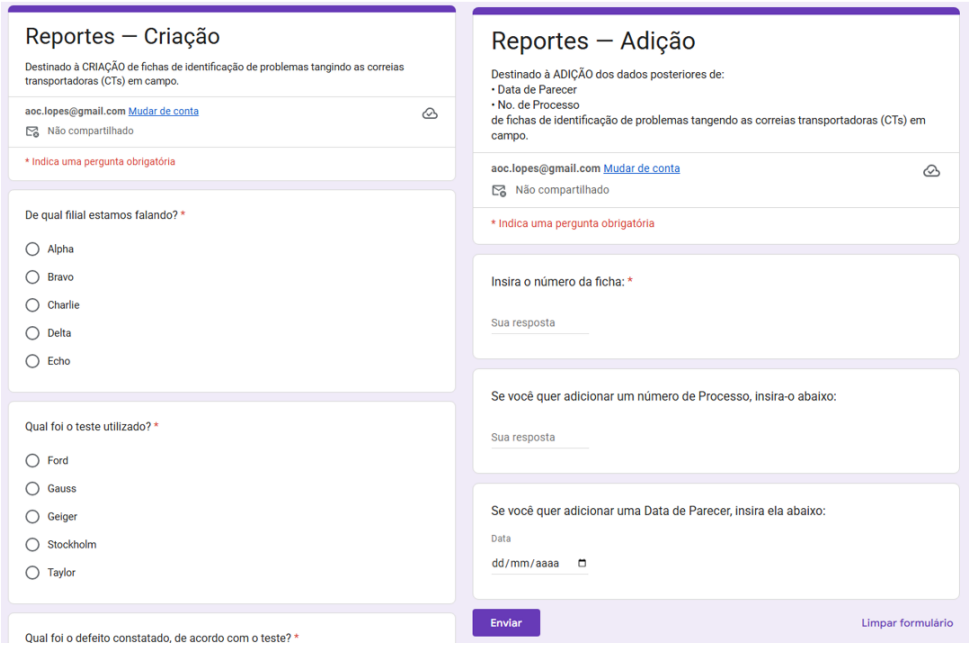
os campos de "Processo" e "Data do Parecer". Para isso, o formulário de adição de reportes foi criado.

No formulário de criação de reportes, o usuário deve preencher os campos de:

- Filial, apresentado como "De qual filial estamos falando?";
- Teste, apresentado como "Qual foi o teste utilizado?";
- Problema, resultante da mesma fórmula utilizada na tabela de Reportes para sua criação (ver seção 4.1.1, juntamente com o campo "Qual foi o defeito constatado, de acordo com o teste?";
- Local, apresentado como "Qual é o local em que você identificou o problema?";
- Código da Estrutura, resultante da concatenação do campo anterior como campo "Nesse local, qual é o código da estrutura em que você identificou o problema?";
- Código do Componente, resultante da concatenação dos dois campos mencionados anteriormente com o dígito do campo "Nessa estrutura, qual é o código do componente em que o problema foi identificado?";
- Processo, apresentado como "Já existe um processo em que essa ficha deva ser atribuída? Se sim, preencha o campo abaixo com o código de 6 algarismos.".

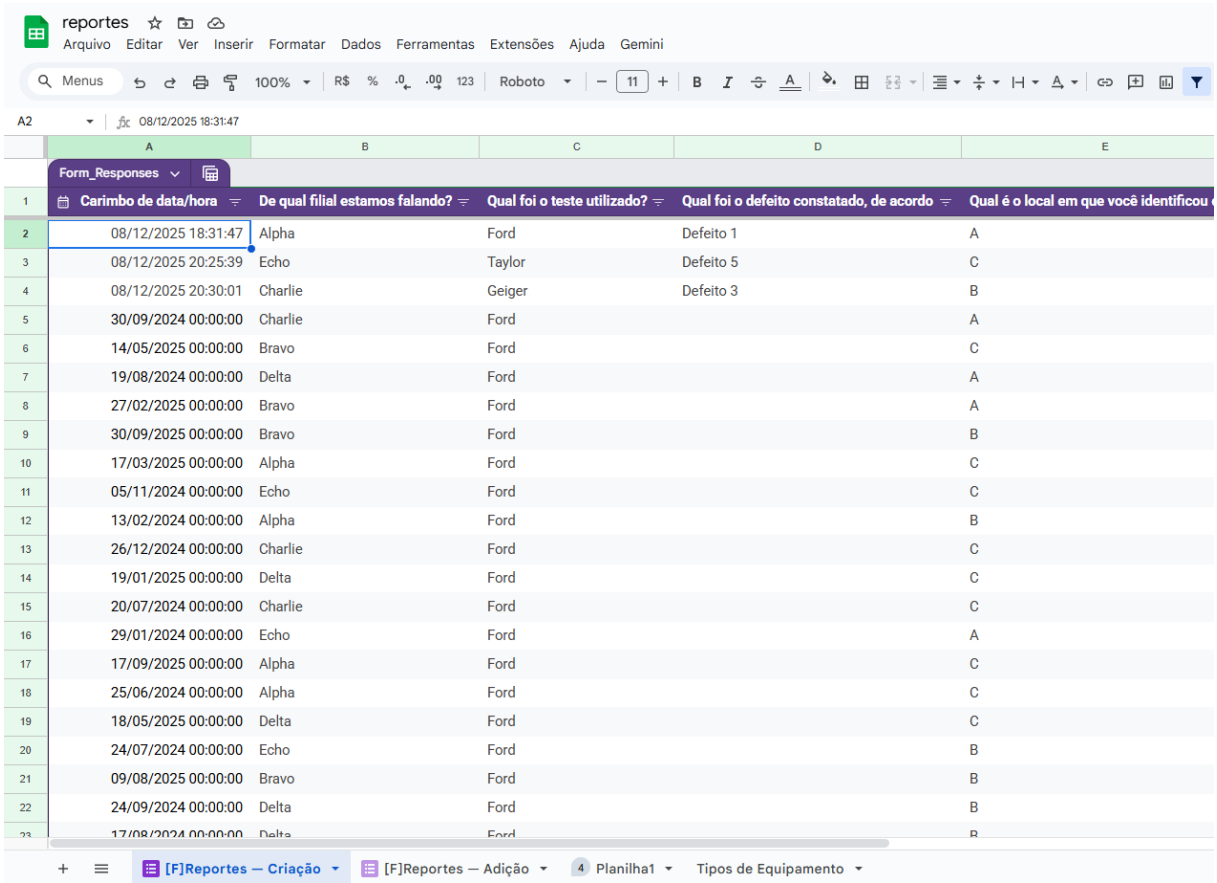
Todas as perguntas, exceto a última, são de caráter obrigatório, visto que são dados que o usuário deve possuir de antemão. O campo de processo, entretanto, pode ser inserido no mesmo momento, caso o usuário já tenha ciência de que o problema identificado tenha de ser inserido já num número de processo específico. Como segundo passo, foi necessário vincular uma Planilha Google de despejo dos dados do preenchimento. Na figura 8, é possível observar alguns desses campos, e como eles são armazenados no Google Planilhas através da 9.

Figura 8 – Formulários associados à tabela de Reportes.



Fonte: Elaborado pelo autor, a partir do Google Formulários (2026).

Figura 9 – Planilha Google da tabela de Reportes.



Fonte: Google Formulários, 2026.

Esse recebimento de formulários torna necessário algumas modificações no formato de entrada de dados na tabela de Reportes. Como o próprio preenchimento dos formulários gera um carimbo de tempo contendo data e hora na planilha de despejo, o campo de Data de Criação passou a se tornar ele ("Carimbo de data/hora" na Planilha Google), em que todas as datas anteriores receberam o horário de 00:00:00 concatenado após as mesmas, não resultando em nenhum impacto relevante naquele campo.

O campo de Ficha, anteriormente gerado aleatoriamente, passou a ser automaticamente gerado através de contagem incremental, que conta a quantidade de valores já presentes na coluna em questão, subtrai seu número atual e soma um. A soma do número um é necessária para a não-contabilização da linha ocupada pelo cabeçalho. Por programação interna do software, todas as fórmulas presentes na coluna mais recente da planilha são replicadas em cada novo preenchimento, garantindo seu funcionamento automatizado.

No formulário de adição de dados aos reportes, os seguintes campos são disponibilizados:

- "Insira o número da ficha:";
- "Se você quer adicionar um número de Processo, insira-o abaixo:";
- "Se você quer adicionar uma Data de Parecer, insira ela abaixo:",

Adicionalmente, outro carimbo de tempo é gerado automaticamente pelo formulário a cada resposta ao formulário, sendo desprezado aqui, visto que é de geração automática e sem utilidade para o escopo do projeto. Nesse formulário, apenas o campo que solicita o número de ficha é obrigatório, visto que o usuário com o desejo de inserir dados na mesma pode conter apenas um dos dois dados, ou ambos. Então, um vínculo é realizado trazendo os dados que são armazenados nessa planilha auxiliar oriunda do formulário em pauta, para a tabela de Reportes. Para o campo de "Data do Parecer", a seguinte lógica é feita: caso a busca pela ficha correspondente à linha atual, na tabela de adição, não retorne uma correspondência exata com a data procurada e falhe, ocasionando um erro, o campo é mantido em branco. Isso é realizado através das funções VLOOKUP (PROCV nativo) e IFERROR (SEERRO nativo).

Para o campo de "Processo", uma lógica similar, mas mais complexa é realizada. Aqui, uma cláusula IF adicional é inserida entre o VLOOKUP e o IFERROR, fazendo uma busca dupla tanto no campo via formulário principal (de criação), quanto no campo via formulário auxiliar (de adição), buscando preservar o valor preenchido em qualquer um deles. Com isso, é concluída a migração de formato Excel, para Google Planilhas integrado ao Google Formulários.

4.2.2 *Hospedagem das Tabelas de Sensores, Processos e OSs*

A hospedagem dessas tabelas foram realizadas de uma forma diferente da tabela de Reportes, fazendo uso de um servidor local utilizando o XAMPP Control Panel e o MySQL

Workbench como ferramenta de SGBD. Para o *upload* das tabelas contidas em Excel para o servidor local instalado, foi necessária a criação de um código em MySQL (ver apêndice A.2.1) utilizando o VSC. Em primeiro lugar, foi necessário criar um novo esquema relacional para o banco de dados do projeto, denominado "tcc", que seria o banco de dados propriamente dito. Então, foi necessário criar as tabelas de Sensores, Processos, OSs e seus campos respectivos via código, e definir o tipo de variável que cada campo poderia receber, variando dentre VARCHAR, INT, DATE, TIME e DECIMAL. A criação dessas tabelas não foi realizada diretamente pelo MySQL por conta das limitações da linguagem de programação, em que funções como MD5, SHA e dicionários alimentados manualmente para cada coluna tornariam a criação dos dados vistos até agora muito mais morosa e difícil, tornando a Prova de Conceito imprática mediante à pluralidade de tipos de dados aqui criados.

Após a criação de cada tabela, um mecanismo de importação dos dados armazenados em Excel foi criado. Para isso, foi preciso que os arquivos XLSX fossem salvos como arquivos CSV — formato editável até mesmo em bloco de notas, em que as colunas são separadas por vírgulas. Nesse mecanismo, uma conversão automatizada foi implementada, em função de duas divergências existentes por conta do padrão brasileiro de que o formato de armazenamento de datas comumente conhecido difere do formato aceito pelo MySQL. Para isso, uma variável intermediária foi criada naqueles campos — "@data_texto"— que formatava a data antes de ser enviada para o banco de dados.

Outra conversão se demonstrou necessária referente ao formato de armazenamento de números decimais. No formato brasileiro, as casas decimais são separadas por vírgulas — enquanto o MySQL adota o ponto. Para isso, uma conversão dentro do próprio Excel foi realizada, em que os valores originais, contendo um grande número de casas decimais, foram truncados para apenas duas e as vírgulas substituídas por pontos.

Com o servidor local em operação, o código de upload em MySQL fora copiado para a tela de edição nativa do programa e executado. Então, formulários executáveis programados em linguagem Python foram criados para realizar a adição de novas entradas à base de dados em MySQL.

Cada tabela deste projeto demandou um código intermediador MySQL Workbench-Python — representando um formulário para cada (ver apêndices A.2.2 a A.2.4). Então, após a importação de todas as bibliotecas necessárias para a execução, uma função foi definida, responsável por estabelecer a conexão com o servidor local, fornecer o código de tradução para MySQL e obter os valores oriundos dos formulários fazendo uso da função *get()*. Ademais, uma mensagem de sucesso e uma de erro foram definidas, para o caso de falha em qualquer parte desse processo, informando o usuário do caso de não-inserção do que foi preenchido em formulário.

Como este é um código que gera um formulário — ou seja, todos os *inputs* para os campos são enviados ao código simultaneamente, a biblioteca *tkinter* foi utilizada para realizar a retenção desses valores em GUI até o momento de envio. No código de tradução para MySQL,

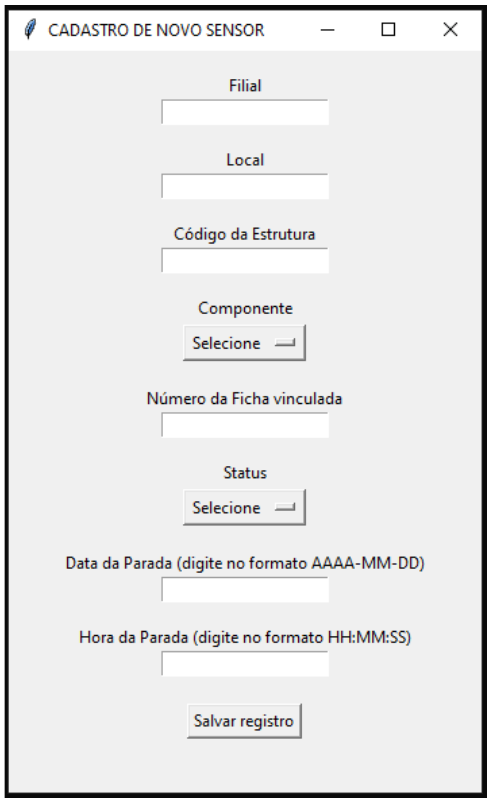
os caracteres *%s* funcionam como *placeholders* de variável, representando que a função fica aguardando o envio dos mesmos.

No restante do código, é possível observar os comandos de abertura da janela referente ao formulário e seus campos. Tratando-se da janela, uma dimensão de *pop-up* e título customizados foram definidos para cada formulário. Quanto aos campos, eles apresentaram como de inserção por digitação (sufixo *var*) e caixa de seleção (prefixo *entry*). Essencialmente, cada campo tinha seu agrupamento de código composto por três comandos:

1. O de posicionamento do rótulo do campo — *tk.label().pack();*
2. O de transcrição do campo para a variável — *var=tk.entry(janela);*
3. E o de posicionamento da caixa de entrada/seleção — *var.pack();*

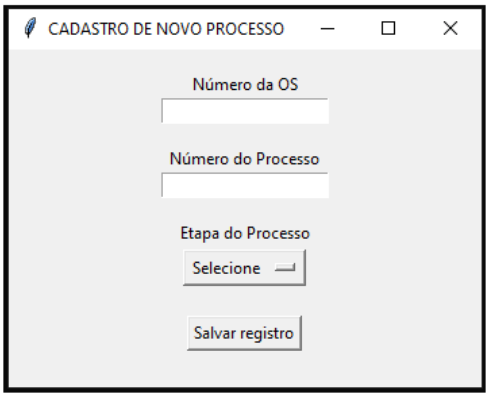
Na figuras 10 à 12, é possível observar a GUI desses formulários criados em Python, composto pelas caixas de seleção e de inserção de dados previamente citadas.

Figura 10 – Formulário Python para inserção de dados na tabela de Sensores.



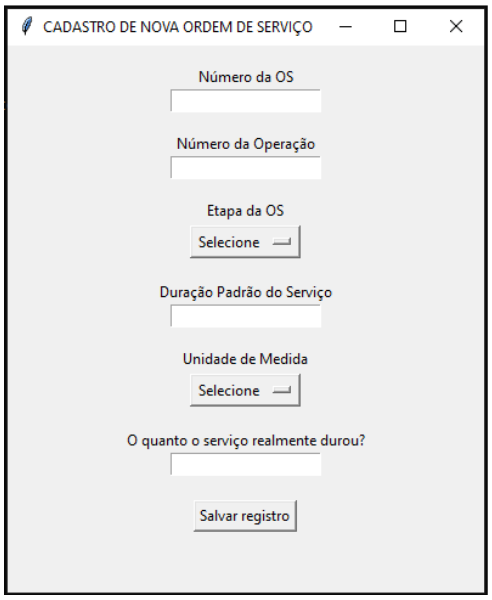
Fonte: Elaborado pelo autor, 2026.

Figura 11 – Formulário Python para inserção de dados na tabela de Processos.



Fonte: Elaborado pelo autor, 2026.

Figura 12 – Formulário Python para inserção de dados na tabela de OSs.



Fonte: Elaborado pelo autor, 2026.

O formulário referente à adição de entradas na tabela de Sensores contou, em caráter exclusivo, com a criação de uma função que associasse o componente selecionado, descrito por extenso, com um código numérico respectivo a ele, em que foi criada uma espécie de seletor *switch*, como visto na figura 13.

Figura 13 – Estrutura do seletor de componente no formulário de Processos.

```
def atualizar_tipo_componente(valor):
    mapa = {
        "Estrutura": 1,
        "Esteira": 2,
        "Rolete": 3,
        "Mancal": 4,
        "Truque": 5,
        "Rolamento": 6,
        "Motor": 7,
        "Redutor": 8
    }

    global entry_Tipo_de_Componente
    entry_Tipo_de_Componente = mapa.get(valor)

tk.Label(janela, text="\nComponente").pack()
Componente_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, Componente_var,
"Estrutura", "Esteira", "Rolete", "Mancal", "Truque", "Rolamento", "Motor", "Redutor",
command=atualizar_tipo_componente).pack()
```

Fonte: Elaborado pelo autor, a partir do Visual Studio Code (2026).

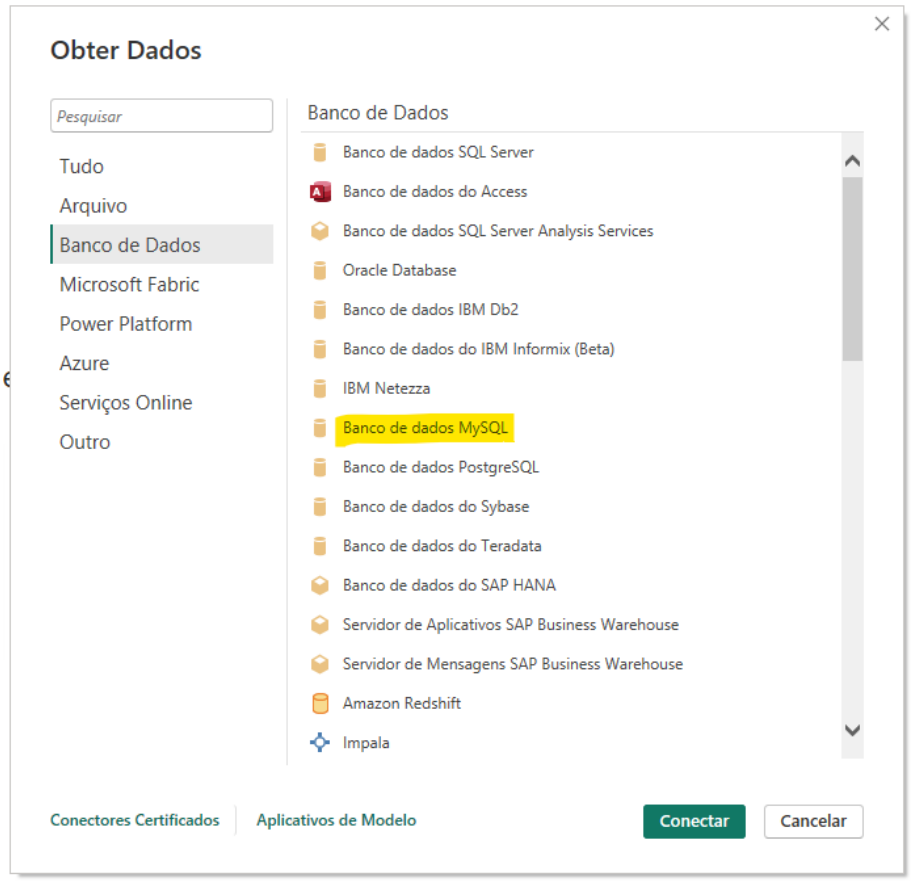
Com os códigos concebidos, a hospedagem da base de dados se dava como concluída, através do Google Formulários e do servidor MySQL.

4.3 *Front-end* em Power BI

Visualizações por tabelas e planilhas não fornecem um bom controle analítico sobre processos, sendo necessários gráficos dinâmicos para a observação de métricas e a tomada de decisões precisas e eficazes, resultando em processos eficientes e mais econômicos (COTA, 2024).

Uma ferramenta que se destaca no ramo de Extração, Transformação e Carregamento de dados (Extract, Transform, Load no inglês — ETL) é o Microsoft Power BI, que permite a criação de relatórios dinâmicos dentro de uma plataforma altamente modular com uma vasta gama de conexões a todo tipo de bancos de dados, além de automações e complementos via *script*, aliados à uma excelente GUI.

Figura 14 – Conector MySQL para servidor local.



Fonte: Microsoft Power BI, 2026.

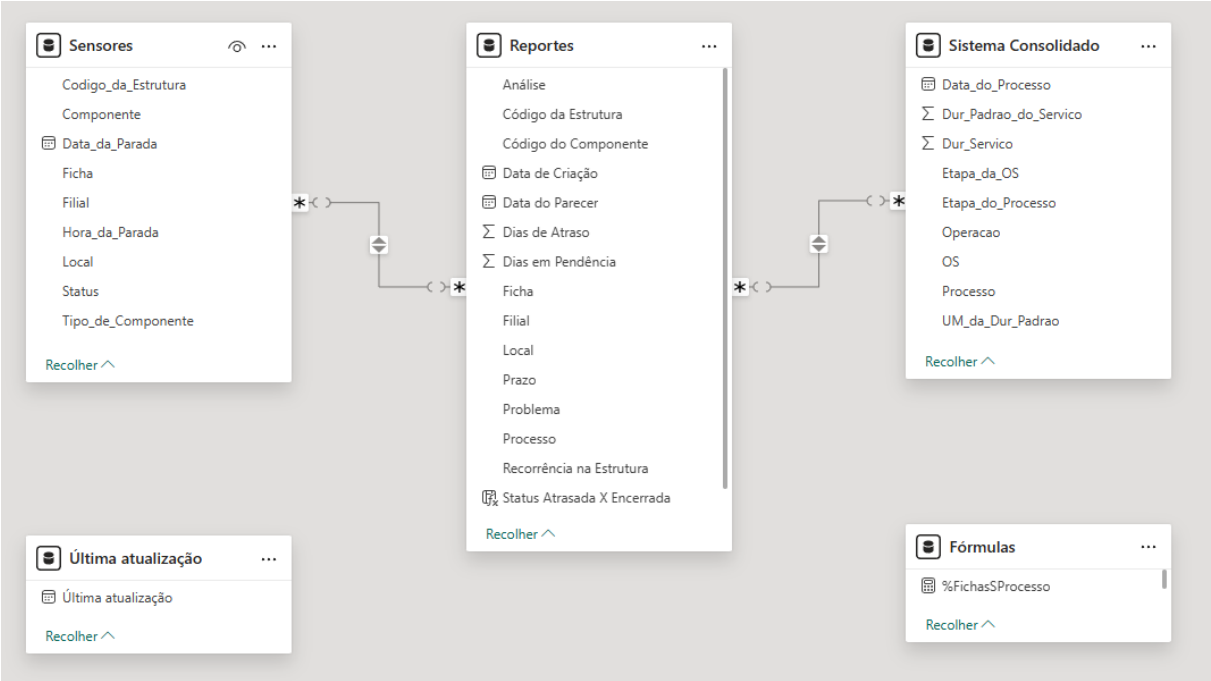
Para sua elaboração, foi necessário conectar a base de dados ao editor de consultas Power Query interno à ferramenta do Power BI, vinculando o banco de dados em MySQL com o serviço do Power BI (ver figura 14). Ao conectá-las, algumas etapas foram aplicadas às consultas envolvendo a formatação dos campos e cabeçalhos da base de dados advinda das plataformas de hospedagem. Aqui, duas etapas ganham destaque:

- Na consulta da tabela "Processos", uma mesclagem foi realizada com a consulta da tabela "Reportes" por junção à direita pelo campo de "Processos", mantendo apenas aqueles registros que contivessem os processos contidos na consulta "Reportes". Isso foi realizado no intuito de manter apenas os registros pertinentes ao processo, simulando uma extração direcionada dentro de um *data lake*;
- As consultas de "Processos" e "OSs" foram mescladas como novas, constituindo a consulta "Sistema Consolidado", através de junção à esquerda pelo campo de "OS", mantendo apenas os registros que contivessem as OSs da consulta de "Processos". Essa mesclagem foi conduzida para centralizar a análise que tange a parte administrativa do processo em apenas uma tabela, facilitando a visualização dos dados no *front-end* do relatório.

Com a consulta "Sistema Consolidado" criada, as consultas de Processos e OSs se tornavam dispensáveis e, por isso, não foram carregadas para o *front-end* do relatório. Uma consulta para obtenção da última data de atualização do relatório também fora criada (ver apêndice A.3.1).

Saindo do Power Query interno ao aplicativo do Power BI e indo para a interface de elaboração de relatórios, os vínculos por campos-chave foram estabelecidos, através de relações muitos-para-muitos (ver figura 15) utilizando os campos de ficha e processo das consultas de Sensores, Reportes e Sistema Consolidado (de acordo com o diagrama na figura **??**), uma vez que elas se encontravam hospedadas *online*.

Figura 15 – Relacionamentos criados no Power BI.



Fonte: Microsoft Power BI, 2026.

Na página do relatório, exposta de forma completa no apêndice A.3.15, diversos visuais — denominação do software para todos os elementos dinâmicos inseríveis no relatório, como filtros, gráficos, matrizes, cartões, etc. — foram criados tendo em vista prover um alto nível de controle e variedade de análises com os dados selecionados para o projeto. Eles fornecem análises de:

- **Taxa de vínculo e consistência entre consultas:** os visuais, através de várias análises, ilustram a taxa de aderência ao preenchimento correto de fichas, processos e ordens, e o quão bem a base de dados está sendo manutenida. Essa informação se torna relevante para a identificação de lacunas tangendo a gestão do processo de manutenção.

- **Análises temporais:** tudo que tange o campo de *lead-time*, que é o tempo total desde o início de um processo até sua conclusão, abrangendo todas as etapas intermediárias como aprovação e inspeção, sendo crucial para a eficiência de negócio e logística, onde sua redução aumenta a satisfação do cliente e reduz custos.

- **Desempenho por diversos níveis de localidade:** torna possível parametrizar e comparar o desempenho entre filiais, e também identificar problemas por localidade, seja ela generalizada ou mais específica.

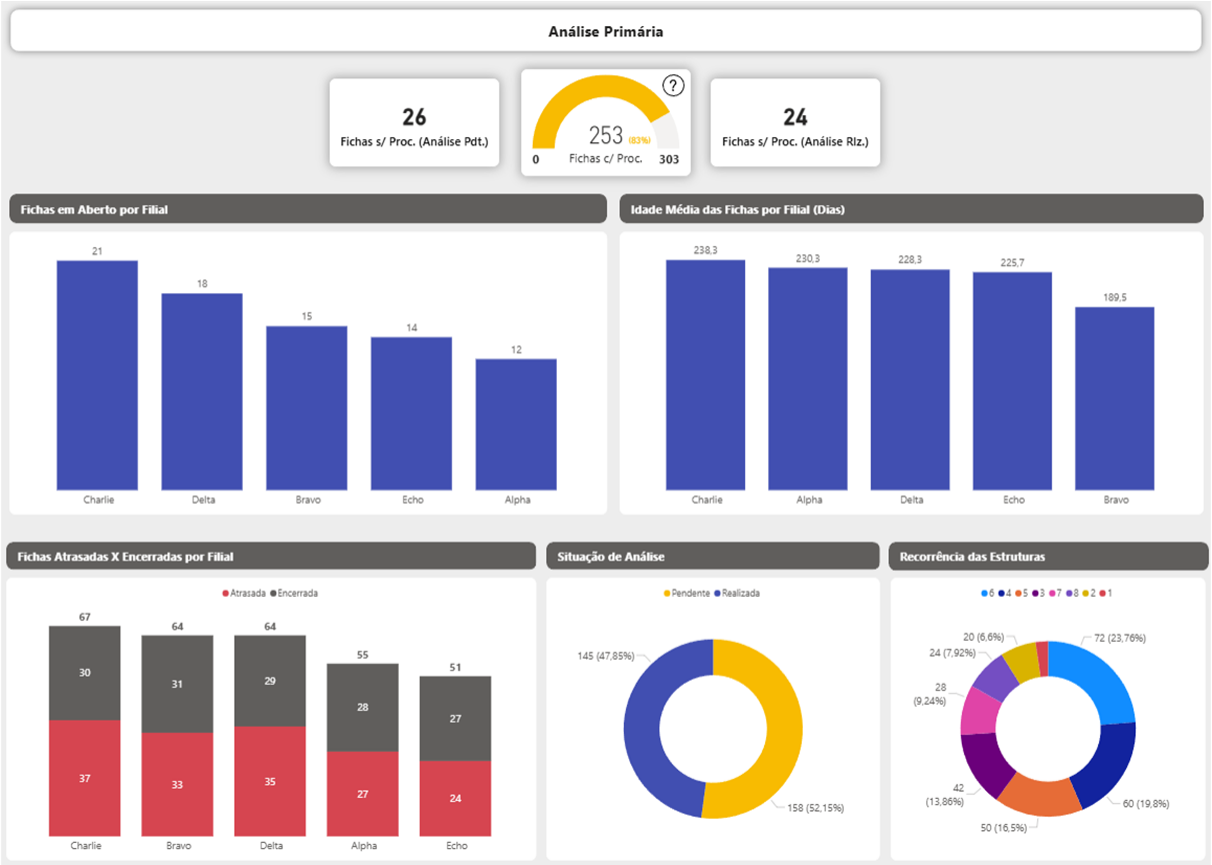
- **Status das fichas e dados correlacionados:** permitem a identificação de necessidades e priorização de esforços dentro da companhia, como por exemplo, através da identificação de equipes que possam estar formando um *backlog* — lista de atividades e demandas que precisam ser executadas — muito grande numa certa etapa da cadeia de manutenção.

- **Pareto:** uma técnica de tomada de decisão que utiliza o Princípio 80/20, indicando que 80% dos problemas (efeitos) geralmente advêm de apenas 20% das causas. Essa metodologia prioriza esforços para focar nas causas essenciais que geram maior impacto, facilitando a melhoria contínua e a eficiência.

Esses visuais dinâmicos são acompanhados por filtros de data, localidade, prazo, status e busca por processo, além de filtragem cruzada entre visuais. Para que todas as consultas criadas fossem possíveis, diversas fórmulas DAX (*Data Analysis Expressions* — uma linguagem de fórmulas) foram utilizadas para criar cálculos personalizados, análises avançadas e novas informações a partir do grupo de dados existentes.

Os filtros do relatório foram organizados de forma a seguirem uma ordem, da esquerda para a direita, da menor para a maior granulação de dados, começando pelo de "Data de Criação", da consulta de Reportes, até um filtro de processo por processo, vindo da mesma consulta. O relacionamento estabelecido previamente garante que essa filtragem cruzada se torne efetiva por toda a base de dados.

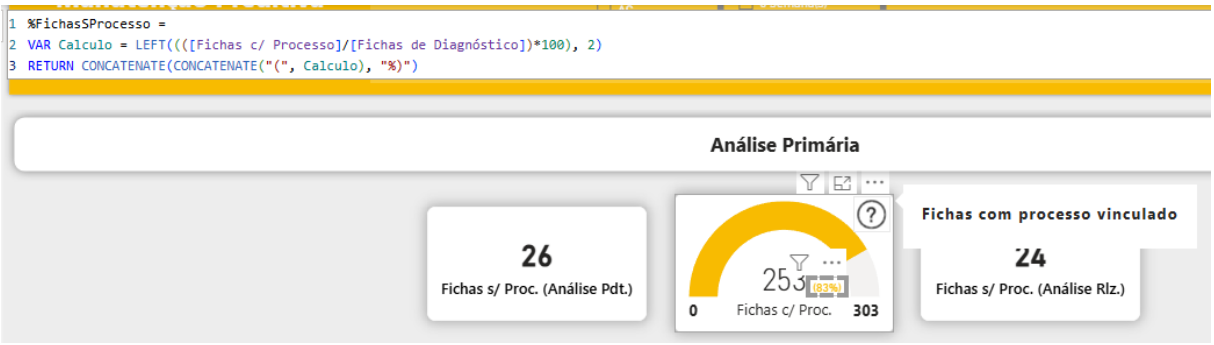
Figura 16 – Seção de Análise Básica do relatório.



Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

Os visuais foram organizados de forma a realizar um aumento gradual no nível de complexidade das análises à medida que o usuário rolasse a página para baixo, sendo iniciado no agrupamento denominado **Análise Básica** (ver figura **??**). De início, eram encontrados três visuais relativos à integridade e manutenção da base de dados, como é observado na figura 17, fazendo uso de cinco medidas DAX distintas. Eles foram divididos entre um indicador — visual utilizado para mostrar o desempenho atual de uma métrica ou indicador-chave de desempenho (*Key Process Indicator* no inglês — KPI) em relação a uma meta, utilizando uma escala semelhante a um velocímetro — e dois cartões — elemento para destacar um único número ou métrica chave, fornecendo uma visão rápida e direta ideal para resumos executivos.

Figura 17 – Fórmula DAX utilizada para um cartão composto no Power BI.



Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

- No indicador, duas dessas fórmulas era utilizadas: uma de contagem de linhas da consulta de Reportes, simbolizando o número de fichas (ver apêndice A.3.3), e outra que contava essas linhas desprezando aquelas que não possuíam registro de número de processo no campo de "Processo"(ver apêndice A.3.2). Uma terceira medida era sobreposta a esse indicador, que indicava, através da criação uma variável e uma fórmula de retorno, a porcentagem da diferença dos cálculos das fórmulas citadas anteriormente aqui, como mostrado na figura 17 e no apêndice A.3.4. Em adição, para facilitar o entendimento do indicador, uma dica de ferramenta era encontrada no mesmo, sendo um visual da comunidade importado da biblioteca online.
- Nos cartões à esquerda e direita, essa mesma contagem de fichas sem registro era realizada, filtrando aquelas que possuíam os termos "Pendente" ou "Realizada" no campo de "Análise"(ver apêndices A.3.5 e A.3.6).

Em sequência a esses cartões, era possível encontrar os primeiros gráficos dinâmicos do relatório. O gráfico de **Fichas em Aberto por Filial** representava uma análise de lacuna de fichas não solucionadas por localização macro, filtrado internamente para exibir apenas fichas que estivessem em análise ou em espera de acordo com o campo de "Etapa do Processo" da consulta Sistema Consolidado. O de **Idade Média das Fichas por Filial** ilustrou um indicador de performance, se beneficiando de uma filtragem por data para a análise por período. O gráfico de **Fichas Atrasadas versus Encerradas por Filial** utilizou da mesma construção do primeiro gráfico deste parágrafo; entretanto, perdendo o filtro daquele visual e sendo segmentado pela coluna calculada criada através de funções DAX "Status Atrasada X Encerrada", exibindo apenas aquelas que contivessem os termos "Encerrada" ou "Atrasada" de acordo com a sua lógica, observada na figura 18 e no apêndice A.3.7.

Figura 18 – Coluna calculada em DAX utilizada no relatório.

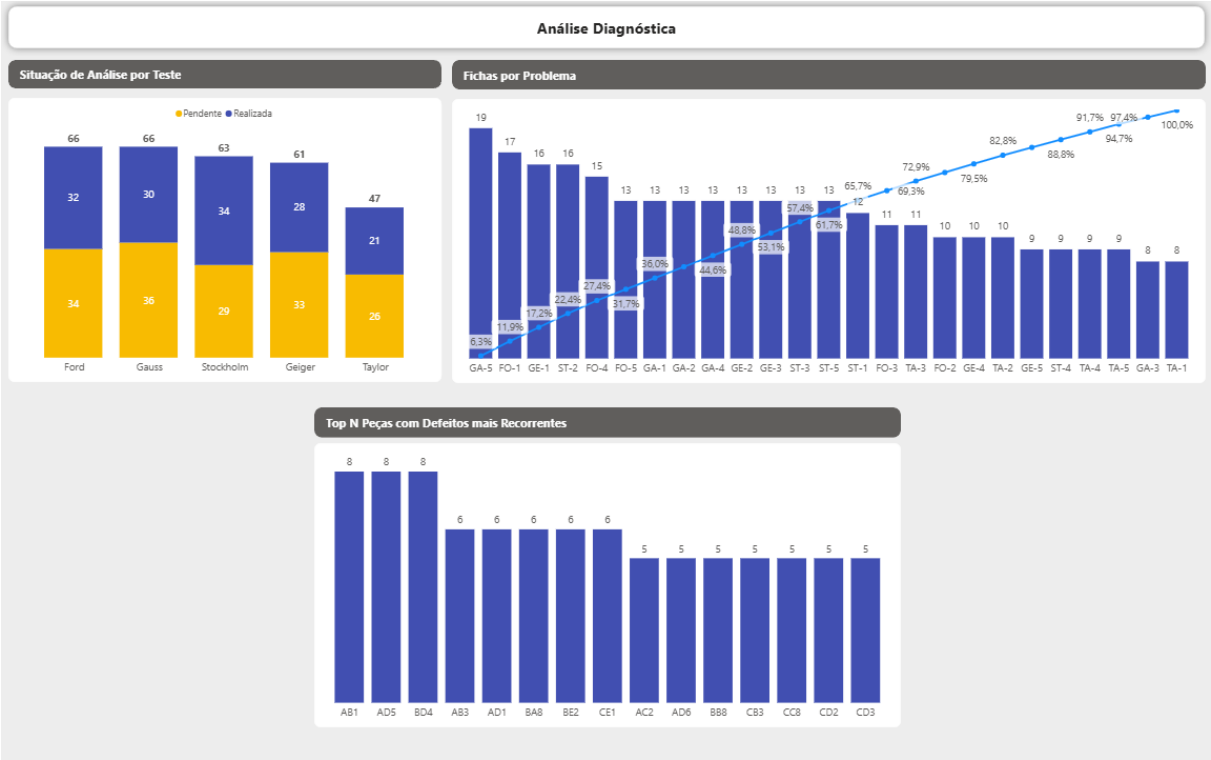
Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

O gráfico de rosca **Situação de Análise** exibia o número e proporção de fichas de acordo com seus dois status, e o de **Recorrência nas Estruturas**, a quantidade de recorrências dentro das fichas existentes, assim encerrando o primeiro agrupamento.

No grupo de **Análise Diagnóstica**, exposto na figura 19, três gráficos de colunas e barras empilhadas podiam ser observados. O de **Situação de Análise por Teste** contava as fichas e as dividia pelos termos do campo "Teste", e exibia o status de análise por cada um deles. O gráfico de **Fichas por Problema** classificava os problemas com maior taxa de reincidência nos componentes das CTs, apresentando uma linha de Pareto que permite observar de forma mais objetiva, através de um total acumulado, quais são os problemas com maior participação para a piora da saúde das plantas. Para a sua criação, três medidas DAX tiveram de ser utilizadas para o funcionamento correto da linha, podendo ser observadas nos apêndices A.3.8 à A.3.10.

O gráfico **Top N Peças com Defeitos mais Recorrentes** representou um ranking das peças com o maior número de fichas associadas, sendo limitado dinamicamente em no mínimo 10 e a no máximo *n* registros de número igual de fichas ao 10º registro, no intuito de prover uma representação prática dos principais agentes sujeitos a quebras dentre um volume muito grande de resultados, que dificultaria a obtenção de análises úteis do visual.

Figura 19 – Seção de Análise Diagnóstica do relatório.



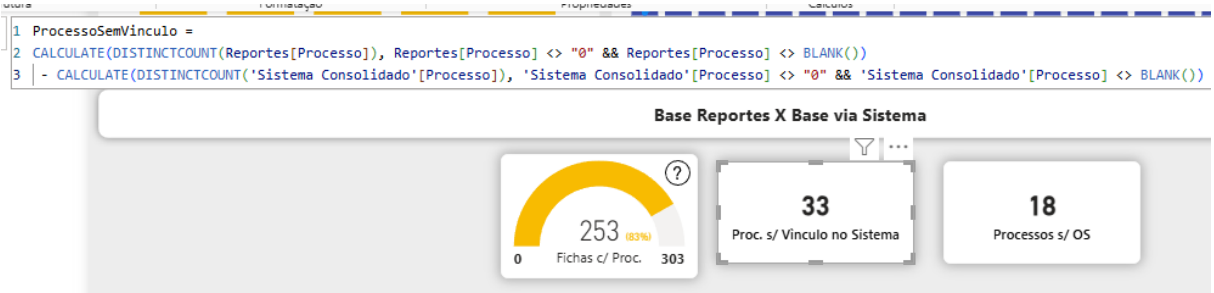
Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

O grupamento de **Base Reportes versus Base via Sistema** (ver figura 20) representa uma análise qualitativa da base de dados, expondo informações que demonstram a consistência entre as informações das consultas. Ele se iniciava com um indicador composto conhecido previamente, replicado aqui em função de seu caráter fundamental para o relatório, acompanhado de dois novos cartões e suas medidas DAX respectivas. No primeiro, a diferença entre a conta distinta de números de processo na consulta de Reportes era feita com a conta distinta de números de processo na consulta de Sistema Consolidado, mantendo apenas aqueles processos que não possuíssem vínculo no sistema (ver figura 21 e apêndice A.3.11).

Figura 20 – Seção comparativa entre consultas da base de dados.

Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

Figura 21 – Cartão de número de processos sem vínculo.



Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

O segundo cartão tratava do vínculo Processo-OS, fazendo a diferença do número de linhas com número de Processo para as com número de OS, retratando o número de processos sem Ordem de Serviço (ver figura 22 e apêndice A.3.11).

Figura 22 – Cartão de número de processos no sistema sem OS associada.

Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

O primeiro gráfico a ser encontrado na seção era o de **Status das Fichas Vide Sistema**, fazendo a conta do número de fichas por Etapa da OS, desconsiderando aquelas OSs que não possuíssem número. O próximo era o de **Fichas com Processo Vinculado por Filial**, visto na figura 23, utilizando de duas medidas DAX: a de contagem de fichas com processo, já utilizada no indicador da figura 17, e a sua medida inversa — a contagem sem processo (ver apêndice A.3.13. Essas duas contagens são empilhadas e divididas por filial.

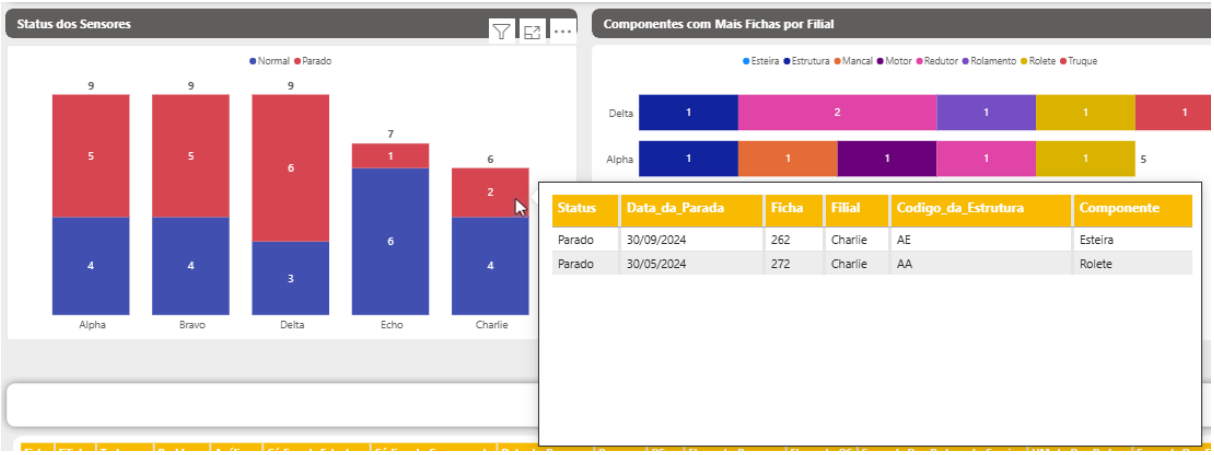
Figura 23 – Gráfico empilhado de fichas com processo vinculado por filial.

Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

O gráfico de **Fichas com Processo sem OS por Filial** faz exatamente o que o título sugere através de filtragem exclusiva, mantendo apenas aquelas fichas que se encontram com análise pendente de acordo com o campo "Análise". Por fim, existe o gráfico de **Fichas Pendentes de Alteração na Base Reportes**, que utiliza uma medida DAX de análise de atualização da consulta de Reportes (ver apêndice A.3.14), contando aqueles Processos que possuam "Executado" no campo de etapa da OS e análise pendente — esse resultado exibido por filial novamente. Assim, esse grupamento se dava como encerrado, tendo utilizado diversas dicas de ferramenta para auxílio do entendimento.

Como penúltima seção, se encontrava a de **Análise dos Sensores Instalados em Campo**, representando aqueles já instalados para reduzir fluxo de operadores para medições periódicas. Toda essa seção utilizou exclusivamente a consulta de Sensores. O primeiro gráfico, denominado **Status dos Sensores**, contou o número de fichas e o exibiu por filial, dividindo esse resultado de acordo com o status dos sensores. Uma peculiaridade desse gráfico é que ele possui uma dica de ferramenta dinâmica em formato de tabela (ver figura 24), possibilitando observar individualmente os dados de local e data de parada de cada sensor, acelerando a identificação de gargalos nas plantas. O segundo gráfico ilustrava os **Componentes com Mais Fichas em Aberto por Filial**, através de uma visualização em barras para melhor visibilidade das parcelas associadas a cada componente.

Figura 24 – Dica de ferramenta em formato de tabela associada ao gráfico de Status dos Sensores no Power BI.



Fonte: Elaborado pelo autor, a partir do Microsoft Power BI (2026).

Na última seção, se encontrava uma tabela de consulta dos dados com diversos campos das consultas de Reportes e Sistema Consolidado, permitindo sua extração de dados através de botão e comparação do somatório de tempo dos campos de "Duração do Serviço" e "Duração Padrão do Serviço", para observação de performance. Esses campos podem ser encontrados na visualização geral da página do relatório no apêndice A.3.15 no fim desta monografia.

O relatório contou com uma cortina acinzentada para melhor leitura noturna e visualização dos visuais, aliado a um visual moderno e simplificado. O termo "Base" foi utilizado em alguns títulos em detrimento do termo correto "Consulta" para um entendimento mais simplificado por leitores pouco instruídos sobre o campo de análise de dados, buscando tornar o relatório o mais intuitivo possível para todos, visto que esses são termos mais comumente utilizados, mesmo que equivocadamente.

5 CONCLUSÃO E TRABALHOS FUTUROS

A informatização do registro e análise de dados da manutenção das CTs, aqui através do uso de ferramentas Google, de um banco de dados MySQL e do Microsoft Power BI, representaram o alcance dos objetivos deste trabalho, demonstrando ser uma solução tecnicamente viável e estrategicamente necessária para a competitividade e agilidade de tomada de decisões estratégicas.

Apesar dos limites encontrados no uso da IA para a limitação da quantidade de linhas geradas em algumas colunas na criação das tabelas, como observado na seção 4.1, a sua variabilidade respeitou a Lei dos Grandes Números, o que simbolizou que esta ainda sim foi uma solução viável para a execução do projeto.

Se tratando da hospedagem de bases de dados, o MySQL, no espaço de SGBD, e o Google Planilhas aliado ao Google Formulários permitiu uma hospedagem semelhante a um cenário real no que tange a um *data lake* de um ambiente corporativo, ao mesmo tempo que essas se apresentaram como soluções de custo zero na implementação em nuvem da base de dados.

O Microsoft Power BI permitiu que diversas análises, outrora complexas fora daquela plataforma e de atualização da informação morosa, fossem criadas possibilitando um acompanhamento *up-to-date* dos dados e paradas de equipamento, representando uma plataforma altamente modular e com alto potencial para aplicações futuras.

Com base na arquitetura já construída e nos resultados obtidos, vislumbram-se as seguintes direções para a continuidade desta pesquisa:

- **Integração com Dados Reais de Campo:** A etapa lógica subsequente é a transição da base de dados fictícios para uma conexão direta com os sensores instalados nos ativos de uma companhia real, substituindo a alimentação via formulários por um fluxo de dados em tempo real via sistema supervisório ou SAP HANA;
- **Implementação de Modelos de *Machine Learning (ML)*:** Conforme abordado na revisão bibliográfica, o sistema pode ser evoluído para incluir análises prognósticas, utilizando redes neurais ou modelos matemáticos avançados para prever o tempo de vida útil restante de motores e redutores, baseando-se no balanço entre custos de reparo e probabilidade de falha;
- **Desenvolvimento de Gêmeos Digitais (*Digital Twins*):** Sugere-se a criação de uma réplica digital completa das correias transportadoras, permitindo que gestores realizem simulações de cenários de estresse operacional e planejem manutenções preventivas com base no comportamento virtual do equipamento;
- **Expansão da Mobilidade e Alertas:** O desenvolvimento de um aplicativo mobile que utilize as interfaces em Python (tkinter) ou o Power BI Mobile para enviar alertas automáticos

via notificações *push* aos técnicos quando um sensor atingir níveis críticos de trepidação ou temperatura.

- **Validação e Adequação por Necessidade Específica:** É possível coletar *feedback* sobre a aplicabilidade da solução desenvolvida, possibilitando fazer uma análise qualitativa e melhor adaptar a mesma para o atendimento de demandas e visões mais específicas através de um formulário ou entrevista com usuários que testassem a interface.

REFERÊNCIAS

AHMADI, S.; MOOSAZADEH, S.; HAJIHASSANI, M.; MOOMIVAND, H.; RAJAEI, M. Reliability, availability and maintainability analysis of the conveyor system in mechanized tunneling. **Measurement**, v. 145, p. 756–764, 10 2019. ISSN 02632241. Citado na página 19.

AISHWARYA, C. K.; LAHARI, C. S.; SAHEB, S. H. Data analytics tools and applications for business and finance systems. In: ____. [S.l.]: Auerbach Publications, 2024. p. 18–34. Citado na página 19.

AL-KAHWATI, K.; BIRK, W.; NILSFORS, E. F.; NILSEN, R. 6. experiences of a digital twin based predictive maintenance solution for belt conveyor systems. **PHM Society European Conference**, 2022. Citado na página 19.

BARBIERI, G.; LASERNA, J.; MATEUS, L. M. Towards a business intelligence application for evidence-based maintenance. **IFAC-PapersOnLine**, v. 58, p. 37–42, 2024. ISSN 24058963. Citado na página 19.

BRANCO, M. S. P. S. **Digitalisation in the Portuguese Banking System: A Business Intelligence Solution to Strengthen Supervisory Knowledge**. [S.l.], 2023. Disponível em: <https://www.proquest.com/openview/6360066fd2e20d2e5d0332763ce5ab20/1?pq-origsite=gscholar&cbl=2026366&diss=y>. Citado na página 19.

BULL, D. J.; WHEELER, C. A.; BULL, J. P.; MEYER, S. H. Patent Application, **A sensor assembly and monitoring system for an idler roller in a belt conveyor system**. US 2022/0281690 A1: [s.n.], 2022. Citado na página 19.

CODD, E. F. A relational model of data for large shared data banks. **Communications of the ACM**, v. 13, p. 377–387, 6 1970. ISSN 0001-0782. Citado na página 17.

COTA, L. C. **Desenvolvimento de aplicativo de help desk para solicitação de compra de equipamentos de proteção individual**. Tese (Doutorado) — Instituto Federal de Minas Gerais, 2 2024. Disponível em: <https://hdl.handle.net/20.500.14387/1767>. Citado na página 44.

DUARTE, J. C.; CUNHA, P. F.; CRAVEIRO, J. T. Maintenance database. **Procedia CIRP**, v. 7, p. 551–556, 2013. ISSN 22128271. Citado na página 17.

GARG, A.; DESHMUKH, S. Maintenance management: literature review and directions. **Journal of Quality in Maintenance Engineering**, v. 12, p. 205–238, 7 2006. ISSN 1355-2511. Citado na página 17.

KHAN, A. H. Effective decision making using data analytics. **International Journal of Scientific Research in Engineering and Management**, v. 08, p. 1–5, 4 2024. ISSN 25823930. Citado na página 19.

LABIB, A. W. World-class maintenance using a computerised maintenance management system. **Journal of Quality in Maintenance Engineering**, v. 4, p. 66–75, 3 1998. ISSN 1355-2511. Citado na página 18.

LLAVE, M. R. Business intelligence and analytics in small and medium-sized enterprises: A systematic literature review. **Procedia Computer Science**, v. 121, p. 194–205, 2017. ISSN 18770509. Citado na página 18.

MAFIA, M. M. P.; AYOUB, N.; TRUMPLER, L.; HANSEN, J. P. de O. A digital twin design for conveyor belts predictive maintenance. In: ____. [S.l.]: Springer, Cham, 2024. p. 111–119. Citado na página 19.

MUCHIRI, P.; PINTELON, L.; GELDERS, L.; MARTIN, H. Development of maintenance function performance measurement framework and indicators. **International Journal of Production Economics**, v. 131, n. 1, p. 295–302, May 2011. Disponível em: <https://ideas.repec.org/a/eee/proeco/v131y2011i1p295-302.html>. Citado na página 13.

OJHA, R.; SINGH, R.; SINGH, A. Big data analysis: Structuring of data. In: **2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)**. [S.l.]: IEEE, 2017. p. 144–147. ISBN 978-1-5090-4797-0. Citado na página 18.

PICOZZI, P.; NOCCO, U.; PEZZILLO, A.; COSMO, A. D.; CIMOLIN, V. The use of business intelligence software to monitor key performance indicators (kpis) for the evaluation of a computerized maintenance management system (cmms). **Electronics**, v. 13, p. 2286, 6 2024. ISSN 2079-9292. Citado na página 19.

PULCINI, V.; MODONI, G. E. 3. machine learning-based digital twin of a conveyor belt for predictive maintenance. **The International Journal of Advanced Manufacturing Technology**, 2024. Citado na página 19.

RUSSO, M.; FERRARI, A. **The definitive guide to DAX : business intelligence with Microsoft Power BI, SQL server analysis services, and Excel**. [S.l.]: Published with the authorization of Microsoft Corporation by Pearson Education, 2020. ISBN 1509306978. Citado na página 18.

SALONEN, A.; BENGTSSON, M.; FRIDHOLM, V. The possibilities of improving maintenance through cmms data analysis. In: ____. [S.l.]: IOS Press, 2020. Citado na página 19.

SHAFIEE, M. Maintenance strategy selection problem: an mcdm overview. **Journal of Quality in Maintenance Engineering**, v. 21, p. 378–402, 10 2015. ISSN 1355-2511. Citado na página 20.

SHERWIN, D. A review of overall models for maintenance management. **Journal of Quality in Maintenance Engineering**, v. 6, p. 138–164, 9 2000. ISSN 1355-2511. Citado na página 17.

SHINDE, M. B. G.; SHIVTHARE, D. S. Impact of data visualization in data analysis to improve the efficiency of machine learning models. **Journal of Advanced Zoology**, p. 107–112, 3 2024. Citado na página 19.

STüRMER, M.; NUSSBAUMER, J.; STöCKLI, P. Security implications of digitalization: The dangers of data colonialism and the way towards sustainable and sovereign management of environmental data. **Computer**, 7 2021. Citado 2 vezes nas páginas 13 e 18.

TSANG, A. H.; JARDINE, A. K.; KOLODNY, H. Measuring maintenance performance: a holistic approach. **International Journal of Operations & Production Management**, v. 19, p. 691–715, 7 1999. ISSN 0144-3577. Citado na página 13.

APÊNDICE A – CÓDIGOS PRINCIPAIS REFERENTES AO DESENVOLVIMENTO DO PROJETO

A.1 Concepção da Base de Dados

A.1.1 *Código Python via ChatGPT para geração da coluna "Técnica"*

```
import random

terms = ["Gauss", "Stockholm", "Taylor", "Geiger", "Ford"]
counter = 0
limit = 300

while True:
    counter += 1
    print(random.choice(terms))
    if counter >= limit:
        break
```

A.1.2 *Código Python via ChatGPT para geração da coluna "Local"*

```
import random

letters = ["A", "B", "C"]
count = 300

for _ in range(count):
    print(random.choice(letters))

print("TOTAL_LINHAS =", count)
```

A.1.3 *Código de Power Query referente à criação da tabela de OS*

```
let
    Fonte = Excel.CurrentWorkbook(){[Name="Tabela1"]}[Content],
    #"Tipo Alterado" = Table.TransformColumnTypes(Fonte,{{"cods",
    Int64.Type}, {"mult", Int64.Type}}),
    #"Personalização Adicionada" = Table.AddColumn(#"Tipo Alterado",
    "Personalizar", each List.Numbers(1, [mult])),
    #"Personalizar Expandido" = Table.ExpandListColumn(
    #"Personalização Adicionada", "Personalizar"),
```

```
    #"Colunas Removidas" = Table.RemoveColumns(
    #"Personalizar Expandido", {"mult"}),
    #"Linhas Filtradas" = Table.SelectRows(#"Colunas Removidas",
    each [cods] <> null and [cods] <> "")
in
    #"Linhas Filtradas"
```

A.1.4 *Fórmula de Excel para a criação do campo "Duração do Serviço"*

```
=F9*(0,66+(1,5-0,66)*ALEATÓRIO())
```

A.2 Formato de Hospedagem

A.2.1 *Código MySQL de upload de arquivos CSV*

```
CREATE SCHEMA `tcc`;

CREATE TABLE base_sensores(
    Filial VARCHAR(20),
    `Local` VARCHAR(3),
    Codigo_da_Estrutura VARCHAR(10),
    Tipo_de_Componente INT,
    Componente VARCHAR(30),
    Ficha INT,
    `Status` VARCHAR(20),
    Data_da_Parada DATE,
    Hora_da_Parada TIME
);

LOAD DATA INFILE 'D:/Ruindows/Downloads/IFMG/TCC/Novas Bases/Server -
Base Sensores.csv'
INTO TABLE base_sensores
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(Filial, `Local`, Codigo_da_Estrutura, Tipo_de_Componente,
Componente, Ficha, `Status`, @data_texto1, Hora_da_Parada)
SET Data_da_Parada = STR_TO_DATE(@data_texto1, '%d/%m/%Y');

--------------------------------
```

```
CREATE TABLE ordens_de_servico(
    OS INT,
    Operacao INT,
    Etapa_da_OS VARCHAR(20),
    Dur_Padrao_do_Servico DECIMAL(4,2),
    UM_da_Dur_Padrao VARCHAR(10),
    Dur_Servico DECIMAL(4,2)
);

LOAD DATA INFILE 'D:/Ruindows/Downloads/IFMG/TCC/Novas Bases/Server -
Ordens de Serviço.csv'
INTO TABLE base_sensores
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(OS, Operacao, Etapa_da_OS, Dur_Padrao_do_Servico, UM_da_Dur_Padrao,
Dur_Servico);

---------------------------------

CREATE TABLE processos(
    OS INT,
    Processo INT,
    Data_do_Processo DATE,
    Etapa_do_Processo VARCHAR(20)
)

LOAD DATA INFILE 'D:/Ruindows/Downloads/IFMG/TCC/Novas Bases/Server -
Processos.csv'
INTO TABLE processos
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
(OS, Processo, @data_texto2, Etapa_do_Processo)
SET Data_do_Processo = STR_TO_DATE(@data_texto2, '%d/%m/%Y');
```

A.2.2 *Código intermediador da tabela de Sensores*

```
# No console:
```

```
# 1. D:
# 2. cd D:\Windows\Downloads\IFMG\TCC
# 3. python Intermediador_base_sensores_gui.py

import tkinter as tk
from tkinter import messagebox
import mysql.connector


# ===== FUNÇÃO PARA SALVAR NO BANCO =====
def salvar_dados():
    try:
        conexao = mysql.connector.connect(
            host = "localhost",
            user = "root",
            password = "",
            database = "tcc"
        )
        cursor = conexao.cursor()

        sql = """
        INSERT INTO base_sensores
        (Filial, Local, Codigo_da_Estrutura, Tipo_de_Componente,
        entry_Tipo_de_Componente, Ficha, Status, Data_da_Parada,
        Hora_da_Parada)
        VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s)
        """

        valores = (
            entry_Filial.get(),
            entry_Local.get(),
            entry_Codigo_da_Estrutura.get(),
            entry_Tipo_de_Componente.get(),
            Componente_var.get(),
            entry_Ficha.get(),
            Status_var.get(),
            entry_Data_da_Parada.get(),
            entry_Hora_da_Parada.get()
        )
```

```
        cursor.execute(sql, valores)
        conexao.commit()
        messagebox.showinfo(" REGISTRO INSERIDO COM SUCESSO!")
        cursor.close()
        conexao.close()

    except Exception as e:
        messagebox.showerror("Erro", str(e))


# ===== JANELA PRINCIPAL =====
janela = tk.Tk()
janela.title("CADASTRO DE NOVO SENSOR")
janela.geometry("350x550")


# ===== CAMPOS =====
tk.Label(janela, text="\nFilial").pack()
entry_Filial = tk.Entry(janela)
entry_Filial.pack()

tk.Label(janela, text="\nLocal").pack()
entry_Local = tk.Entry(janela)
entry_Local.pack()

tk.Label(janela, text="\nCódigo da Estrutura").pack()
entry_Codigo_da_Estrutura = tk.Entry(janela)
entry_Codigo_da_Estrutura.pack()


def atualizar_tipo_componente(valor):
    mapa = {
        "Estrutura": 1,
        "Esteira": 2,
        "Rolete": 3,
        "Mancal": 4,
        "Truque": 5,
        "Rolamento": 6,
```

```
        "Motor": 7,
        "Redutor": 8
    }

    global entry_Tipo_de_Componente
    entry_Tipo_de_Componente = mapa.get(valor)

tk.Label(janela, text="\nComponente").pack()
Componente_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, Componente_var, "Estrutura", "Esteira",
"Rolete", "Mancal", "Truque", "Rolamento", "Motor", "Redutor",
command=atualizar_tipo_componente).pack()


tk.Label(janela, text="\nNúmero da Ficha vinculada").pack()
entry_Ficha = tk.Entry(janela)
entry_Ficha.pack()

tk.Label(janela, text="\nStatus").pack()
Status_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, Status_var, "Parado", "Normal").pack()

tk.Label(janela, text="\nData da Parada (digite no formato
AAAA-MM-DD)").pack()
entry_Data_da_Parada = tk.Entry(janela)
entry_Data_da_Parada.pack()

tk.Label(janela, text="\nHora da Parada (digite no formato
HH:MM:SS)").pack()
entry_Hora_da_Parada = tk.Entry(janela)
entry_Hora_da_Parada.pack()


# ===== BOTÃO =====
tk.Button(janela, text="Salvar registro",
command=salvar_dados).pack(pady=20)

janela.mainloop()
```

A.2.3 *Código intermediador da tabela de Processos*

```
# No console:
# 1. D:
# 2. cd D:\Windows\Downloads\IFMG\TCC
# 3. python Intermediador_processos_gui.py

import tkinter as tk
from tkinter import messagebox
import mysql.connector
from datetime import date


# ===== FUNÇÃO PARA SALVAR NO BANCO =====
def salvar_dados():
    try:
        conexao = mysql.connector.connect(
            host = "localhost",
            user = "root",
            password = "",
            database = "tcc"
        )
        cursor = conexao.cursor()

        sql = """
        INSERT INTO processos
        (OS, Processo, Data_do_Processo, Etapa_do_Processo)
        VALUES (%s, %s, %s, %s)
        """

        valores = (
            entry_OS.get(),
            entry_Processo.get(),
            Data_do_Processo,
            Etapa_do_Processo_var.get()
        )

        cursor.execute(sql, valores)
        conexao.commit()
        messagebox.showinfo(" REGISTRO INSERIDO COM SUCESSO!")
```

```
        cursor.close()
        conexao.close()

    except Exception as e:
        messagebox.showerror("Erro", str(e))


# ===== JANELA PRINCIPAL =====
janela = tk.Tk()
janela.title("CADASTRO DE NOVO PROCESSO")
janela.geometry("350x250")


# ===== CAMPOS =====
tk.Label(janela, text="\nNúmero da OS").pack()
entry_OS = tk.Entry(janela)
entry_OS.pack()

tk.Label(janela, text="\nNúmero do Processo").pack()
entry_Processo = tk.Entry(janela)
entry_Processo.pack()

Data_do_Processo = date.today().strftime("%Y-%m-%d")

tk.Label(janela, text="\nEtapa do Processo").pack()
Etapa_do_Processo_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, Etapa_do_Processo_var, "Analisando",
"Aguardando OS", "Encerrado").pack()


# ===== BOTÃO =====
tk.Button(janela, text="Salvar registro",
command=salvar_dados).pack(pady=20)

janela.mainloop()
```

A.2.4 *Código intermediador da tabela de OS*

```
# No console:
# 1. D:
```

```
# 2. cd D:\Windows\Downloads\IFMG\TCC
# 3. python Intermediador_ordens_de_servico_gui.py

import tkinter as tk
from tkinter import messagebox
import mysql.connector


# ===== FUNÇÃO PARA SALVAR NO BANCO =====
def salvar_dados():
    try:
        conexao = mysql.connector.connect(
            host = "localhost",
            user = "root",
            password = "",
            database = "tcc"
        )
        cursor = conexao.cursor()

        sql = """
        INSERT INTO ordens_de_servico
        (OS, Operacao, Etapa_do_Processo, Dur_Padrao_do_Servico,
        UM_da_Dur_Padrao, Dur_Servico)
        VALUES (%s, %s, %s, %s, %s, %s)
        """

        valores = (
            entry_OS.get(),
            entry_Operacao.get(),
            Etapa_da_OS_var.get(),
            entry_Dur_Padrao_do_Servico.get(),
            UM_da_Dur_Padrao_var.get(),
            entry_Dur_Servico.get()
        )

        cursor.execute(sql, valores)
        conexao.commit()
        messagebox.showinfo(" REGISTRO INSERIDO COM SUCESSO!")
        cursor.close()
```

```
        conexao.close()

    except Exception as e:
        messagebox.showerror("Erro", str(e))


# ===== JANELA PRINCIPAL =====
janela = tk.Tk()
janela.title("CADASTRO DE NOVA ORDEM DE SERVIÇO")
janela.geometry("350x450")


# ===== CAMPOS =====
tk.Label(janela, text="\nNúmero da OS").pack()
entry_OS = tk.Entry(janela)
entry_OS.pack()

tk.Label(janela, text="\nNúmero da Operação").pack()
entry_Operacao = tk.Entry(janela)
entry_Operacao.pack()

tk.Label(janela, text="\nEtapa da OS").pack()
Etapa_da_OS_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, Etapa_da_OS_var, "Planejando", "Agendado",
"Executado").pack()

tk.Label(janela, text="\nDuração Padrão do Serviço").pack()
entry_Dur_Padrao_do_Servico = tk.Entry(janela)
entry_Dur_Padrao_do_Servico.pack()

tk.Label(janela, text="\nUnidade de Medida").pack()
UM_da_Dur_Padrao_var = tk.StringVar(value="Selecione")
tk.OptionMenu(janela, UM_da_Dur_Padrao_var, "horas", "dias").pack()

tk.Label(janela, text="\nO quanto o serviço realmente durou?").pack()
entry_Dur_Servico = tk.Entry(janela)
entry_Dur_Servico.pack()
```

```
# ===== BOTÃO =====
tk.Button(janela, text="Salvar registro",
command=salvar_dados).pack(pady=20)

janela.mainloop()
```

A.3 Front-end em Power BI

A.3.1 *Código M para obtenção de última atualização do relatório*

```
let
    Fonte = DateTime.LocalNow()
in
    Fonte
```

A.3.2 *Medida DAX para contagem de fichas com número de processo na consulta de Reportes*

```
Fichas c/ Processo =
CALCULATE(COUNTROWS(Reportes),
Reportes[Processo] <> "Sem registro")
```

A.3.3 *Medida DAX para contagem de fichas na consulta de Reportes*

```
Fichas de Diagnóstico = COUNTROWS(Reportes)
```

A.3.4 *Medida DAX para porcentagem de fichas com número de processo na consulta de Reportes*

```
%FichasSProcesso =
VAR Calculo =
LEFT((([Fichas c/ Processo]/[Fichas de Diagnóstico])*100), 2)
RETURN CONCATENATE(CONCATENATE("(", Calculo), "%)")
```

A.3.5 *Medida DAX para contagem de fichas sem número de processo na consulta de Reportes e análise pendente*

```
FichaSProcessoPendente =
COUNTAX(
    FILTER(Reportes,
        Reportes[Processo] = "Sem registro" &&
```

```
        Reportes[Análise] = "Pendente"
    ),
    Reportes[Processo]
)
```

A.3.6 *Medida DAX para contagem de fichas sem número de processo na consulta de Reportes e análise realizada*

```
FichaSProcessoRealizado =
COUNTAX(
    FILTER(Reportes,
        Reportes[Processo] = "Sem registro" &&
        Reportes[Análise] = "Realizada"
    ),
    Reportes[Processo]
)
```

A.3.7 *Coluna calculada DAX de status da ficha*

```
Status Atrasada X Encerrada =
IF(Reportes[Análise] = "Realizada", "Encerrada",
IF(Reportes[Análise] <> "Encerrada"
&& Reportes[Dias de Atraso] > 0, "Atrasada", "No prazo"))
```

A.3.8 *Medida DAX de conta contextual da análise de Pareto de fichas por problema*

```
P_QtdFichas = COUNT(Reportes[Ficha])
```

A.3.9 *Medida DAX de conta total da análise de Pareto de fichas por problema*

```
P_TotalFichas =
CALCULATE(
    [P_QtdFichas],
    ALL(Reportes[Problema])
)
```

A.3.10 *Medida DAX de conta acumulada da análise de Pareto de fichas por problema*

```
P_Problemas =
```

```
VAR ProbAtual = SELECTEDVALUE (Reportes[Problema])
VAR QtdAtual  = [P_QtdFichas]
VAR Acumulado =
    CALCULATE (
        [P_QtdFichas],
        FILTER (
            ALL(Reportes[Problema]),
            [P_QtdFichas] > QtdAtual || (
                    [P_QtdFichas] = QtdAtual
                    && Reportes[Problema] <= ProbAtual
                )
        )
    )
RETURN
DIVIDE (Acumulado, [P_TotalFichas])
```

A.3.11 *Medida DAX de processo sem vínculo no sistema*

```
ProcessoSemVinculo =
CALCULATE(DISTINCTCOUNT(Reportes[Processo]),
Reportes[Processo] <> "0" &&
Reportes[Processo] <> BLANK())
 - CALCULATE(DISTINCTCOUNT('Sistema Consolidado'[Processo]),
 'Sistema Consolidado'[Processo] <> "0" &&
 'Sistema Consolidado'[Processo] <> BLANK())
```

A.3.12 *Medida DAX de processo no sistema sem OS*

```
VinculoSemOS =
DISTINCTCOUNTNOBLANK('Sistema Consolidado'[Processo])
- DISTINCTCOUNTNOBLANK('Sistema Consolidado'[OS])
```

A.3.13 *Medida DAX para contagem de fichas sem número de processo na consulta de Reportes*

```
Fichas s/ Processo =
CALCULATE(COUNTROWS(Reportes),
Reportes[Processo] = "Sem registro")
```

A.3.14 *Medida DAX para contagem de fichas pendentes de alteração na consulta de Reportes*

```
PendenteEncrRep =
CALCULATE(COUNT(Reportes[Processo]),
Reportes[Análise] <> "Realizada",
'Sistema Consolidado'[Etapa_da_OS] = "Executado")
```

A.3.15 *Relatório Dinâmico completo em Power BI.*

Informatização da Manutenção Preditiva

Data de Criação: Selecionar tudo; 2024; 2025
Código da Estrutura: Selecionar tudo; A; B; C
Prazo: Selecionar tudo; 1 semanal(s); 2 semanal(s); 3 semanal(s); 4 semanal(s); 5 semanal(s); 6 semanal(s); 7 semanal(s)
Etapa_do_Proce...: Aguardando OS; Analisando; Encerrado
Etapa_da_OS: (Em branco); Agendado; Executado; Planejando
Processo: Pesquisar; 167465; 300007; 300041; 300066; 300080; 300104; 300111

22/12/25 05:08
Última atualização

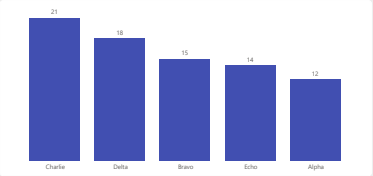
Análise Primária

26
Fichas s/ Proc. (Análise Pdt.)

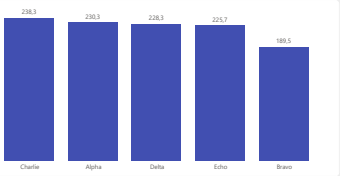
253 (83%)
0 Fichas c/ Proc. 303

24
Fichas s/ Proc. (Análise Rlz.)

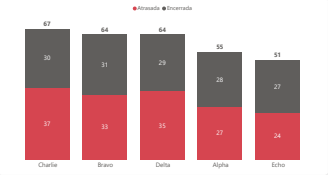
Fichas em Aberto por Filial



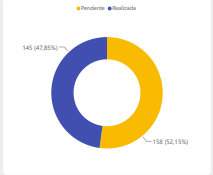
Idade Média das Fichas por Filial (Dias)



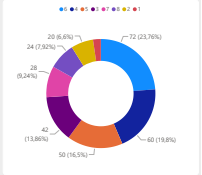
Fichas Atrasadas X Encerradas por Filial



Situação de Análise

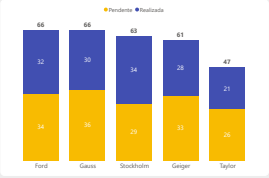


Recorrência das Estruturas

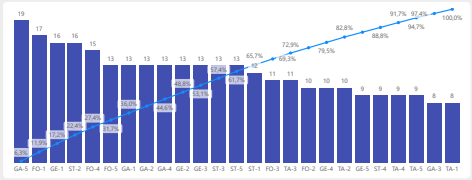


Análise Diagnóstica

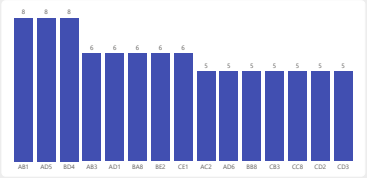
Situação de Análise por Teste



Fichas por Problema



Top N Peças com Defeitos mais Recorrentes



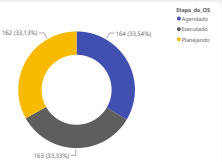
Base Reportes X Base via Sistema

253 (83%)
0 Fichas c/ Proc. 303

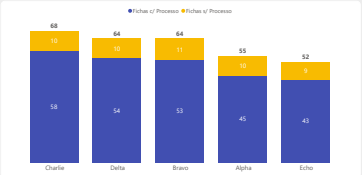
33
Proc. s/ Vínculo no Sistema

18
Processos s/ OS

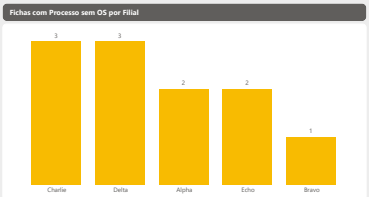
Status das Fichas Vide Sistema



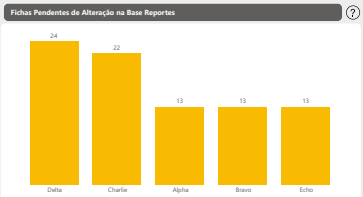
Fichas com Processo Vinculado por Filial



Fichas com Processo sem OS por Filial

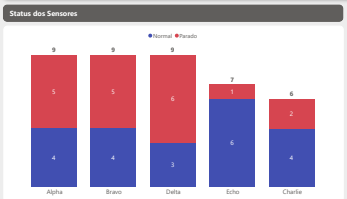


Fichas Pendentes de Alteração na Base Reportes



Análise dos Sensores Instalados em Campo

Status dos Sensores



Componentes com Mais Fichas em Aberto por Filial

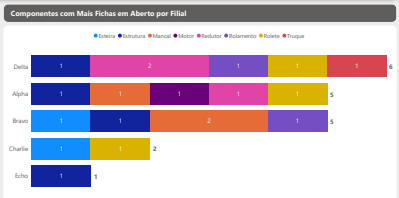


Tabela de Consulta

Ficha	Filial	Teste	Problema	Análise	Código da Estrutura	Código do Componente	Data_do_Processo	Processo	OS	Etapa_do_Processo	Etapa_da_OS	Soma da Dur_Padrao_do_Serviço	UM_da_Dur_Padrao	Soma da Dur_Servid
1	Bravo	Gauss	GA-2	Realizada	BC	BC1	27/09/2025	309253	58845	Encerrado	Agendado	9	horas	10,2
1	Bravo	Gauss	GA-2	Realizada	BC	BC1	27/09/2025	309253	58845	Encerrado	Executado	5	dias	5,9
1	Bravo	Gauss	GA-2	Realizada	BC	BC1	27/09/2025	309253	58845	Encerrado	Executado	2	horas	2,5
1	Bravo	Gauss	GA-2	Realizada	BC	BC1	27/09/2025	309253	58845	Encerrado	Planejando	1	horas	1,2
10	Charlie	Gauss	GA-3	Pendente	AB	AB5	09/08/2024	302242	15770	Analisando	Agendado	3	dias	3,0
10	Charlie	Gauss	GA-3	Pendente	AB	AB5	09/08/2024	302242	15770	Analisando	Executado	2	dias	2,5
10	Charlie	Gauss	GA-3	Pendente	AB	AB5	09/08/2024	302242	15770	Analisando	Executado	10	horas	10,0
10	Charlie	Gauss	GA-3	Pendente	AB	AB5	09/08/2024	302242	15770	Analisando	Planejando	3	dias	4,2
10	Charlie	Gauss	GA-3	Pendente	AB	AB5	09/08/2024	302242	15770	Analisando	Planejando	8	horas	10,2
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Agendado	2	dias	2,4
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Agendado	20	horas	29,9
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Executado	13	dias	13,5
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Executado	6	horas	6,2
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Planejando	2	dias	3,0
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	36729	Encerrado	Planejando	2	horas	1,4
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	78590	Encerrado	Agendado	5	dias	4,9
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	78590	Encerrado	Agendado	10	horas	13,3
100	Echo	Ford	FO-3	Realizada	BB	BB1	25/07/2024	309216	78590	Encerrado	Planejando	6	dias	4,4
Total												4335		4.713,8