LINKS Veda(V2) システム設計書

1. 開発スコープ

1.1. 概要

1.1.1　LINKS Veda 開発の背景

国土交通省をはじめとする行政機関では、EBPM（Evidence-Based Policy Making、エビデンスに基づく政策立案）の推進が求められており、政策判断の根拠となるデータを収集・整理・活用する能力が行政の重要な課題となっている。しかし、行政機関が日常業務で作成・蓄積する文書の多くは、紙帳票・PDF・スキャン画像など機械処理が困難な形式のままであり、データとして有効活用されていないのが実態である。
国土交通省の職員は、担当部門に分かれ、それぞれ異なる行政分野（ドローン・船舶・観光・建設・物流等）を管轄している。こうした部門に共通して発生するのが、民間事業者や補助金受給事業者などから様式の定まった報告書・申請書類を受け取り、その内容を集計して実績報告や政策資料に仕上げる「集計業務」である。
現場では、紙や PDF で届いた書類を職員が手作業で表計算ソフトに転記し、集計・グラフ作成を行ったうえで、最終的にプレゼンテーション資料や定点観測レポートにまとめるという作業が広く行われている。こうした手作業には膨大な工数がかかるだけでなく、転記ミスや集計誤りといった品質上のリスクも伴う。一部の部門では独自システムを構築して自動化しているケースもあるが、すべての部門がそのような環境整備を行うことは難しく、多くの現場では職員が人力で作業を続けている。
LINKS Veda は、こうした課題を技術で解消することを目的として立ち上げられた取り組みである。各部門が保有する非構造データを簡単に構造化して転記業務の省力化を実現し、さらにノーコードで資料作成・データ可視化ができる環境を提供すること

で、行政現場の業務負荷を抜本的に軽減する基盤を目指している。

1.1.2　課題解決のアプローチ

LINKS Veda では、以下の 3 つのアプローチで課題解決に取り組む。

1.データ構築モジュールの開発

「データ構築モジュール」は、LLM（大規模言語モデル）と OCR（光学文字認識）を活用した非構造データの自動構造化処理を中核機能とするサブシステムである。PDF や紙文書から意味情報を抽出し、あらかじめ定義したスキーマに従って構造化データを生成する。生成されたデータはデータテーブルとして管理され、BI での分析や G 空間情報センターへのオープンデータ公開に利用される。

2.「データ活用モジュール（LINKS BI）」の開発

「データ活用モジュール（LINKS BI）」は、自然言語による対話でデータ分析・可視化を実現する AI チャット型 BI アプリケーションである。ユーザーは専門的な SQL やプログラミングの知識を必要とせず、自然言語で問い合わせるだけでグラフ・地図・テーブルとして結果を得ることができる。非エンジニアである国交省職員でも、データ分析の成果を直接確認し、ダッシュボードやレポートとして活用できる UI/UX を提供する。

3.ワークショップ・実証実験の実施

実証実験では、実際に国交省の現場の職員を招き、使い方講習会や実演デモ、実際に使ってもらう機会を設けている。これにより、国交省職員がデータ構築モジュール、データ活用モジュールを利用する機会を創出するとともに、リアルなフィードバック

を収集してシステムの本格的な改修につなげている。

1.1.3　開発スケジュール

Project LINKS Veda の開発は 2024 年度業務から開始している。

表 1.1　開発スケジュール

年度	フェーズ	内容
2024 年度	プロトタイプ開発（第 1 期）	LINKS Veda の初期プロトタイプを構築
2025 年度	プロトタイプ開発（第 2 期）	BI の開発、プロトタイプの品質向上・機能拡張・実証実験の実施
2026 年度	プロトタイプ開発（第 3 期）	社会実装に向けた品質向上・本格運用準備（予定）
2027 年度～	本格運用	省内システムとしての本格運用開始（予定）

本書は 2025 年度のプロトタイプ開発（第 2 期）における要件を定義するものである。

1.2. システムを利用する業務全体像とシステム利用フロー

1.2.1　業務フロー

以下の図は、国交省職員が LINKS Veda において、データ構築モジュールおよび LINKS BI を利用してデータ構造化から資料作成・データ公開までを行う業務フローの全体像を示す。

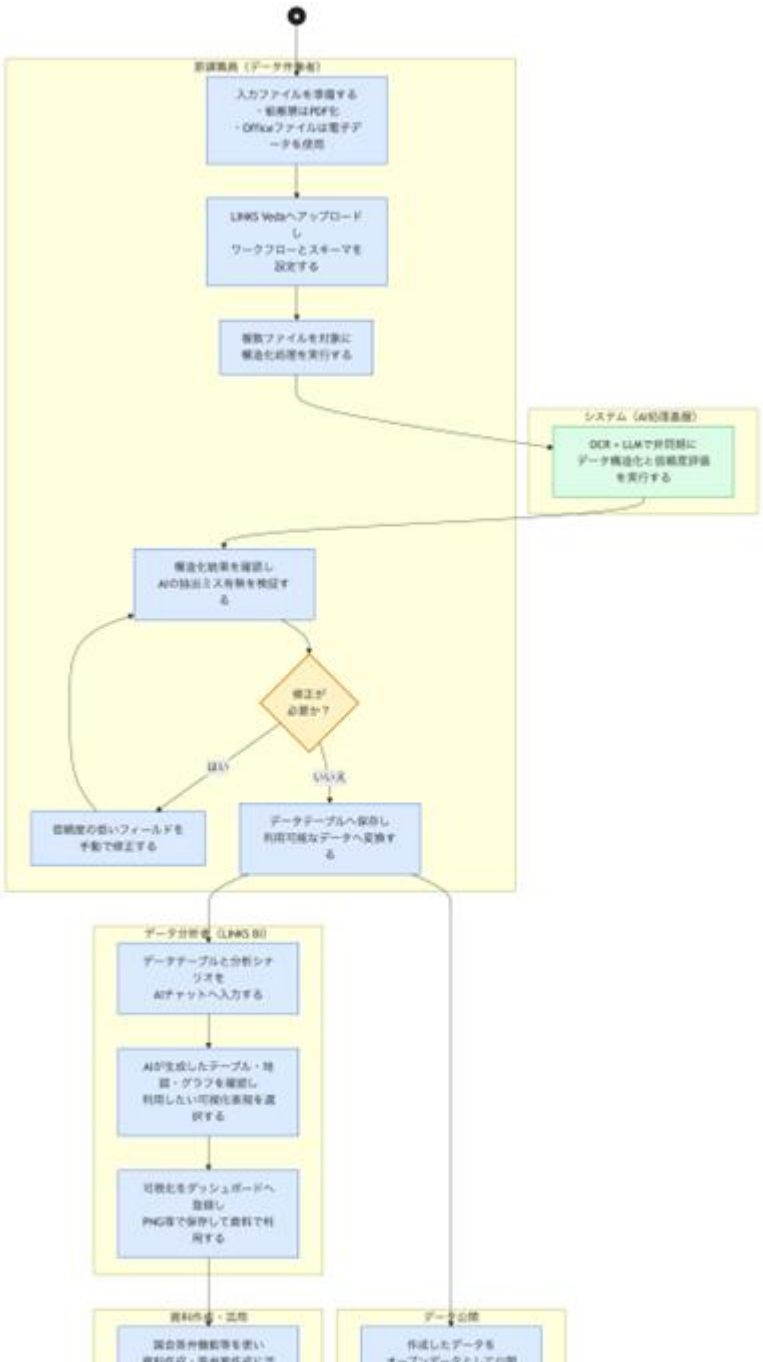

入力ファイルを準備する
・帳票類はPDF化
・Officeファイルは電子データを使用
LINKS Vegaへアップロードし
ワークフローとスキーマを設定する
複数ファイルを対象に構造化処理を実行する
システム（AI処理基盤）
OCR＋LLMで非同期にデータ構造化と信頼度評価を実行する
構造化結果を確認しAIの抽出ミス有無を検証する
修正が必要か？
はい
いいえ
信頼度の低いフィールドを手動で修正する
データテーブルへ保存し利用可能なデータへ変換する
データテーブルと分析シナリオをAIチャットへ入力する
AIが生成したテーブル・地図・グラフを確認し利用したい可視化表現を選択する
可視化をダッシュボードへ登録しPNG等で保存して資料で利用する
作成したデータをオープンデータとして公開

図 1.1 業務フローの全体像

凡例

記号	意味
●	プロセスの開始点
◎	プロセスの終了点
角丸四角（青）	ユーザーが実行する作業
角丸四角（緑）	システムが自動実行する処理
ひし形（黄）	条件分岐

1.2.2　システムシーケンス図

データ構造化（データ構築モジュール）のシステムシーケンス図

以下の図は、ユーザーがファイルをアップロードしてから、AI による構造化処理が完了し、データテーブルとして保存されるまでの内部処理の流れを示す。

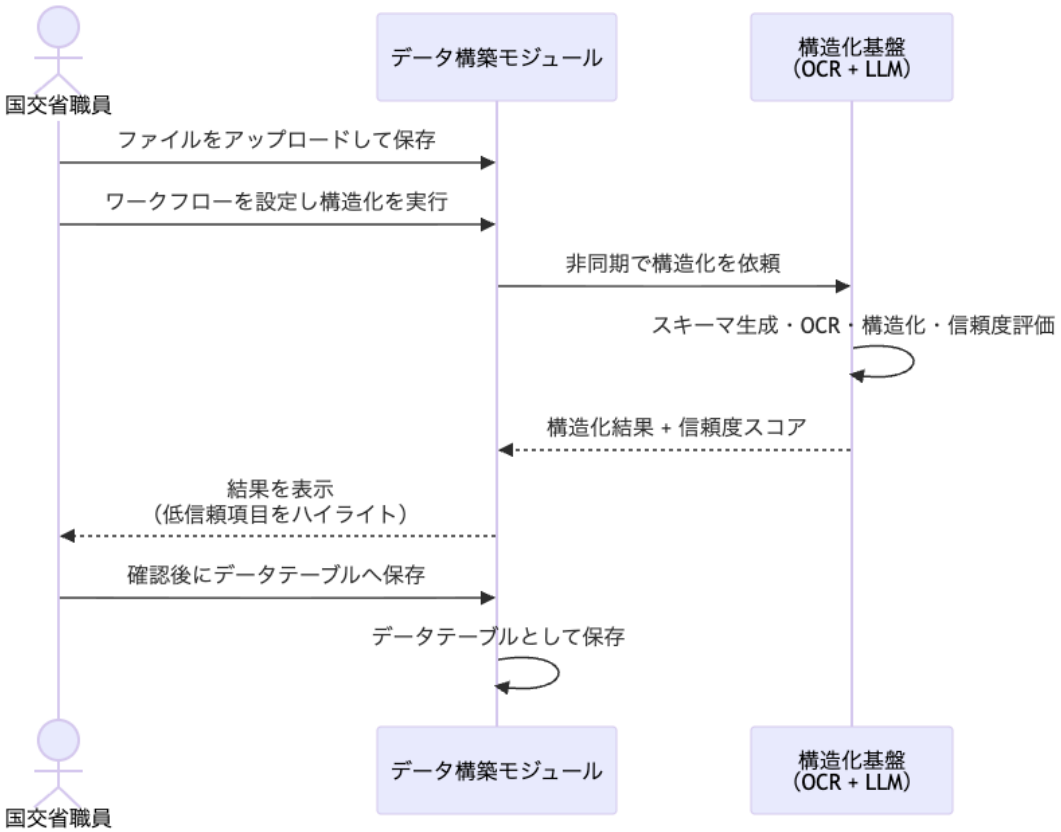


図 1.2 データ構造化（データ構築モジュールのシステムシーケンス図

データ分析・可視化（LINKS BI）のシステムシーケンス図

以下の図は、ユーザーが LINKS BI で自然言語による分析指示を入力してから、AI が可視化結果を生成し、ダッシュボードに保存するまでの処理の流れを示す。

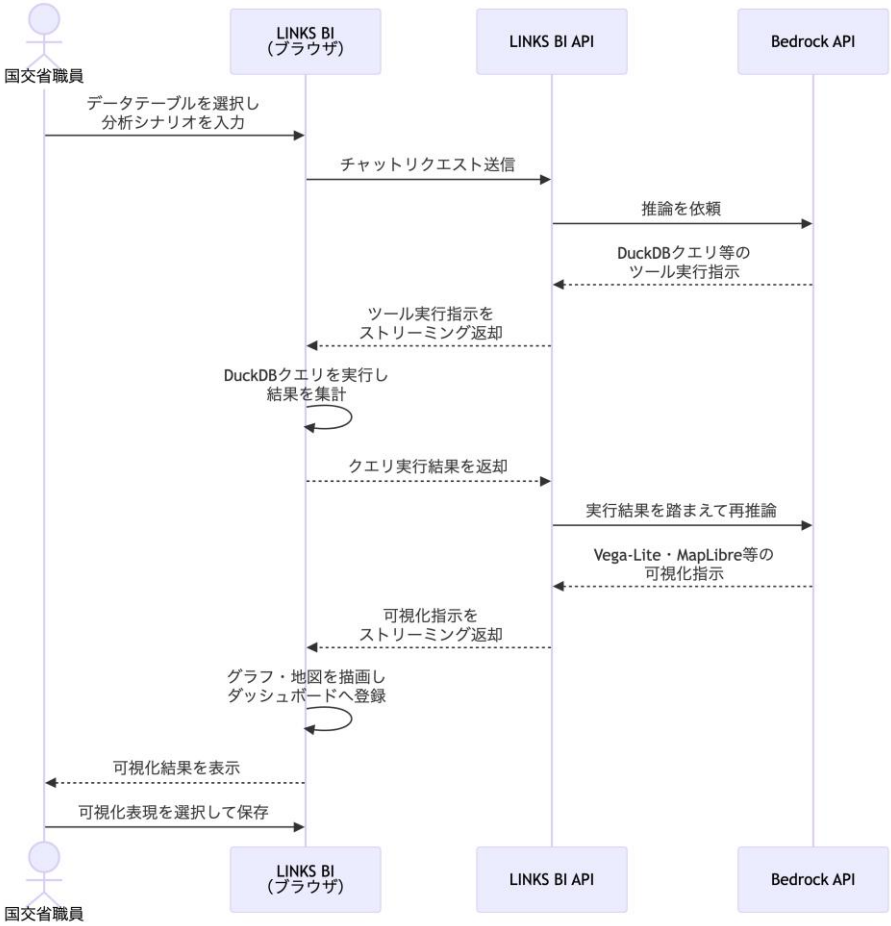


図 1.3 データ分析・可視化（LINKS BI）のシステムシーケンス図

2. 開発するシステム：機能要件

本章で使用する用語の定義は第 5 章の用語集を参照のこと。

2.1. システム機能（FN）

2.1.1　システムアーキテクチャ

● 本システムは、認証・チーム管理を前提に、アセット（原本ファイル）とデータセット／データテーブル（構造化・加工データ）を中核データとして管理し、その上でワークフロー実行基盤（構造化・評価・補正）と各種テーブルアクション（正規化・結合・集計等）を組み合わせてデータ生成から利活用までを一気通貫で提供する構成である。ユーザーは GUI からワークフローを作成・実行し、処理は非同期ジョブとして実行され、結果はデータテーブルとして蓄積される。

● 全体構成は大きく、(1)フロントエンド（GUI）、(2) API／アプリケーション層（認証・権限、チーム、ワークフロー、データ管理）、(3)処理ワーカー層（OCR／VLM／LLM による構造化、不要ページ除去、信頼度評価、バウンディングボックス解決、各種変換・集計）、(4)ストレージ／DB 層（ファイルストレージ、メタデータ、構造化結果データ、検索用ベクトル DB）から構成される。外部サービスとして OCR や LLM 基盤、ジオコーディング等を利用し、データは API を介して各コンポーネント間で授受される。

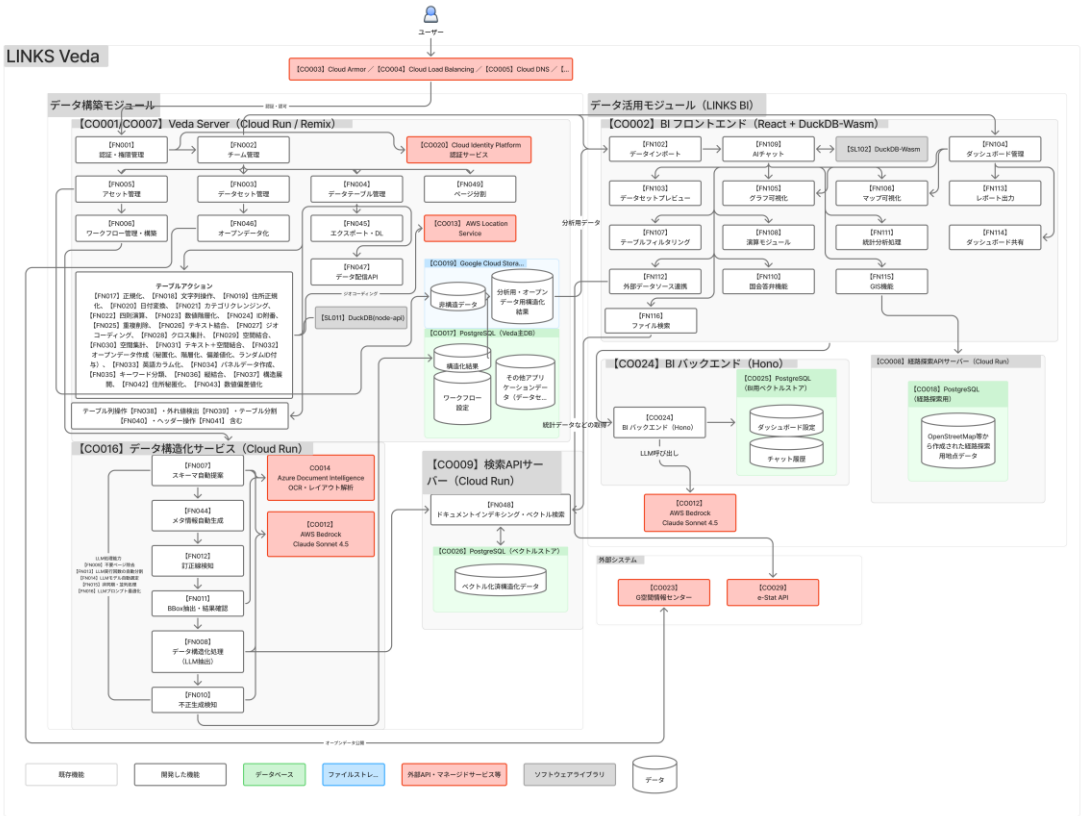


図 2.1 システムアーキテクチャ

2.1.2 システム機能一覧

本節では利用・開発するシステム機能一覧を示し、次節ではその詳細を示す。

本システム設計書では、システムの利用者を以下の 2 種類に分類する。各機能説明において、対象となる利用者を明示する。

- 利用者
  - 職員：　各部門の職員。データの構造化・分析・可視化等の業務を行う。
  - 管理者：　情報政策課の職員。システム全体のアカウント管理・チーム管理等の管理業務を行う。

表 2.1　機能一覧（データ構築モジュール）

ID	機能名	機能説明
FN001	ユーザー認証・権限管理	職員または管理者が、ログイン・セッション管理・チーム内ロール制御を行うことができる（編集者・閲覧者の 2 ロール。管理者による全アカウント管理・パスワードリセットを含む）。
FN002	チーム管理	管理者が、チームの作成・編集・削除・メンバー管理を行うことができる（マルチテナントの単位として機能する。チーム切り替え・権限確認を含む）。
FN003	データセット管理	職員が、データテーブルの一括管理・カラム構造の変更・メタデータの登録編集・オープンデータ公開制御・データ配信を行うことができる（テーブルの複製・テーブルアクション実行基盤・G 空間情報センター連携・メタデータ自動生成を含む）。

FN004	データテーブル管理	職員が、データテーブルの一覧表示・詳細表示・メタデータの編集・カラム構造の編集・CSV・JSON・GeoJSON 形式でのエクスポート・データテーブルの削除・検索を行うことができる。
FN005	アセット管理	職員が、PDF・Excel・CSV・JSON・GeoJSON・Word・PowerPoint 等のファイルをアップロード・閲覧・削除・ダウンロードし、ストレージに保存することができる（複数ファイルの同時アップロードに対応する。ファイル名・ファイルタイプ・ファイルサイズ・作成日時等アセットのメタデータの確認やアセット名での検索に対応する）。
FN006	ワークフロー管理・構築	職員が、データ構造化処理のスキーマ定義・構造化の設定・実行履歴・実行ログ等をワークフローとして一元管理することができる（ワークフローの作成・一覧表示・更新・削除を含む）。

FN007	スキーマ自動提案	職員が、PDF をもとに OCR・LLM による解析で自動提案された出力スキーマ候補を確認・選択することができる（カラム名・データ型・カラムの説明等を含む）。
FN008	データ構造化処理	職員が、PDF・Word・Excel・PowerPoint 等の非構造ファイルから OCR・LLM を用いてスキーマに沿ったデータ抽出を実行することができる（並列・非同期処理、ネスト構造対応、日付クレンジング、実行履歴管理、再実行、ステータス確認を含む）。
FN009	不要ページ除外	職員が、データ構造化の際に、構造化が必要な項目を指定すると、その項目が載っていないページを除外し、構造化対象のページのみを抽出できる。
FN010	不正生成検知	職員が、各フィールドの抽出結果について LLM が評価・推定した不正検知スコアを確認し、スコアの低いフィールドを特定することができる。

FN011	バウンディングボックス抽出・構造化結果確認	職員が、データ構造化の結果から各フィールドが文書のどの位置から取得されたかを UI のハイライト表示で確認することができる（バウンディングボックス座標を用いて特定する）。
FN012	訂正線検知	職員が、訂正線を含む文書に対して VLM と OCR+LLM の両方の構造化結果をマージした高精度な構造化結果を取得することができる（ページ単位で訂正線の有無を自動判定する）。
FN013	LLM 実行回数の自動分割	職員が、トークン上限を超過する大量データに対しても、システムが自動分割した複数回の LLM 実行により、構造化処理を完了することができる。
FN014	LLM モデル自動選定	職員が、データ構造化の内容に応じて、システムが自動選定した最適な LLM モデルで、コストを抑えた構造化処理を利用することができる。
FN015	LLM 実行の非同期・並列処理	職員が、複数件のデータ構造化処理を並列・非同期で同時に実行し、待ち時間を短縮することができる。

FN016	LLM プロンプト最適化	職員が、Few-Shot Learning・Chain of Thought 等の手法で最適化されたプロンプトにより、高精度なデータ構造化結果を得ることができる。
FN017	テキスト正規化（全角/半角統一）	職員が、データテーブルのカラムに対して、英数字・記号・スペースの半角/全角表記を統一することができる（対象カラムと変換方向を指定して実行する）。
FN018	文字カラム操作（テキスト抽出・検索と置換・カラム結合）	職員が、データテーブルのカラムに対して、テキスト抽出・検索と置換・カラム結合の 3 種の文字カラム操作を実行することができる。
FN019	住所文字列の正規化	職員が、データテーブルの住所カラムに対して、日本の住所形式と文字表記を正規化することができる（都道府県名の補完や番地表記の統一等を含む）。
FN020	日付文字列の正規化・データ型変換	職員が、データテーブルの日付カラムに対して、和暦や様々な日付形式を YYYY-MM-DD 等の標準形式に変換することができる（整数・小数・テキスト・日付・日時等へのデータ型変換を含む）。

FN021	カテゴリクレンジング	職員が、データテーブルのカラムに対して、部分一致または類似度を使用してカテゴリ値を標準化することができる（表記ゆれのあるカテゴリ値を正規カテゴリにマッチングし、しきい値未満は null を設定する。都道府県・経営形態のプリセットを備える）。
FN022	四則演算で新しいカラムを作成	職員が、既存の数値カラムに対して、四則演算（+, -, ×, ÷）を行い、結果を新しいカラムとして作成することができる（フィールド・数値・演算子・括弧を組み合わせて式を構築する）。
FN023	数値階層化・数値置換	職員が、データテーブルの数値カラムに対して、指定された数のランク層にデータを分割することができる（自動の等間隔または手動のしきい値指定で分割方法を選択できる。比較条件に基づく数値の置き換えや NULL 置換にも対応する）。

FN024	ID 附番・マルチテーブル ID 付与	職員が、データテーブルのカラムに対して、類似度のしきい値に基づいて類似した値に同じ ID を割り当てることができる（複数カラムの条件を設定できる。複数テーブル間でレコードをマッチングし共通 ID を割り当てる）。
FN025	重複行の削除	職員が、選択したカラムの値が一致する行を重複として削除することができる（残すレコードを最初のレコード・最新/最大・最古/最小から選択できる）。
FN026	テキスト結合	職員が、類似度閾値を使用してテキストカラムをマッチングしテーブルを結合することができる（必須条件・部分条件を設定できる。結合タイプを左外部結合・内部結合・外部結合から選択できる）。
FN027	ジオコーディング	職員が、選択したカラムの住所値を WKT 形式のジオメトリに変換し、新しいジオメトリカラムを作成することができる。

FN028	空間結合	職員が、ジオメトリの交差または最近傍でテーブルを結合することができる（WKT 形式のジオメトリカラムをキーに使用する。最近傍検索では探索半径をメートル単位で指定する）。
FN029	テキスト＋空間結合	職員が、テキストマッチングと半径内の空間最近傍を組み合わせた結合を行うことができる（テキストと位置の両方を考慮した高精度な結合に対応する）。
FN030	オープンデータ作成	職員が、データセットに対して、住所秘匿化・数値階層化・数値偏差値化等、オープンデータ化に必要なテーブルアクションをノーコードで実行することができる。
FN031	カラム名を翻訳する	職員が、データテーブルの日本語カラム名を LLM を使って英語カラム名に変換することができる（オープンデータ化・国際利用での活用を想定する）。

FN032	キーワード分類	職員が、データテーブルのテキストカラムに対して、キーワードを分類・割り当てることができる（手動モードでは事前定義したキーワードパターンに基づきルールベースで分類する。AIモードでは LLM がテキスト内容を分析して適切なキーワードを自動で割り当てる）。
FN033	縦結合	職員が、テーブルを縦に積み重ねることができる（両テーブルのすべてのカラムを保持し、行を垂直に積み重ねる。結合条件はカラム名の完全一致とする）。
FN034	構造展開（テーブルの展開・グループの展開）	職員が、テーブルカラムを複数行に展開することができる（グループカラムを個別のカラムに展開する）。
FN035	テーブル列操作	職員が、カラムの追加（空カラム）・名前変更・複製・削除・単一セル値更新の基本テーブル操作を行うことができる（GUI からのインライン編集に使用する）。
FN036	外れ値フィルター	職員が、数値カラムの値を昇順・降順ソートし、上位・下位の外れ値候補を視覚的に確認することができる。

FN037	フィルター結果の新規テーブル保存	職員が、データテーブルのフィルター機能を使用した状態のテーブルを新規派生テーブルとして保存することができる。
FN038	テーブルヘッダー操作	職員が、データテーブルのカラムヘッダーに対するフィルター・重複行閲覧を行うことができる。
FN039	住所秘匿化	職員が、住所文字列から都道府県または市区町村レベルまでを抽出し、それ以降の詳細住所を除去することができる（秘匿化レベルを都道府県または市区町村から選択できる）。
FN040	数値偏差値化	職員が、数値カラムの偏差値（平均 50・標準偏差 10 の標準化スコア）を算出し、新規カラムとして追加することができる。
FN041	メタ情報自動生成	職員が、スキーマ自動提案で提案されたフィールドの説明文等を活用し、メタ情報の入力を最小限に抑えることができる（フィールドの名称・型・説明文をシステムが自動補完する）。
FN042	データエクスポート・ダウンロード	職員が、データセット・データテーブル・アセットを CSV / JSON / GeoJSON 形式でエクスポート・ダウンロードすることができる。

FN043	オープンデータ化	職員が、G 空間情報センターへのオープンデータ化対応のステータス管理を行うことができる（内部連携や G 空間情報センターとの API 連携により、システムで管理されているメタデータやデータ本体を G 空間情報センターへ連携し、データ公開を行う）。
FN044	データ配信 API	職員が、認証済みアクセストークンによるアクセス制御のもと、特定のチームに所属するデータテーブルを CSV または Parquet 形式で API 経由で取得することができる。
FN045	ドキュメントインデキシング・ベクトル検索	職員が、OCR 処理済みドキュメントに対して、キーワードだけでなく意味的な類似性でも検索することができる（ベクトル検索とキーワード検索を統合したハイブリッド検索を用いる）。
FN046	ページ分割	職員が、1 ファイル内に複数の文書が混在している場合に、ページ数またはキーワードで複数ファイルに分割することができる。

※朱文字：新規開発・既存改修

表 2.2　機能一覧（LINKS BI）

ID	機能名	機能説明
FN101	プロジェクト管理	管理者が、BI のプロジェクトを管理することができる（データ構築モジュールと同じ認証基盤で動作し、同一チーム内でユーザーに割り振られた権限に基づく認可処理を行う）。
FN102	データインポート・データソース管理	職員が、データ構築モジュールで管理されているデータテーブルを選択し、分析対象のデータとしてインポートすることができる（複数ファイルのインポートにも対応する。インポートしたデータソースの一覧管理も行える）。
FN103	データセットプレビュー	職員が、選択したデータセットをテーブル形式や地図等でプレビューすることができる。
FN104	ダッシュボード管理	職員が、グリッドレイアウト上にウィジェット（グラフ・マップ・テキスト・テーブル）を配置・保存することができる。

FN105	グラフ可視化	職員が、棒グラフ・折れ線グラフ・散布図・円グラフ・ヒストグラム・積み上げ棒グラフなど多様なグラフ形式でデータを可視化することができる（色やフォントサイズ等のスタイルを変更することも可能）。
FN106	マップ可視化	職員が、地図ライブラリを利用して、【FN102】でインポートしたデータを地図上に可視化することができる（レイヤーの追加・スタイルの編集・複数レイヤーの重畳表示・凡例の表示・地物クリック時のポップアップ表示・フィルタリングの適用・時系列表示に対応する）。
FN107	テーブルフィルタリング	職員が、テーブルウィジェットに対して、ダッシュボード上でカラムの条件による並び替えや絞り込みを行うことができる。
FN108	演算モジュール	職員が、インポートしたデータセットに対して、カラム同士の四則演算・集計・統計値の算出・空間集計などを、ノーコードで実行することができる。

FN109	AI チャット	職員が、エージェント AI との対話を通して、自然言語でデータの分析・可視化等を実行することができる。
FN110	国会答弁機能	職員が、過去答弁を読み込んだ状態で、自然言語で答弁案生成を依頼し、過去答弁の内容を踏まえた答弁案を作成することができる（答弁案の生成に際しては、過去答弁への参照情報も含めて提示する）。
FN111	統計分析処理	職員が、K-Means クラスタリングや回帰分析等を使った発展的な統計処理・統計解析を実行することができる。
FN112	外部データソース連携	職員が、政府統計 e-Stat 内の人口データとの連携により、外部統計データを取得し分析処理に利用することができる。
FN113	レポート出力	職員が、ダッシュボードをレポートとして PNG 画像形式でダウンロードすることができる。

FN114	ダッシュボード共有	職員が、ダッシュボードを URL で共有することができる（同一チーム内のユーザーには編集権限で、チーム外のユーザーには閲覧権限で URL を共有する）。
FN115	GIS 機能	職員が、自然言語でのチャットを通して、バッファ処理や空間結合等の基本的な GIS 演算処理をノーコードで実行することができる。
FN116	ファイル検索	職員が、自然言語で AI とのチャットを通して検索条件を指定し、データ構築モジュールで構造化された PDF 等の元のアセットを検索することができる。

※朱文字：新規開発・既存改修

2.1.3　システム機能の詳細

以下に、システム機能の詳細を記す。なお、本業務で開発（新規・改修）を行うシステム機能は、機能名称を朱文字で示す。

【FN001】ユーザー認証・権限管理＜既存改修＞

本システム機能の概要

職員または管理者が、ログイン・セッション管理・チーム内ロール制御を行うことができる（編集者・閲覧者の 2 ロール。管理者による全アカウント管理・パスワードリセットを含む）。

本機能では、認証は Google Cloud の Cloud Identity Platform を利用し、メール/パスワード認証および OAuth 認証プロバイダーによるログインをサポートする。認証に成功したユーザーには ID トークン（JWT）が発行され、以降のすべての API リクエストで認証・認可チェックが行われる。データ構築モジュールと LINKS BI は同一の認証基盤を共有しており、データ構築モジュールで発行されたセッションを LINKS BI が検証する委譲方式を採用している。すべての API エンドポイントは、リクエストユーザーのチームメンバーシップとロールを検証し、権限のないリソースへのアクセスを拒否する。

本システム機能の入力・処理・出力のフローチャート

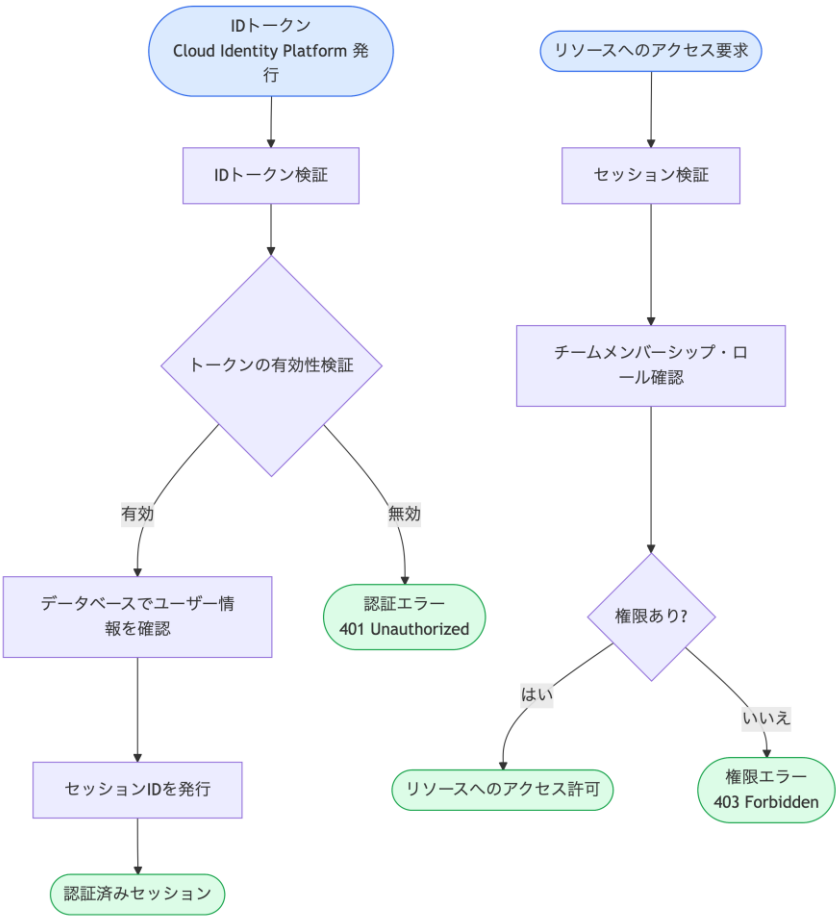


図 2.2 【FN001】のフローチャート

本システム機能の処理の詳細

- **ログイン処理**
  - 処理内容：ユーザーが提供した ID トークンを検証し、トークンの有効期限・署名を確認する。検証後、データベースのテーブルを参照してユーザーレコードが存在しない場合は新規作成し、セッション ID を発行する。
  - 利用するライブラリ：【SL006】Firebase（クライアント）、【SL007】firebase-admin
  - 利用するアルゴリズム：なし

- **権限チェック**
  - 処理内容：各 API エンドポイントでデータベースのテーブルを参照してロールを確認し、操作種別に応じてアクセスを許可または拒否する。管理者のみメンバー管理操作を許可する。編集者は読み書き可能。閲覧者は参照のみ可能。
  - 利用するライブラリ：【SL003】Prisma
  - 利用するアルゴリズム：なし

- **パスワードリセット・アカウント管理**
  - 処理内容：管理者が Cloud Identity Platform を通じて全ユーザーアカウントを管理し、パスワードリセットメールを送信できるようにする。
  - 利用するライブラリ：【SL007】firebase-admin
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**

■ ID トークン（Cloud Identity Platform 発行）

▲ データの形式：JWT

▲ 利用するデータインターフェース：【IF011】Cloud Identity Platform 認証 API、【IF007】PostgreSQL データアクセス。

● **出力**

■ 認証済みセッション（データ構築モジュール / LINKS BI 共通）

▲ データの形式：セッション ID

▲ 利用するデータインターフェース：【IF007】PostgreSQL データアクセス。

【FN002】チーム管理＜新規開発＞

本システム機能の概要

管理者が、チームの作成・編集・削除・メンバー管理を行うことができる（マルチテナントの単位として機能する。チーム切り替え・権限確認を含む）。

本機能では、チームはデータアクセス権限の管理単位であり、すべてのリソース（ファイル・ワークフロー・データセット等）はチームに紐付いて管理される。チームのメンバーにはロール（編集者・閲覧者）が割り当てられ、ロールに応じた操作権限が与えられる。編集者はそのチームにおけるデータの編集権限・登録権限を持つ。閲覧者は閲覧権限のみを持つ。管理者はシステム全体のアカウント管理・メンバー管理の操作権限を持つ。

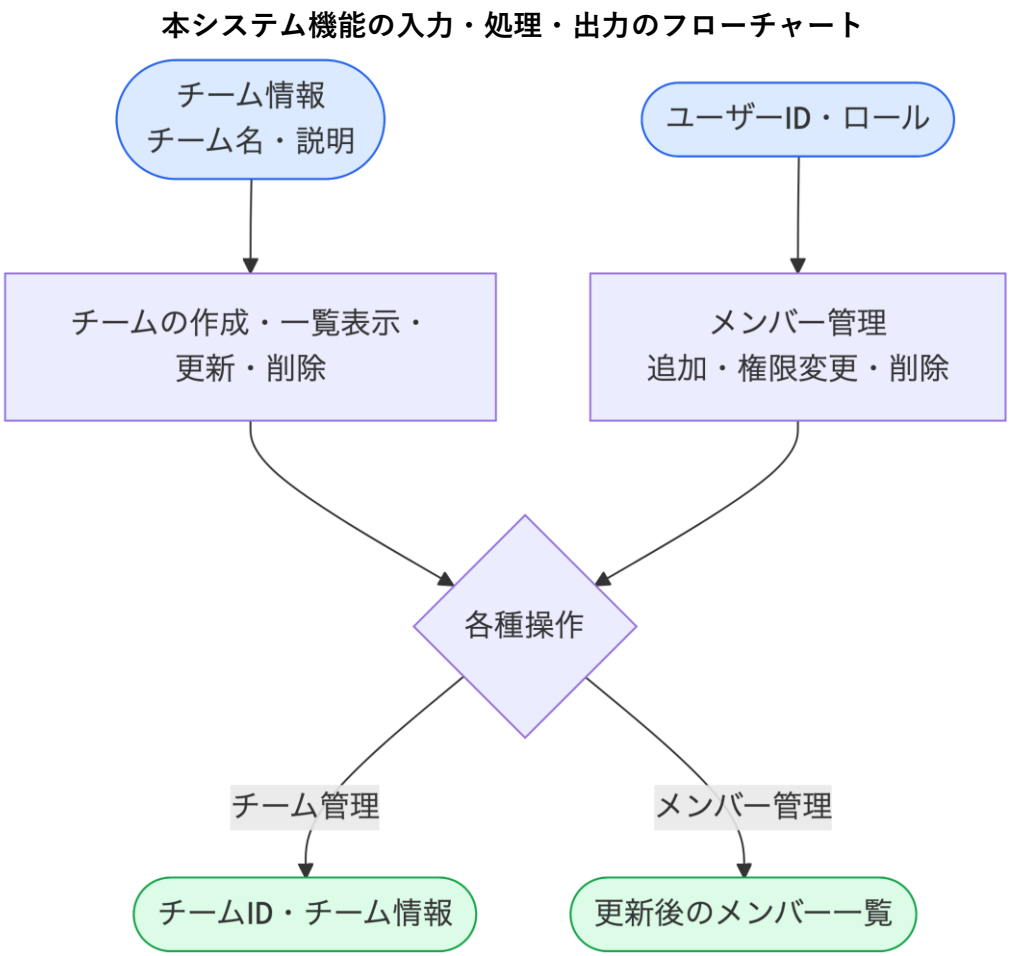


図 2.3 【FN002】のフローチャート

本システム機能の処理の詳細

● **チームの作成・一覧表示・更新・削除**

■ 処理内容：チーム名・説明を指定してチームを作成・更新・削除する。チーム情報はデータベースに保存し、マルチテナント単位として機能する。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

● **メンバー管理**

■ 処理内容：チームにユーザーを追加し、ロール（編集者・閲覧者）を割り当てる。ロールに応じてデータへのアクセス権限が決まる。メンバーの権限変更・削除も行う。認可処理はアプリケーション内で行う。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

● **チーム切り替え**

■ 処理内容：ユーザーが所属する複数チームを切り替えてデータを参照できるようにする。切り替え後は選択チームのデータのみが表示される。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**

■ チーム情報（チーム名・説明）

▲ データの形式：JSON

▲ 利用するデータインターフェース：【IF007】PostgreSQL データアクセス

● **出力**

■ チーム ID・チーム情報

▲ データの形式：JSON（バックエンドからの REST API レスポンス）

▲ 利用するデータインターフェース：【IF007】PostgreSQL データアクセス

【FN003】データセット管理＜既存改修＞

本システム機能の概要

職員が、データテーブルの一括管理・カラム構造の変更・メタデータの登録編集・オープンデータ公開制御・データ配信を行うことができる（テーブルの複製・テーブルアクション実行基盤・G 空間情報センター連携・メタデータ自動生成を含む）。

本機能では、ワークフローによって構造化された結果のデータテーブルや、職員がアップロードした CSV・GeoJSON 等の形式でインポートされたデータテーブルをまとめて管理する。テーブルアクションとは、データテーブルに対してノーコードで実行できるデータ加工・変換処理の総称であり、本機能がその実行基盤を提供する。データセットの作成・一覧表示・更新・削除を行う。また、CSV・JSON 等のファイルをインポートしてデータテーブルとして登録することができる。データセット単位で Parquet 形式によるデータ配信を提供し LINKS BI での利用に対応するほか、CSV 形式で G 空間情報センターへの公開に対応する。

本システム機能の入力・処理・出力のフローチャート

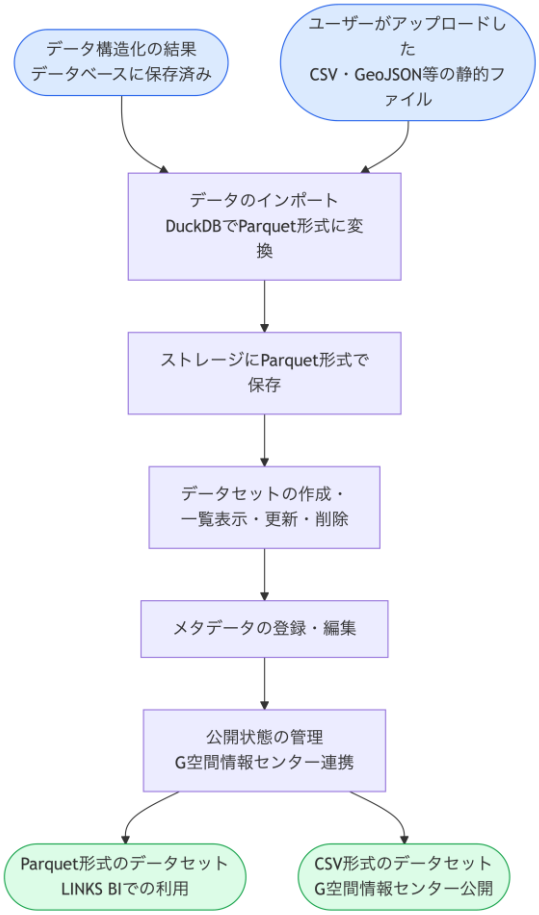


図 2.4 【FN003】のフローチャート

本システム機能の処理の詳細

- **データインポート**
  - ■ 処理内容：データ構造化の結果（データベースに保存済み）またはユーザーがアップロードした CSV・GeoJSON 等の静的ファイルを読み込み、DuckDB を利用して Parquet 形式に変換してストレージに保存する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）、【SL008】@google-cloud/storage
  - ■ 利用するアルゴリズム：なし

- **データセット管理**
  - ■ 処理内容：データセットの作成・一覧表示・更新・削除を行う。複数のデータテーブルをデータセットとしてまとめて管理する。
  - ■ 利用するライブラリ：なし
  - ■ 利用するアルゴリズム：なし

- **メタデータ管理**
  - ■ 処理内容：データセット全体のメタデータおよび各データテーブルに紐づくカラム構造の説明等を登録・編集する。
  - ■ 利用するライブラリ：なし
  - ■ 利用するアルゴリズム：なし

- **公開状態管理・データ配信**

■ 処理内容：データセットのオープンデータ化ステータスを管理し、G 空間情報センターへの公開制御を行う。Parquet 形式でデータ配信を提供する。
■ 利用するライブラリ：【SL008】@google-cloud/storage
■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**
■ データ構造化の結果（データベースに保存された JSON）
▲ データの形式：JSON
■ ユーザーがアップロードした CSV・GeoJSON 等の静的ファイル
▲ データの形式：CSV、GeoJSON、JSON

● **出力**
■ Parquet 形式のデータセット（G 空間情報センター公開・BI 利用）
▲ データの形式：Apache Parquet

【FN004】データテーブル管理＜既存改修＞

本システム機能の概要

職員が、データテーブルの一覧表示・詳細表示・メタデータの編集・カラム構造の編集・CSV・JSON・GeoJSON 形式でのエクスポート・データテーブルの削除・検索を行うことができる。

本システム機能の入力・処理・出力のフローチャート

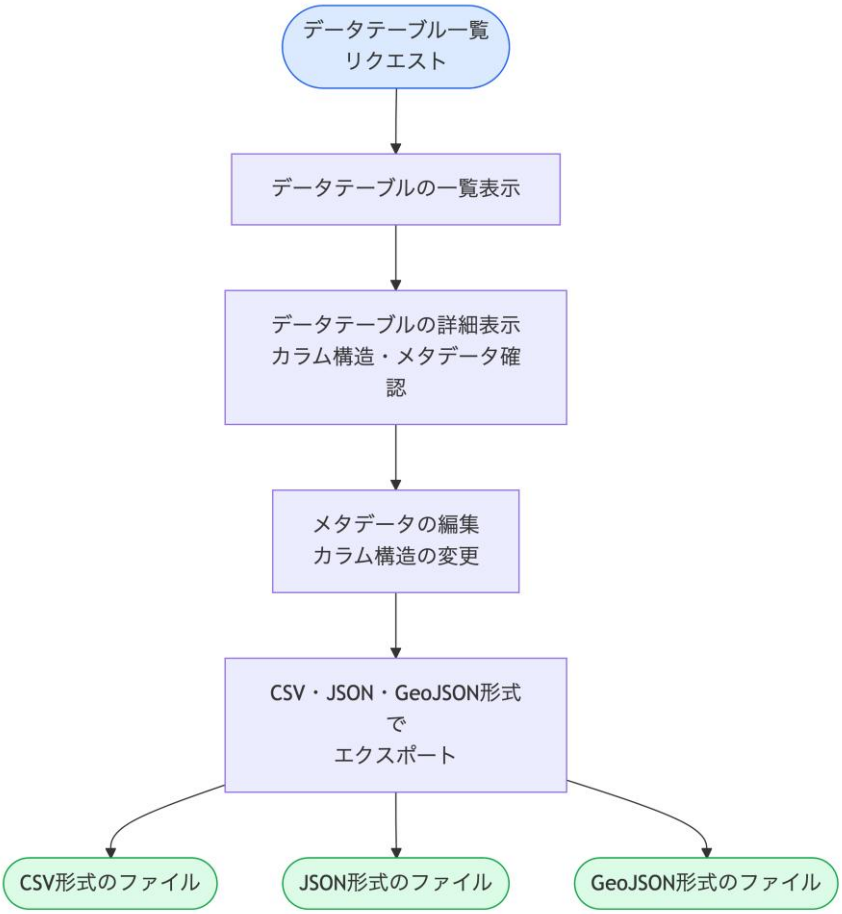


図 2.5 【FN004】のフローチャート

本システム機能の処理の詳細

- **データテーブル一覧表示**
  - 処理内容：データベースに問い合わせ、データテーブルの一覧を取得して表示する。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- **メタデータ・カラム構造の編集**
  - 処理内容：データテーブルのメタデータ（名称・説明等）およびカラム構造（カラムの削除やデータ型の変更等）を編集する。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- **CSV エクスポート**
  - 処理内容：データテーブルを CSV 形式に変換してダウンロードする。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - データテーブル一覧・詳細リクエスト
    - データの形式：JSON
- **出力**

■ CSV 形式のエクスポートファイル

▲ データの形式：CSV

【FN005】アセット管理＜既存改修＞

本システム機能の概要

職員が、PDF・Excel・CSV・JSON・GeoJSON・Word・PowerPoint 等のファイルをアップロード・閲覧・削除・ダウンロードし、ストレージに保存することができる（複数ファイルの同時アップロードに対応する。ファイル名・ファイルタイプ・ファイルサイズ・作成日時等アセットのメタデータの確認やアセット名での検索に対応する）。

本システム機能の入力・処理・出力のフローチャート

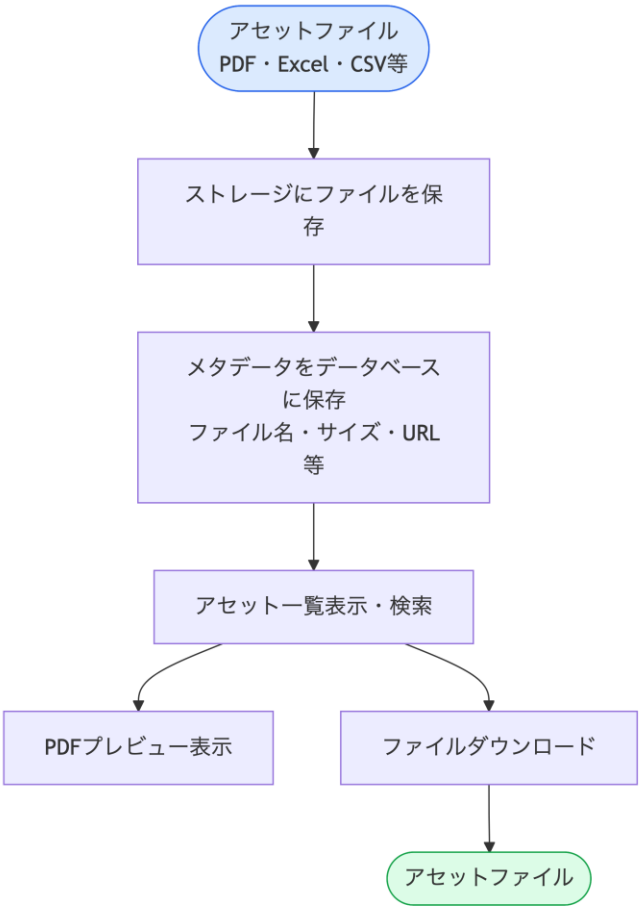


図 2.6 【FN005】のフローチャート

本システム機能の処理の詳細

● **ファイルアップロード**

■ 処理内容：ユーザーがアップロードしたファイルをストレージに保存し、ストレージへの URL 等のメタデータをデータベースに保存する。

■ 利用するライブラリ：【SL008】@google-cloud/storage

■ 利用するアルゴリズム：なし

● **アセット一覧・検索**

■ 処理内容：データベースに問い合わせ、アセットの一覧を取得する。アセット名での部分一致検索に対応する。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

● **プレビュー表示**

■ 処理内容：PDF ファイルのプレビュー表示を行う。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**

■ アセットファイル（PDF・Excel・CSV・JSON・GeoJSON・Word・PowerPoint）

▲ データの形式：各種ファイル形式

● **出力**

■ アセットファイル（ダウンロード）

▲ データの形式：アップロード時と同一形式

【FN006】ワークフロー管理・構築＜既存改修＞

本システム機能の概要

職員が、データ構造化処理のスキーマ定義・構造化の設定・実行履歴・実行ログ等をワークフローとして一元管理することができる（ワークフローの作成・一覧表示・更新・削除を含む）。

本システム機能の入力・処理・出力のフローチャート

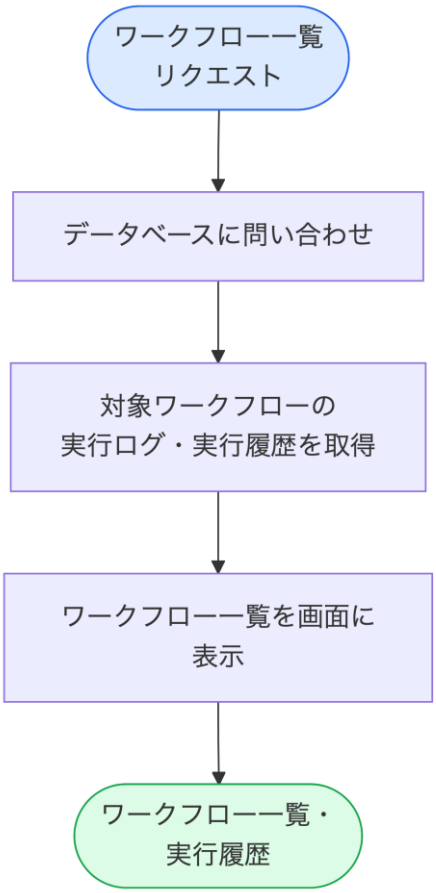


図 2.7 【FN006】のフローチャート

本システム機能の処理の詳細

- **ワークフロー管理**
  - 処理内容：ワークフローの作成・一覧表示・更新・削除を行う。スキーマ定義・構造化の設定を管理する。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- **自動実行**
  - 処理内容：ファイルアップロード時にワークフローが自動起動し、データ構造化処理への非同期リクエストを送信する。ワークフロー実行のステータスは structure → completed / structureError と遷移する。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- **実行履歴・ログ管理**
  - 処理内容：ワークフローの実行履歴・ログを管理する。再実行を可能にする。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ワークフロー設定（スキーマ定義・構造化設定）
    - ▲ データの形式：JSON

- **出力**
  - ■ ワークフロー実行履歴・ログ
    - ▲ データの形式：JSON

【FN007】スキーマ自動提案＜既存改修＞

本システム機能の概要

職員が、PDF をもとに OCR・LLM による解析で自動提案された出力スキーマ候補を確認・選択することができる（カラム名・データ型・カラムの説明等を含む）。
本機能では、処理はデータ構造化サービスとして同期的に実行される。OCR でテキストを抽出した後に LLM がスキーマを推定する（20 ページ/バッチ）。複数バッチにわたって前バッチの結果を累積することで、文書全体を通じた一貫したスキーマを構築する。提案結果はユーザーが確認・編集した上でワークフロー設定として保存する。

本システム機能の入力・処理・出力のフローチャート

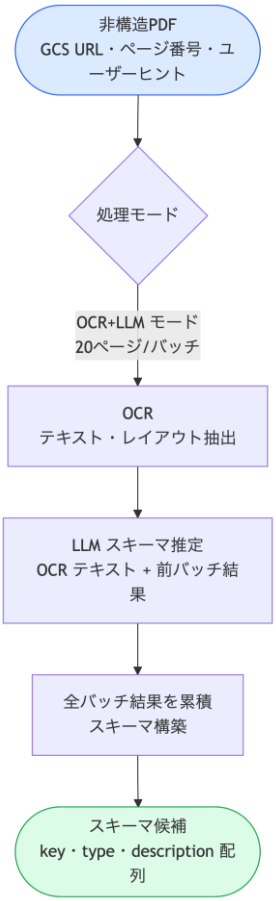


図 2.8 【FN007】のフローチャート

本システム機能の処理の詳細

- **OCR + LLM モード（デフォルト）**
  - ■ 処理内容：OCR でテキスト抽出後、LLM にページテキストを送信してスキーマ候補を生成。20 ページ/バッチで処理。
  - ■ 利用するライブラリ：【SL021】azure-ai-documentintelligence、【SL020】boto3
  - ■ 利用するアルゴリズム：【AL002】スキーマ自動提案アルゴリズム

本システム機能の入出力データの仕様

- **入力**
  - ■ 非構造 PDF（GCS URL・ページ番号・ユーザーヒント）
    - ▲ データの形式：PDF
    - ▲ 利用するデータインターフェース：【IF003】VD010 スキーマ提案 API
- **出力**
  - ■ スキーマ候補（key・type・description・fields 配列）
    - ▲ データの形式：JSON（同期レスポンス）
    - ▲ 利用するデータインターフェース：【IF003】VD010 スキーマ提案 API

【FN008】データ構造化処理＜既存改修＞

本システム機能の概要

職員が、PDF・Word・Excel・PowerPoint 等の非構造ファイルから OCR・LLM を用いてスキーマに沿ったデータ抽出を実行することができる（並列・非同期処理、ネスト構造対応、日付クレンジング、実行履歴管理、再実行、ステータス確認を含む）。

本機能では、OCR の精度を高めるため、前処理として二値化させる。表などが含まれる文書に対しては、ネストされたデータ構造で取得することで複雑なデータ構造も表現できるようにする。構造化結果で曖昧な日付を LLM で高度クレンジング処理で修正する。ファイルごとに実行されるデータ構造化の結果の保存と実行履歴などを管理する。データ構造化処理の再実行を可能にする。実行中のワークフローの処理ステータス（実行中・失敗・完了等）を確認できる。詳細な構造化アルゴリズムは【AL001】OCR + LLM データ構造化アルゴリズムを参照。

本システム機能の入力・処理・出力のフローチャート

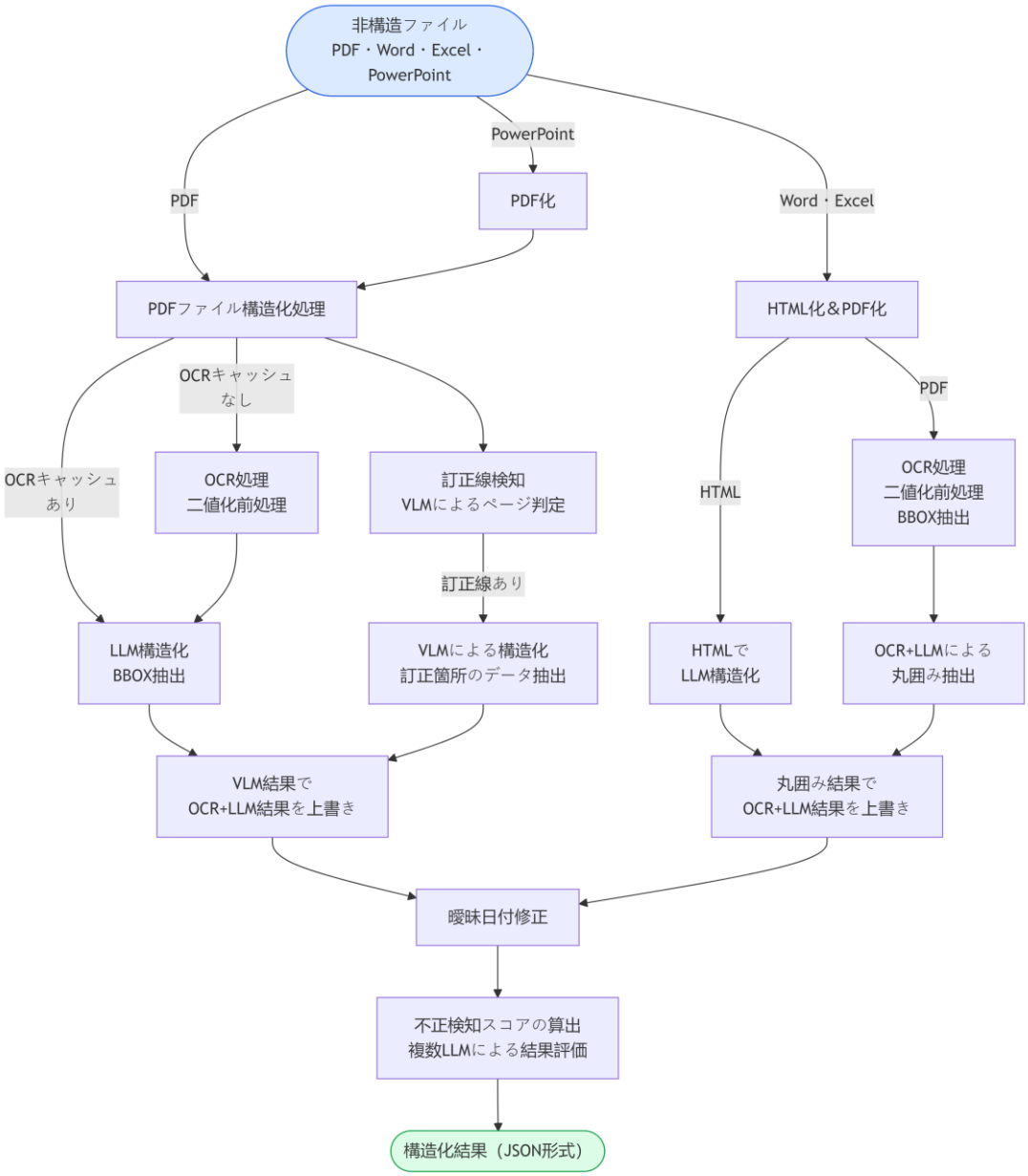

非構造ファイル
PDF・Word・Excel・
PowerPoint
PDF
PowerPoint
Word・Excel
PDF化
PDFファイル構造化処理
HTML化＆PDF化
OCRキャッシュ
なし
OCRキャッシュ
あり
OCR処理
二値化前処理
訂正線検知
VLMによるページ判定
HTML
PDF
OCR処理
二値化前処理
BBOX抽出
訂正線あり
LLM構造化
BBOX抽出
VLMによる構造化
訂正箇所のデータ抽出
HTMLで
LLM構造化
OCR+LLMによる
丸囲み抽出
VLM結果で
OCR+LLM結果を上書き
丸囲み結果で
OCR+LLM結果を上書き
曖昧日付修正
不正検知スコアの算出
複数LLMによる結果評価
構造化結果（JSON形式）

図 2.9 【FN008】のフローチャート

本システム機能の処理の詳細

- **ファイル取得・形式判定**
  - 処理内容：ストレージからファイルを取得し、ファイル形式（PDF / Word・PowerPoint / Excel）を判定する。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- 訂正線検知・修正実行
  - 処理内容：PDF ファイルに対して【FN012】訂正線検知を実行し、ページ単位で訂正線の有無を判定する。訂正線が検出された場合、VLM を利用して訂正箇所のデータ構造化を行い、その結果で OCR+LLM による構造化結果を上書きして精度の高い構造化結果を生成する。詳細は【FN012】訂正線検知を参照。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

- **OCR 処理**
  - 処理内容：PDF ファイルのテキスト・レイアウト・バウンディングボックスを抽出する。精度を高めるため、PDF を前処理として二値化する。OCR 結果はキャッシュされ、同一ファイルの再処理時にはキャッシュを利用する。
  - 利用するライブラリ：【SL021】azure-ai-documentintelligence、【SL034】GhostScript
  - 利用するアルゴリズム：【AL001】OCR + LLM データ構造化アルゴリズム

● **Office ファイル変換**

■ 処理内容：PowerPoint ファイルは PDF 化して PDF 処理フローに合流する。HTML 化し、LLM に直接入力して構造化する。Excel・Word ファイルは PowerPoint と同じ PDF 化処理で丸囲み抽出するために PDF 化して構造化させる。

■ 利用するライブラリ：【SL033】LibreOffice

■ 利用するアルゴリズム：なし

● **LLM による構造化**

■ 処理内容：OCR 結果または HTML を LLM に送信し、定義されたスキーマに沿って構造化データを抽出する。非同期コールバック方式で結果を返却する。詳細なアルゴリズムは【AL001】を参照。

■ 利用するライブラリ：【SL020】boto3、【SL027】langchain-aws / langchain-core

■ 利用するアルゴリズム：【AL001】OCR + LLM データ構造化アルゴリズム

● **曖昧日付修正**

■ 処理内容：構造化結果に曖昧な日付がある場合、LLM で修正できる機能である。

■ 利用するライブラリ：【SL020】boto3、【SL027】langchain-aws / langchain-core

■ 利用するアルゴリズム：なし

● **不正検知**

■ 処理内容：構造化結果の各フィールドに対して複数の LLM で評価を実施し、不正検知スコアを算出する。詳細は【AL004】を参照。

■ 利用するライブラリ：【SL020】boto3、【SL027】langchain-aws / langchain-core

■ 利用するアルゴリズム：【AL004】不正検知アルゴリズム

本システム機能の入出力データの仕様

● **入力**

■ 非構造ファイル（PDF・Word・Excel・PowerPoint）、スキーマ定義

▲ データの形式：各種ファイル形式、JSON

● **出力**

■ 構造化結果（各フィールドのキー・値・不正検知スコア・バウンディングボックス座標）

▲ データの形式：JSON

【FN009】不要ページ除外＜新規開発＞

本システム機能の概要

職員が、データ構造化の際に、構造化が必要な項目を指定すると、その項目が載っていないページを除外し、構造化対象のページのみを抽出できる。

本システム機能の入力・処理・出力のフローチャート

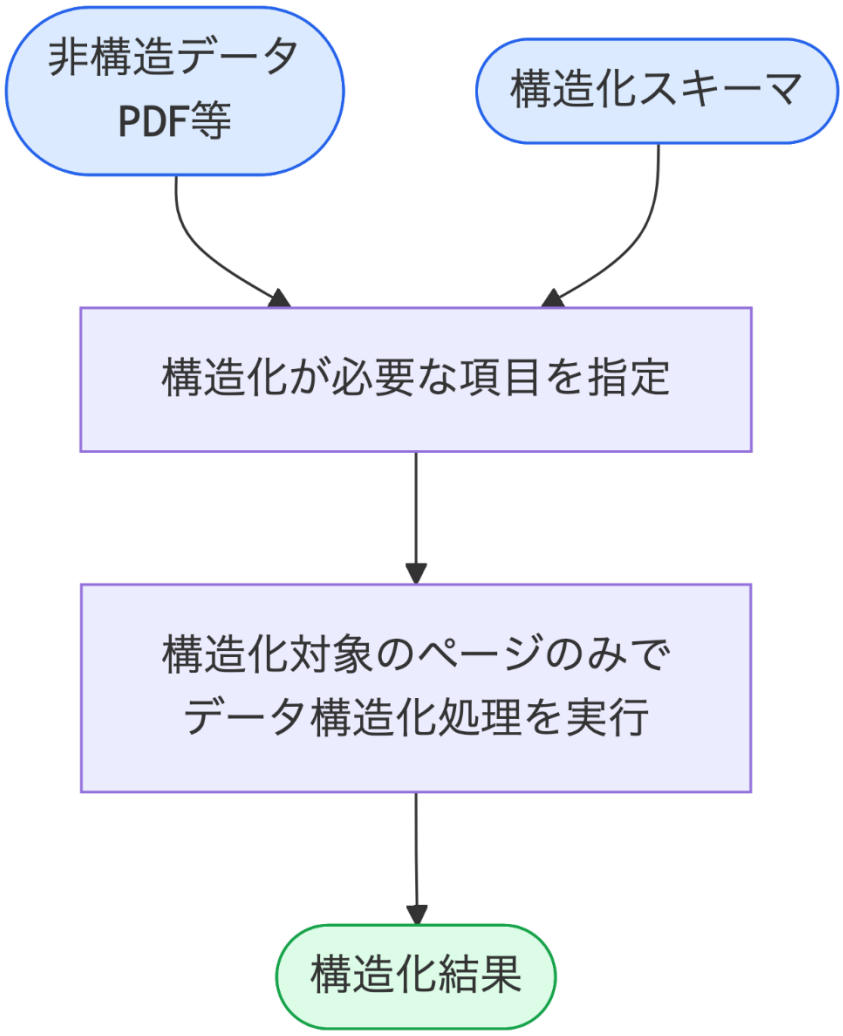

非構造データ
PDF等
構造化スキーマ
構造化が必要な項目を指定
構造化対象のページのみで
データ構造化処理を実行
構造化結果

図 2.10 【FN009】のフローチャート

本システム機能の処理の詳細

- **不要ページ・不要コンテンツの除外**
  - 処理内容：非構造データ（PDF 等）と構造化スキーマを入力として受け取り、スキーマに関係のないページや不要なコンテンツを構造化対象外とする。構造化に必要なページのみを抽出し、後段のデータ構造化処理（【FN008】）へ渡す。
  - 利用するライブラリ：なし
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - 非構造データ（PDF 等）
    - データの形式：各種ファイル形式
    - 利用するデータインターフェース：【IF001】GCS ファイルストレージインターフェース
  - 構造化スキーマ
    - データの形式：JSON
- **出力**
  - 構造化結果
    - データの形式：JSON
    - 利用するデータインターフェース：【IF004】WorkflowRun コールバック API

【FN010】不正生成検知＜新規開発＞

本システム機能の概要

職員が、各フィールドの抽出結果について LLM が評価・推定した不正検知スコアを確認し、スコアの低いフィールドを特定することができる。
本機能では、各フィールドの抽出結果に対して 0.0〜1.0 の不正検知スコアを付与する。ハルシネーション等の不正生成を検知し、GUI で確認できるようにする。

本システム機能の入力・処理・出力のフローチャート

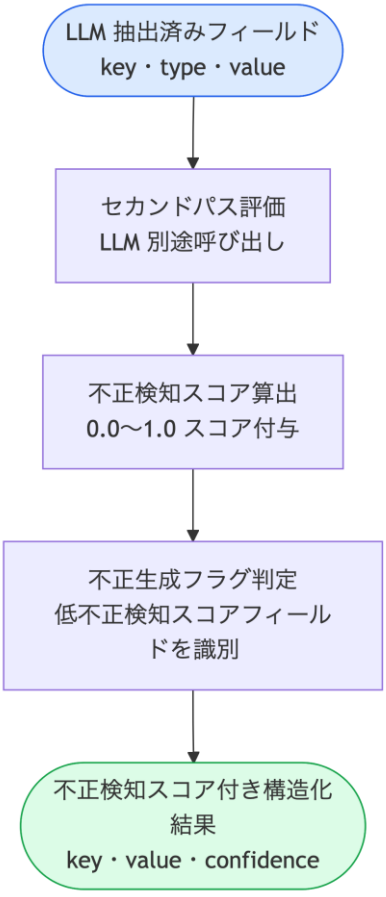


図 2.11　【FN010】のフローチャート

本システム機能の処理の詳細

- **不正検知（セカンドパス評価）**
  - ■ 処理内容：LLM で抽出した各フィールドに対して、LLM による別途の評価呼び出しを実行し、各フィールドの不正検知スコア（0.0〜1.0）を付与する。
  - ■ 利用するアルゴリズム：【AL004】不正検知アルゴリズム

- **不正生成検知**
  - ■ 処理内容：不正検知スコアが低いフィールドをハルシネーション等の不正生成の疑いがあるものとしてフラグを立て、GUI 上でユーザーが確認・修正できるようにする。
  - ■ 利用するアルゴリズム：【AL004】不正検知アルゴリズム

本システム機能の入出力データの仕様

- **入力**
  - ■ LLM 抽出済みフィールドリスト（key・type・value）
    - ▲ データの形式：JSON
    - ▲ 利用するデータインターフェース：【IF002】VD009 構造化処理 API
- **出力**
  - ■ 不正生成スコア付き構造化結果（key・value・confidence）
    - ▲ データの形式：JSON
    - ▲ 利用するデータインターフェース：【IF004】WorkflowRun コールバック API

【FN011】バウンディングボックス抽出・構造化結果確認＜新規開発＞

本システム機能の概要

職員が、データ構造化の結果から各フィールドが文書のどの位置から取得されたかを UI のハイライト表示で確認することができる（バウンディングボックス座標を用いて特定する）。

本機能では、構造化抽出結果の各フィールドが文書のどの位置（バウンディングボックス座標）から取得されたかを特定し、UI 上でフィールド値と対応する文書箇所をハイライト表示する。

本システム機能の入力・処理・出力のフローチャート

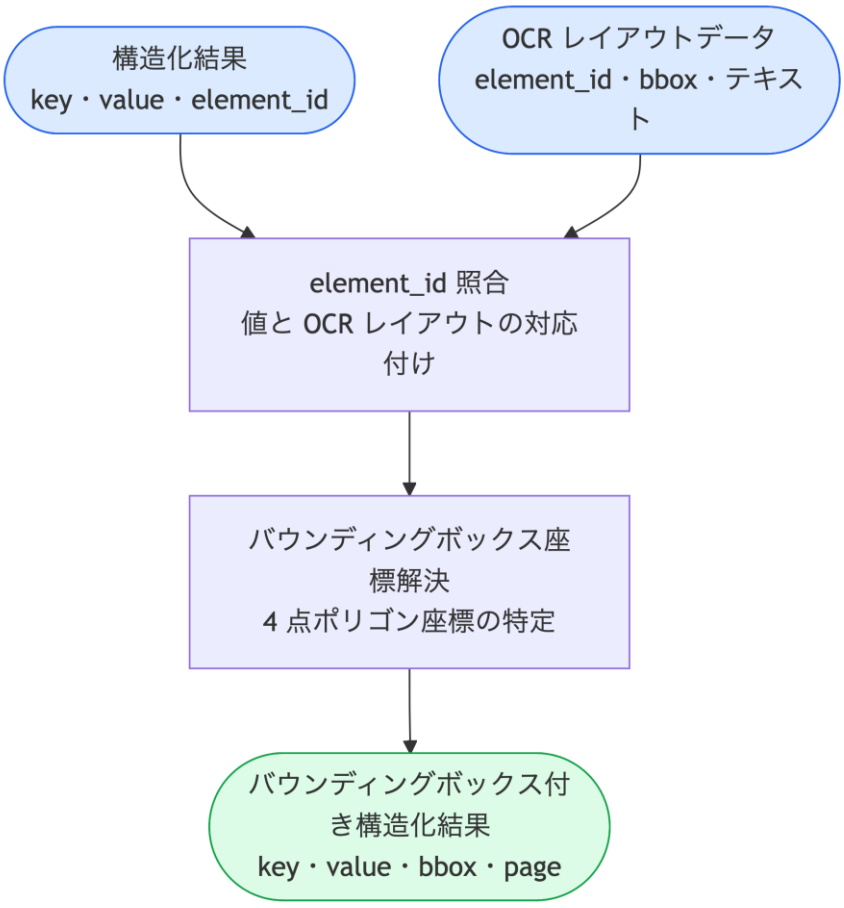


図 2.12　【FN011】のフローチャート

本システム機能の処理の詳細

● **バウンディングボックス抽出**

■ 処理内容：LLM 抽出結果と OCR レイアウト情報の element_id を照合し、各フィールド値が文書上のどの位置（バウンディングボックス座標：4 点ポリゴン形式）に存在するかを特定する。プリミティブ型はテキスト行の element_id と、テーブル型はテーブルセルの element_id とマッチングする。

■ 利用するアルゴリズム：【AL003】バウンディングボックス抽出アルゴリズム

● **構造化結果確認 UI**

■ 処理内容：抽出されたフィールド値と バウンディングボックス座標をもとに、文書 PDF 上の対応箇所をハイライト表示する。ユーザーが各フィールドの根拠箇所を視覚的に確認できるようにする。

■ 利用するライブラリ：react-pdf（9.1.1）/ pdfjs-dist（4.4.168）

本システム機能の入出力データの仕様

● **入力**

■ 構造化抽出結果（key・value・element_id）・OCR レイアウトデータ（element_id・bbox・テキスト）

▲ データの形式：JSON

● **出力**

■ バウンディングボックス付き構造化結果（key・value・bbox・page）

▲ データの形式：JSON

【FN012】訂正線検知＜新規開発＞

本システム機能の概要

職員が、訂正線を含む文書に対して VLM と OCR+LLM の両方の構造化結果をマージした高精度な構造化結果を取得することができる（ページ単位で訂正線の有無を自動判定する）。

本機能では、訂正線が検出された場合、VLM による構造化結果で OCR + LLM による構造化結果を上書きし、不正生成スコアの高い構造化結果を生成する。訂正の有無を示す 「is_edited」 フラグを構造化結果の各フィールド一つ一つに付与する。

本システム機能の入力・処理・出力のフローチャート

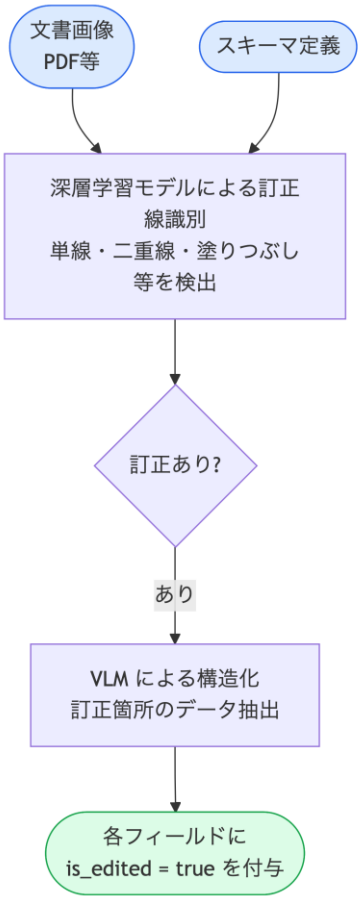


図 2.13　【FN012】のフローチャート

本システム機能の処理の詳細

● **訂正線検出**

■ 処理内容：文書画像において 【SL032】azure-cognitiveservices-vision-customvision が単線・二重線・三重線・塗りつぶし背景等の訂正線パターンを識別し、ページ単位で訂正の有無を判定する。

■ 利用するアルゴリズム：【AL010】訂正線の検知アルゴリズム

● **VLM による構造化・結果マージ**

■ 処理内容：訂正線が検出された場合、VLM を利用して訂正箇所のデータ構造化を行う。VLM での抽出結果を OCR + LLM での構造化結果に上書きすることで、不正生成スコアの高い構造化結果を生成する。

■ 利用するアルゴリズム：【AL010】訂正線の検知アルゴリズム

● **is_edited フラグ付与**

■ 処理内容：構造化結果の各フィールド一つ一つに対して、訂正が存在するフィールドには 「is_edited = true」 を付与し、OCR+LLM 構造化結果に上書きする。

■ 利用するアルゴリズム：【AL010】訂正線の検知アルゴリズム

本システム機能の入出力データの仕様

● **入力**

■ 文書画像（PDF 等）・スキーマ定義

▲ データの形式：バイナリ（PDF 等）、JSON

▲ 利用するデータインターフェース：【IF001】GCS ファイルストレージインターフェース、【IF002】VD009 構造化処理 API

● **出力**

■ is_edited フラグ付き構造化抽出結果（key・value・is_edited・confidence）

▲ データの形式：JSON

【FN013】LLM 実行回数の自動分割＜新規開発＞

本システム機能の概要

職員が、トークン上限を超過する大量データに対しても、システムが自動分割した複数回の LLM 実行により、構造化処理を完了することができる。

本機能では、LLM のトークン上限を考慮し、大量ページを含むアセットのデータ構造化処理を複数の処理単位に自動分割している。トークン上限を超過しそうな場合に複数回の LLM 実行によって処理する。

本システム機能の入力・処理・出力のフローチャート

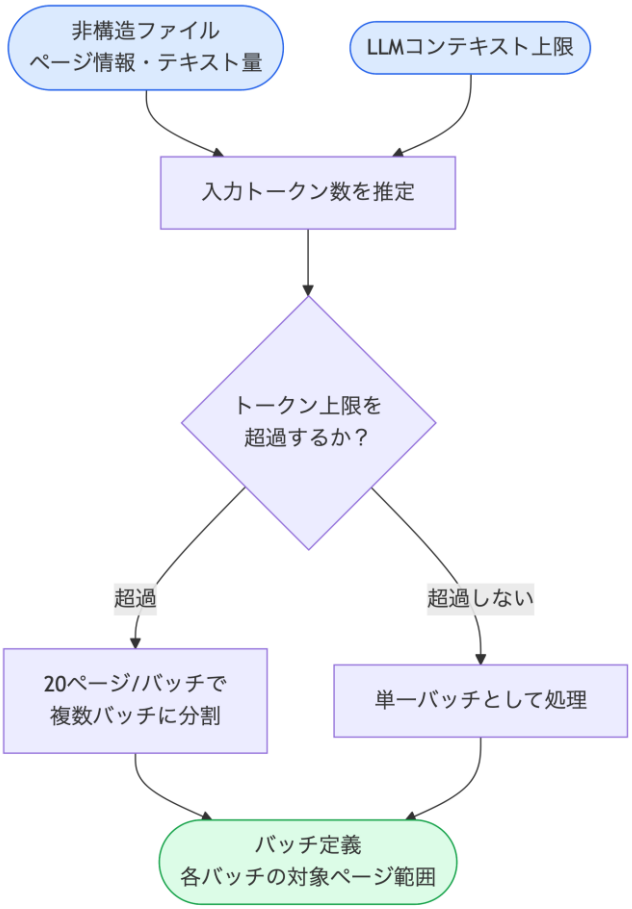


図 2.14　【FN013】のフローチャート

本システム機能の処理の詳細

● **トークン数推定**

■ 処理内容：アセットのページ数・テキスト量から LLM への入力トークン数を推定する。

■ 利用するライブラリ：【SL026】tiktoken

■ 利用するアルゴリズム：なし

● **バッチ分割**

■ 処理内容：推定トークン数が LLM のコンテキスト上限（180,000 トークン、日本語で A4 の 20 枚程度）を超過する場合、20 ページ/バッチを基本単位としてページ群を複数バッチに分割する。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**

■ 非構造ファイル（PDF・Word・Excel・PowerPoint）、スキーマ定義

▲ データの形式：各種ファイル形式、JSON

● **出力**

■ 構造化結果（各フィールドのキー・値・不正検知スコア・バウンディングボックス座標）

▲ データの形式：JSON

【FN014】LLM モデル自動選定＜新規開発＞

本システム機能の概要

職員が、データ構造化の内容に応じて、システムが自動選定した最適な LLM モデルで、コストを抑えた構造化処理を利用することができる。
本機能では、膨大な入力データをデータ構造化していくにあたり、入力データに適したスペックの生成 AI モデルを選定することで、データ構造化に係るコストを削減する。

本システム機能の入力・処理・出力のフローチャート

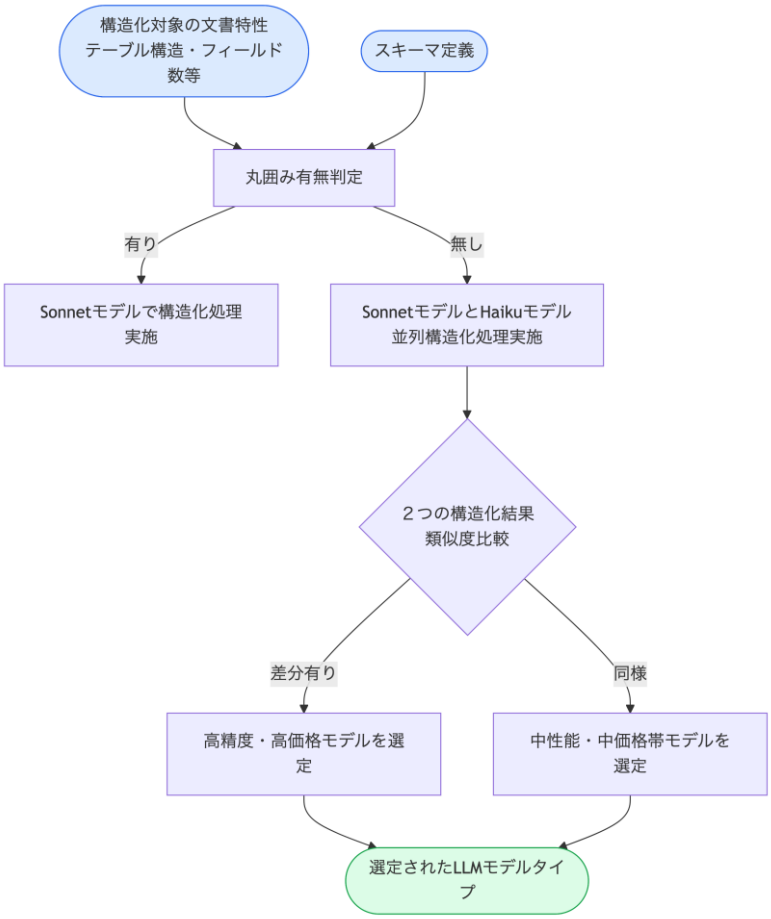


図 2.15 【FN014】のフローチャート

本システム機能の処理の詳細

● **モデル選定判定**

■ 処理内容：

▲ 本機能はデータ構造化の本処理が行われる前段で実施し、1 つの入力ファイルに対して下位モデルと上位モデルを同条件で実行したうえで、両モデルの出力 JSON を完全一致（Exact Matching）により比較し、「上位モデルの結果が下位モデルと実質同等か」を機械的に判定して採用モデルを決定する。

▲ 同条件実行の前提として、入力には同一の OCR テキストを使用し、指示には同一の出力スキーマ、構造化プロンプトを使用する。temperature や max_tokens 等の推論パラメータも固定する。

▲ 処理手順としては、まず入力ファイル（PDF 等）から取得した OCR テキストを正規化し、改行や連続空白の整形、ページ区切りの付与などを行う。次に、正規化済み OCR テキストをプロンプトテンプレートに埋め込み、下位モデルと上位モデルで完全に同一の入力（prompt）になるようにプロンプトを組み立て、それぞれ構造化を実行し、構造化結果を取得する。その後、Exact Matching による比較に進み、構造化結果（文字列、数値、真偽値、null）が完全に一致する場合に一致と判定する。実装上はキー順序を固定した正規化済みのシリアライズ表現を生成し、その文字列表現同士を比較することで一致判定を行う。差分がある場合は、性能差があるため、品質を優先し、上位モデルを採用する。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：【AL011】LLM モデルの自動選定

本システム機能の入出力データの仕様

- **入力**
  - ■ 構造化対象の文書特性・スキーマ定義
    - ▲ データの形式：JSON
- **出力**
  - ■ 選定された LLM モデル ID
    - ▲ データの形式：文字列

【FN015】LLM 実行の非同期・並列処理＜新規開発＞

本システム機能の概要

職員が、複数件のデータ構造化処理を並列・非同期で同時に実行し、待ち時間を短縮することができる。

本機能では、データ構造化を実施するにあたり、各 LLM が相互に依存しない単独実行の構成とする。それぞれの LLM 処理が並列実行できるようにし、システム全体のスループットおよび処理効率を向上する。

本システム機能の入力・処理・出力のフローチャート

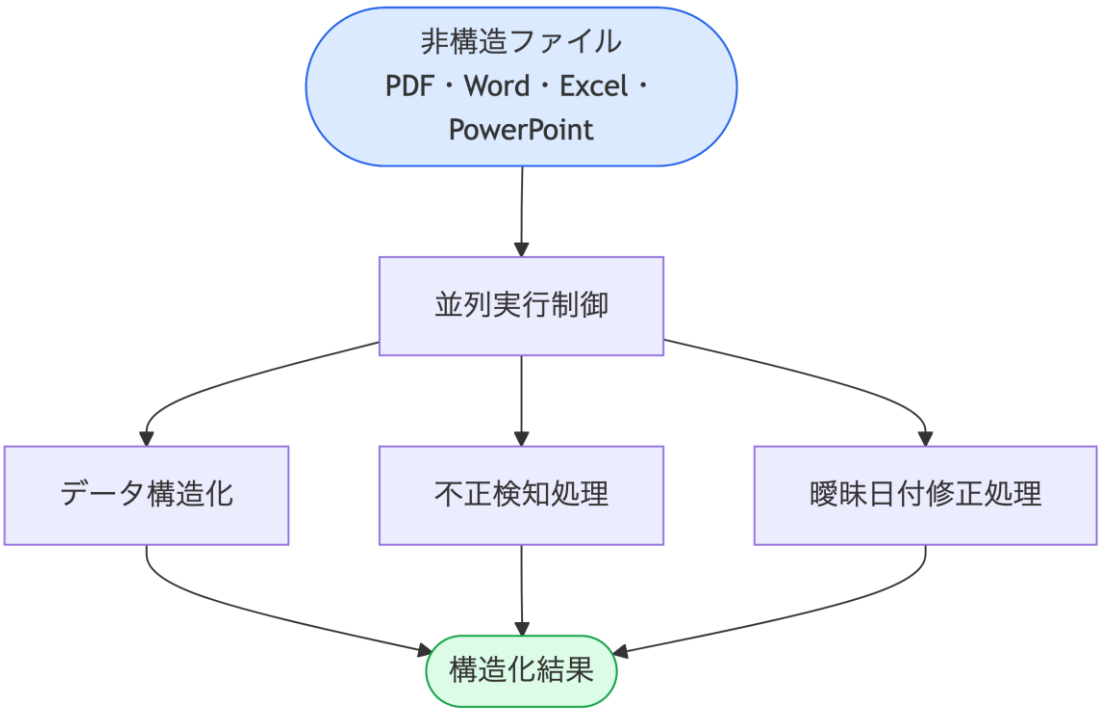


図 2.16　【FN015】のフローチャート

本システム機能の処理の詳細

- **並列実行制御**
  - ■ 処理内容：各 LLM 処理を並列的に単独実行し、構造化機能全体として処理時間が短縮できるように設計する。
  - ■ 利用するライブラリ：なし
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 非構造ファイル（PDF・Word・Excel・PowerPoint）、スキーマ定義
    - ▲ データの形式：各種ファイル形式、JSON
- **出力**
  - ■ 構造化結果（各フィールドのキー・値・不正検知スコア・バウンディングボックス座標）
    - ▲ データの形式：JSON

【FN016】LLM プロンプト最適化＜新規開発＞

本システム機能の概要

職員が、Few-Shot Learning・Chain of Thought 等の手法で最適化されたプロンプトにより、高精度なデータ構造化結果を得ることができる。

本機能では、Few-Shot Learning・Chain of Thought 等の複数のプロンプトエンジニアリング手法を用いて、LLM へのプロンプトを最適化することでデータ構造化の抽出精度を改善している。

本システム機能の入力・処理・出力のフローチャート

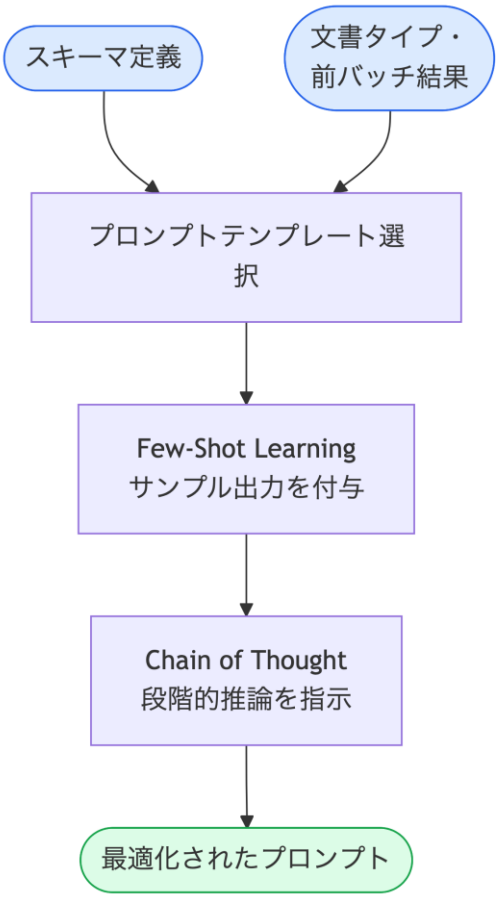


図 2.17　【FN016】のフローチャート

本システム機能の処理の詳細

● **プロンプト構築**

■ 処理内容：スキーマ定義・文書タイプ・前バッチの結果等に基づき、最適なプロンプトを動的に構築する。Few-Shot Learning ではサンプル出力を含め、Chain of Thought では段階的推論を指示する。

■ 利用するライブラリ：なし

■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

● **入力**

■ 非構造ファイル（PDF・Word・Excel・PowerPoint）、スキーマ定義

▲ データの形式：各種ファイル形式、JSON

● **出力**

■ 最適化されたプロンプト

▲ データの形式：テキスト

【FN017】テキスト正規化（全角/半角統一）＜新規開発＞

本システム機能の概要

職員が、データテーブルのカラムに対して、英数字・記号・スペースの半角/全角表記を統一することができる（対象カラムと変換方向を指定して実行する）。
本機能では、DuckDB の文字列変換関数を 5 段階で連鎖適用し、英大文字・英小文字・数字・空白・記号の各文字グループを個別に変換する。

本システム機能のフローチャート

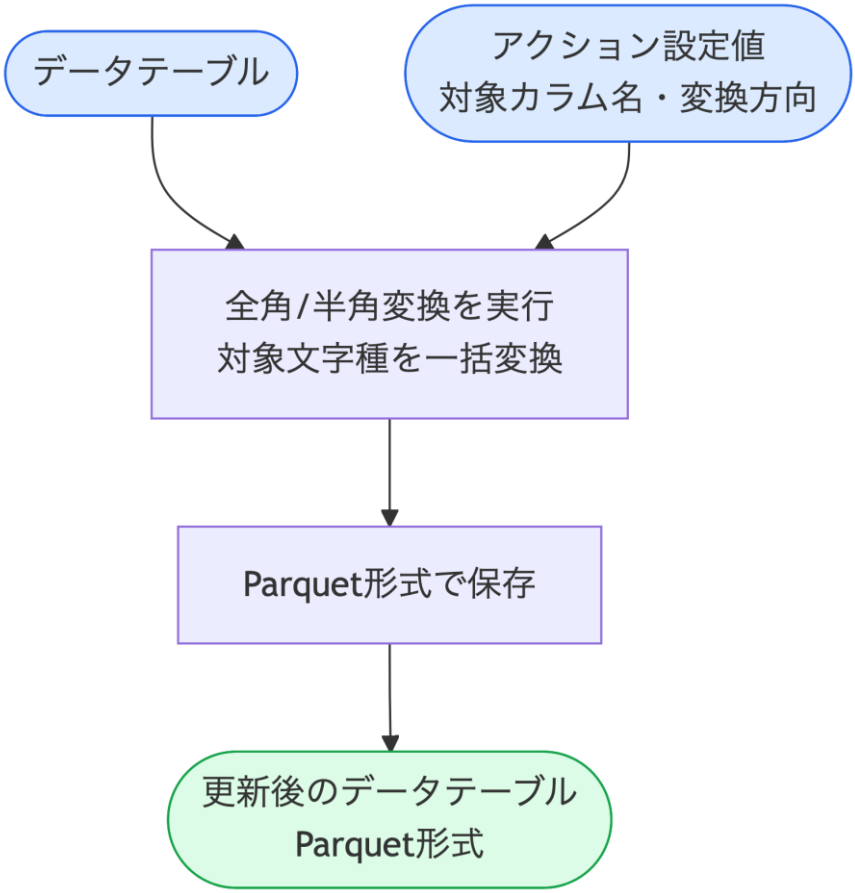


図 2.18　【FN017】のフローチャート

本システム機能の処理の詳細

- **全角/半角変換処理**
  - 処理内容：DuckDB の文字列変換関数を 5 段階で連鎖適用し、対象文字種を一括変換する。ユーザーが変換方向（半角化または全角化）を選択し、英大文字・英小文字・数字・空白・記号の各文字グループを個別に変換する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - 対象カラム名、変換方向（半角化または全角化）
    - データの内容：変換対象の文字列カラムおよび変換方向を指定。
- **出力**
  - 同一カラムの値を全角または半角に統一（上書き）し、Parquet 形式で保存する。
    - データの形式：DuckDB テーブル内カラム（同一列上書き）
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN018】文字カラム操作（テキスト抽出・検索と置換・カラム結合）＜新規開発＞

本システム機能の概要

職員が、データテーブルのカラムに対して、テキスト抽出・検索と置換・カラム結合の 3 種の文字カラム操作を実行することができる。

本機能は、文字列置換、文字列切り出し、2 列連結の 3 種類の操作を提供するテーブルアクションである。

本システム機能のフローチャート

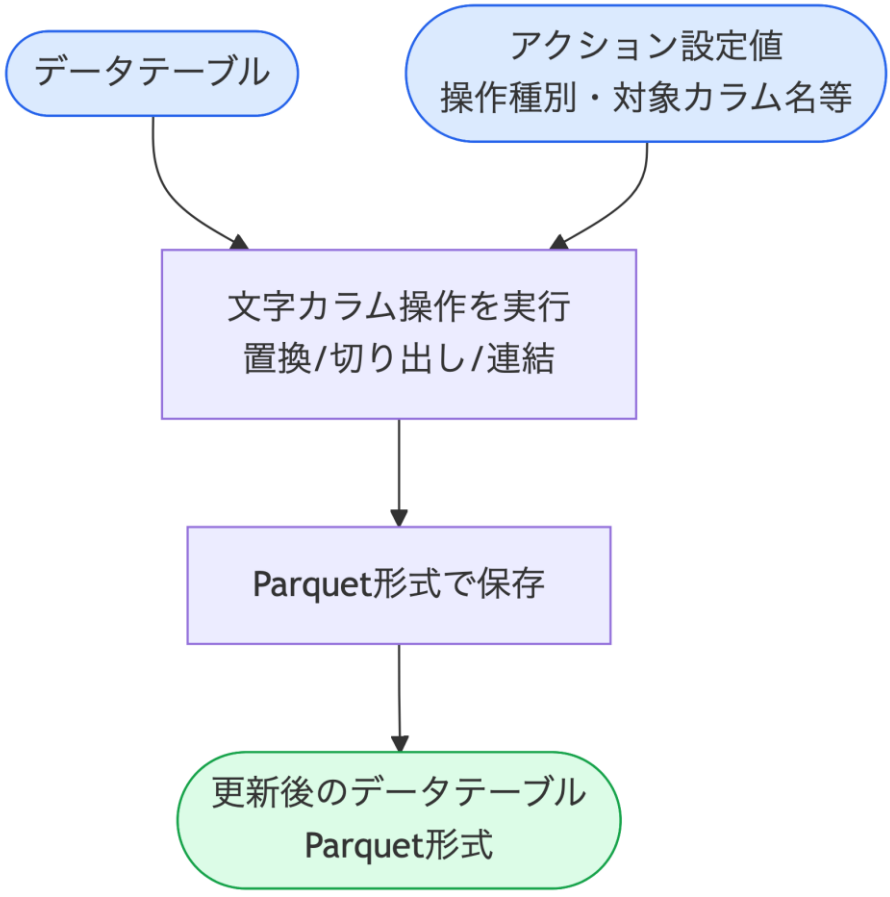


図 2.19　【FN018】のフローチャート

本システム機能の処理の詳細

- **文字列置換**
  - 処理内容：対象カラム内の文字列を検索・置換する。検索文字列が空の場合は NULL および空文字列を置換対象とする。DuckDB の文字列置換関数を使用する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

- **文字列切り出し**
  - 処理内容：対象カラムの文字列から部分文字列を抽出する。切り出しモード（先頭から、末尾から、開始位置・長さ指定）を選択する。文字列長が不足する場合のフォールバック動作（空文字列を返す/元の値を返す）を指定できる。DuckDB の部分文字列抽出関数を使用する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

- **2 列連結**
  - 処理内容：2 つのカラムの値を指定セパレータで連結し、新規カラムまたは既存カラムに出力する。NULL 値は空文字列に変換して連結する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

■ 文字列置換：対象カラム名、検索文字列、置換文字列

■ 文字列切り出し：対象カラム名、出力カラム名、切り出しモード、切り出し長、開始位置、フォールバック動作

■ 2 列連結：対象カラム名、参照カラム名、出力カラム名、セパレータ

● **出力**

■ 文字列置換：同一カラム上書き。Parquet 形式で保存する。

■ 文字列切り出し：新規カラムまたは既存カラム更新。Parquet 形式で保存する。

■ 2 列連結：新規カラムまたは既存カラム更新。Parquet 形式で保存する。

【FN019】住所文字列の正規化＜新規開発＞

本システム機能の概要

職員が、データテーブルの住所カラムに対して、日本の住所形式と文字表記を正規化することができる（都道府県名の補完や番地表記の統一等を含む）。
本機能では、丁目・番地の正規化、地名粒子変換、半角カナから全角への変換、濁点・半濁点の合成を行う。

本システム機能のフローチャート

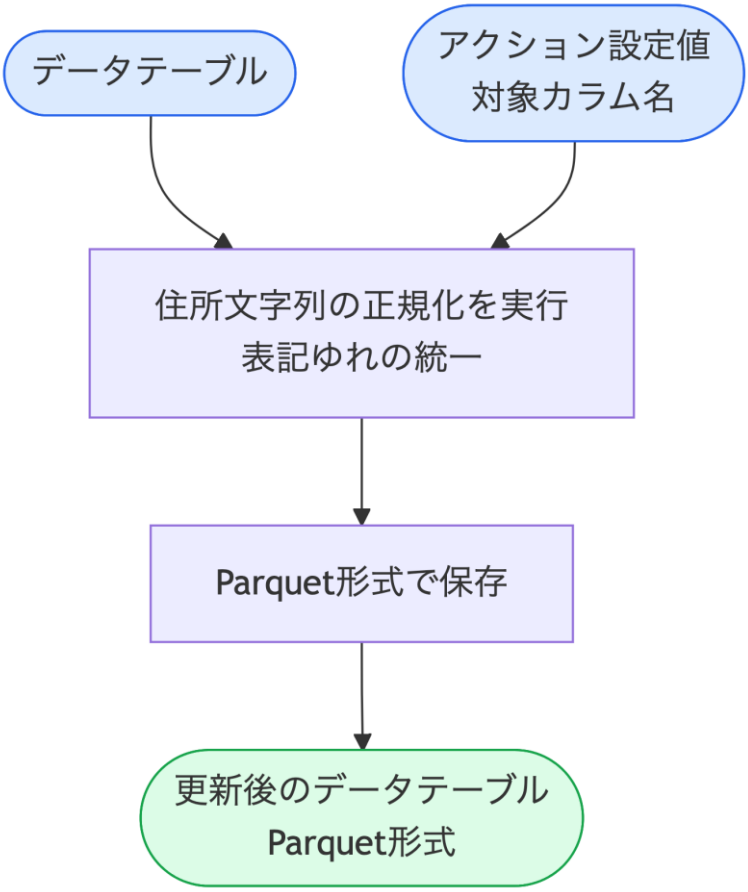


図 2.20　【FN019】のフローチャート

本システム機能の処理の詳細

- **住所正規化処理**
  - ■ 処理内容：DuckDB の正規表現置換関数と文字変換関数を組み合わせ、以下の正規化を実行する。（1）丁目・番地の漢数字変換、（2）地名粒子（ケ・ヶ・が 等）の統一、（3）半角カタカナから全角カタカナへの変換、（4）濁点（゛）・半濁点（゜）と基底文字の合成。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 対象住所カラム名
- **出力**
  - ■ 同一カラムの住所文字列を正規化（上書き）し、Parquet 形式で保存する。
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN020】日付文字列の正規化・データ型変換＜新規開発＞

本システム機能の概要

職員が、データテーブルの日付カラムに対して、和暦や様々な日付形式を YYYY-MM-DD 等の標準形式に変換することができる（整数・小数・テキスト・日付・日時等へのデータ型変換を含む）。

本機能は、3 種類のデータ変換を提供するテーブルアクションである。日付正規化では多様な日付文字列（和暦・西暦・ドット区切り等）を指定フォーマットに統一する。データ型変換ではカラムのデータ型を整数・浮動小数点・文字列・日付・日時に変換する（STRUCT / STRUCT[] の子要素にも対応する）。日付型変換（レガシー）では和暦（令和・平成・昭和）や複数日付表記を TIMESTAMP 型に変換する。

本システム機能のフローチャート

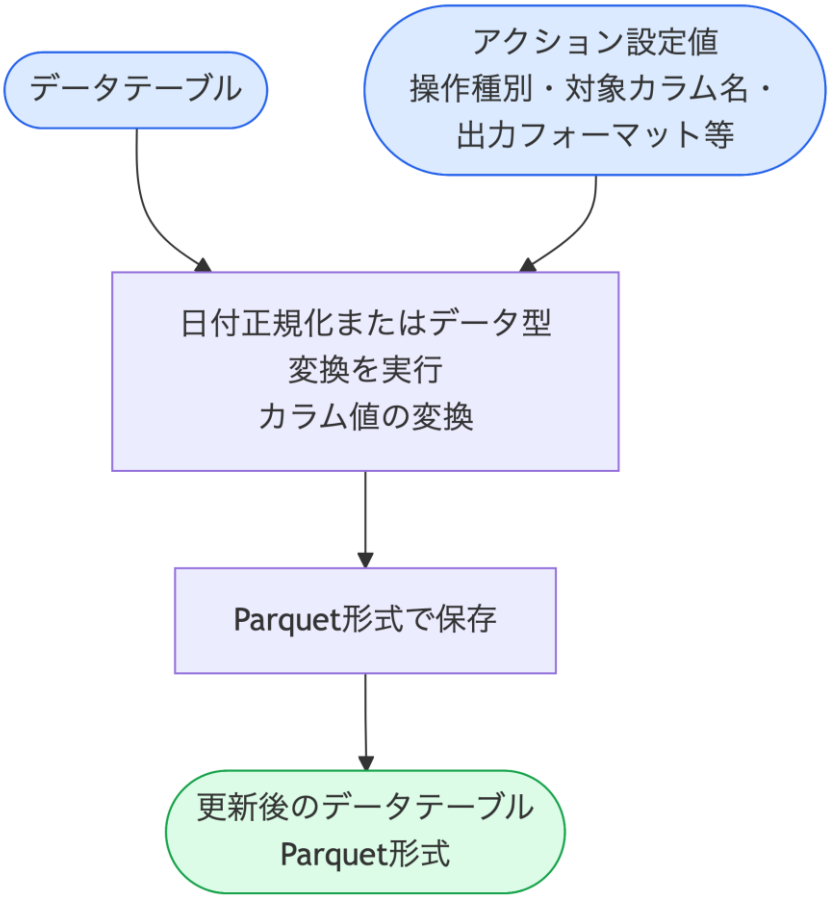


図 2.21　【FN020】のフローチャート

本システム機能の処理の詳細

- **日付正規化**
  - 処理内容：和暦（令和/平成/昭和/R/H/S）・西暦（年月日/スラッシュ/ハイフン/ドット区切り）等の多様な日付表記を順次パースし、ユーザーが指定したフォーマットで日付文字列を出力する。複数のパーサーを組み合わせ、最初に成功したパース結果を採用する
  - 利用するライブラリ：【SL009】DuckDB（node-api）

- **データ型変換**
  - 処理内容：対象カラムを指定型（整数/浮動小数点/文字列/日付/日時）に変換する。変換失敗時は NULL を返す。STRUCT 型フィールドの場合は子要素のみを変換し再構築する。日付型への変換時は和暦パースも適用する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

- **日付型変換（レガシー）**
  - 処理内容：20 以上のパターン（令和/平成/昭和/R/H/S/年月日/ドット等）を正規表現で順次マッチし、TIMESTAMP 型へ変換する。既に日付型の場合はスキップする。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 日付正規化：対象カラム名、出力フォーマット
  - ■ データ型変換：対象カラム名、変換先データ型
  - ■ 日付型変換（レガシー）：対象カラム名
- **出力**
  - ■ 同一カラムの値を変換（上書き）し、Parquet 形式で保存する。変換失敗時は NULL。
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN021】カテゴリクレンジング＜新規開発＞

本システム機能の概要

職員が、データテーブルのカラムに対して、部分一致または類似度を使用してカテゴリ値を標準化することができる（表記ゆれのあるカテゴリ値を正規カテゴリにマッチングし、しきい値未満は null を設定する。都道府県・経営形態のプリセットを備える）。

本機能は、ユーザーが定義した正規カテゴリリストに基づき、カラム値をファジーマッチングで標準化するテーブルアクションである。部分一致を優先し、一致しない場合は Jaro 類似度で最も類似するカテゴリに寄せる。

本システム機能のフローチャート

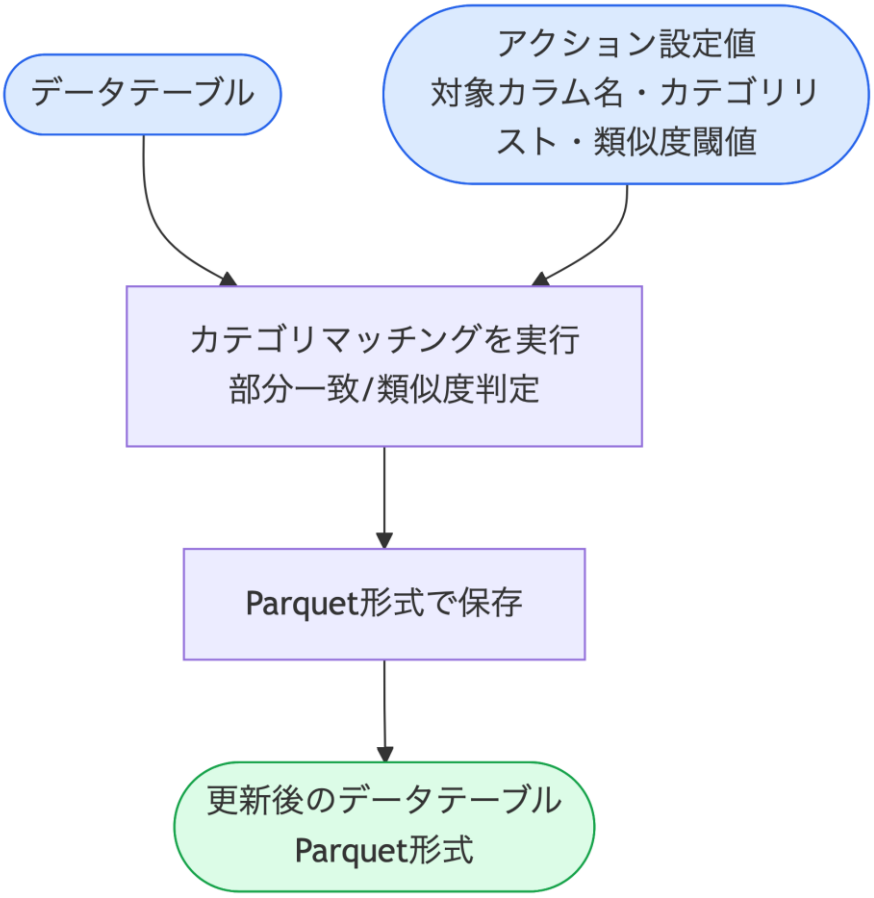


図 2.22　【FN021】のフローチャート

本システム機能の処理の詳細

- **カテゴリマッチング処理**
  - 処理内容：多段階の CTE（Common Table Expression、共通テーブル式）で以下を実行する。（1）部分一致判定、（2）都道府県接尾辞（都/道/府/県）の考慮、（3）Jaro 類似度による類似度計算、（4）閾値（1〜100%を 0〜1 に正規化）以上の最高類似度カテゴリを採用。いずれにも該当しない場合は NULL を設定する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：Jaro 類似度（DuckDB 組み込み関数）

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - 対象カラム名、正規カテゴリリスト、類似度閾値（1〜100%）
- **出力**
  - 同一カラムの値をマッチしたカテゴリに置換（上書き）し、Parquet 形式で保存する。該当なしの場合は NULL。
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN022】四則演算で新しいカラムを作成＜新規開発＞

本システム機能の概要

職員が、既存の数値カラムに対して、四則演算（+, -, ×, ÷）を行い、結果を新しいカラムとして作成することができる（フィールド・数値・演算子・括弧を組み合わせて式を構築する）。

本機能は、算術式（フィールド参照・演算子・定数の組み合わせ）を指定し、計算結果を新規カラムまたは既存カラムに出力するテーブルアクションである。STRUCT / STRUCT[] のネストフィールドに対する演算にも対応する。

本システム機能のフローチャート

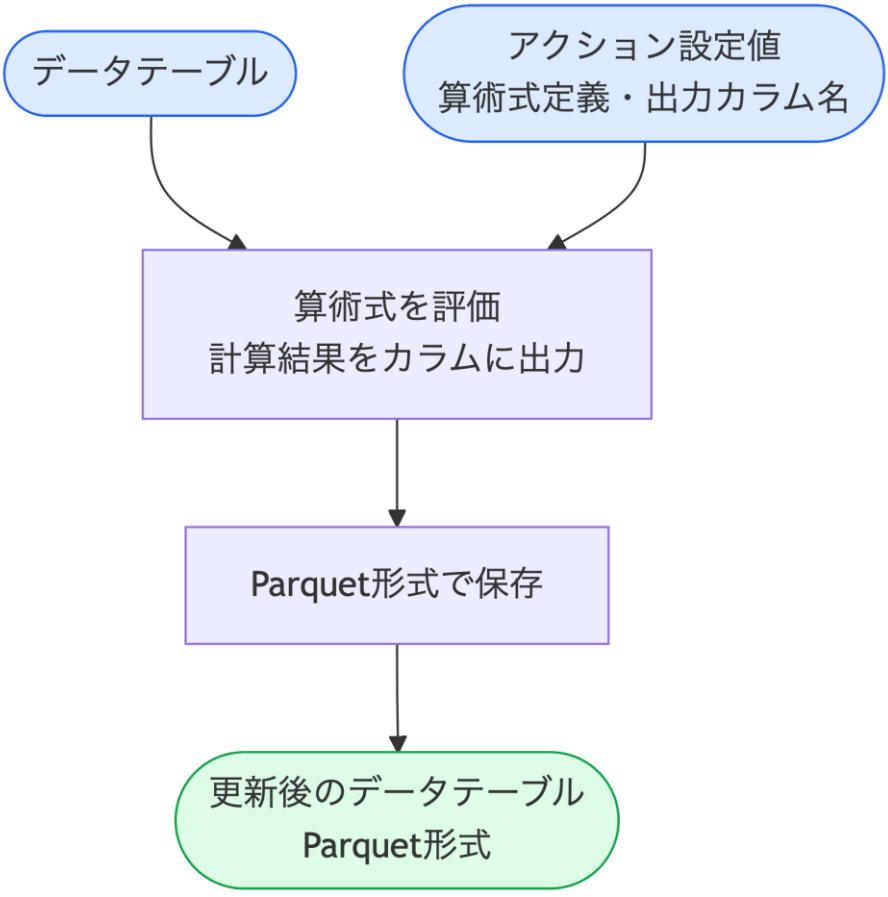


図 2.23　【FN022】のフローチャート

本システム機能の処理の詳細

- **算術式評価**
  - ■ 処理内容：式トークン配列（フィールド参照・演算子・数値定数）から DuckDB SQL 式を構築して実行する。フラットフィールドの場合は直接 SQL 式を組み立て、STRUCT フィールドの場合は構造体を再構築、STRUCT[] の場合はリスト変換関数でラムダ関数として適用する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 式定義配列（フィールド参照・演算子・数値定数のトークン列）、出力カラム名
- **出力**
  - ■ 新規カラムまたは既存カラムに計算結果を出力し、Parquet 形式で保存する。
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN023】数値階層化・数値置換＜新規開発＞

本システム機能の概要

職員が、データテーブルの数値カラムに対して、指定された数のランク層にデータを分割することができる（自動の等間隔または手動のしきい値指定で分割方法を選択できる。比較条件に基づく数値の置き換えや NULL 置換にも対応する）。

本機能は、2 種類の数値変換を提供するテーブルアクションである。数値階層化では数値カラムをランク化してランクカラムを生成する。数値条件置換では数値条件に一致する値を指定値で置換する。

本システム機能のフローチャート

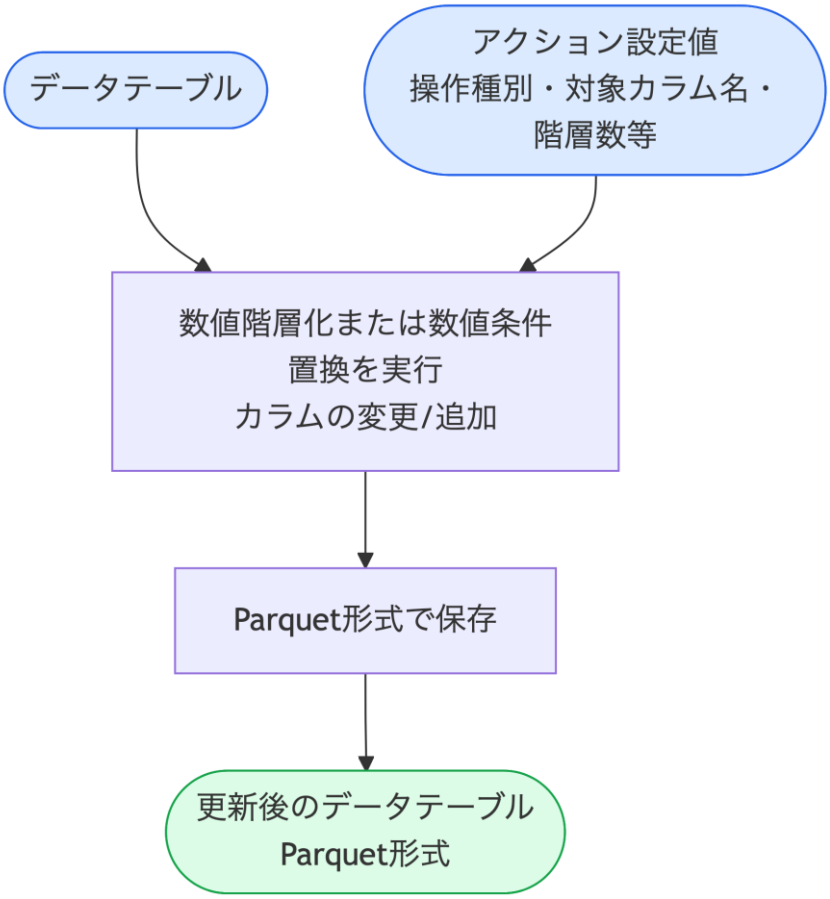


図 2.24　【FN023】のフローチャート

本システム機能の処理の詳細

- **数値階層化**
  - ■ 処理内容：等幅分位モード（階層数指定）またはカスタム閾値モード（閾値配列指定）で数値を階層化する。等幅の場合は最小値・最大値の範囲を指定階層数で等分し、各値が属する階層番号を算出する。カスタム閾値の場合は CASE 式で範囲ラベルを付与する。STRUCT / STRUCT[] フィールドにも対応する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）

- **数値条件置換**
  - ■ 処理内容：対象カラムの値を数値として比較し、条件演算子（=, !=, <, <=, >, >=）と閾値に一致する行の値を置換値で上書きする。置換値に NULL を指定可能。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 数値階層化：対象カラム名、階層数またはカスタム閾値配列
  - ■ 数値条件置換：対象カラム名、比較演算子、比較値、置換値
- **出力**
  - ■ 数値階層化：ランク新規カラムを追加し、Parquet 形式で保存する。

■ 数値条件置換：同一カラム上書きし、Parquet 形式で保存する。
■ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN024】ID 附番・マルチテーブル ID 付与＜新規開発＞

本システム機能の概要

職員が、データテーブルのカラムに対して、類似度のしきい値に基づいて類似した値に同じ ID を割り当てることができる（複数カラムの条件を設定できる。複数テーブル間でレコードをマッチングし共通 ID を割り当てる）。

本機能は、2 種類の ID 付与を提供するテーブルアクションである。単表 ID 附番では単一テーブル内で同一値または類似値のレコードに共通 ID を付与する。複数表 ID 附番では複数テーブル間でマッチング条件に基づき共通 ID を採番する。Jaro 類似度によるファジーマッチングに対応する。

本システム機能のフローチャート

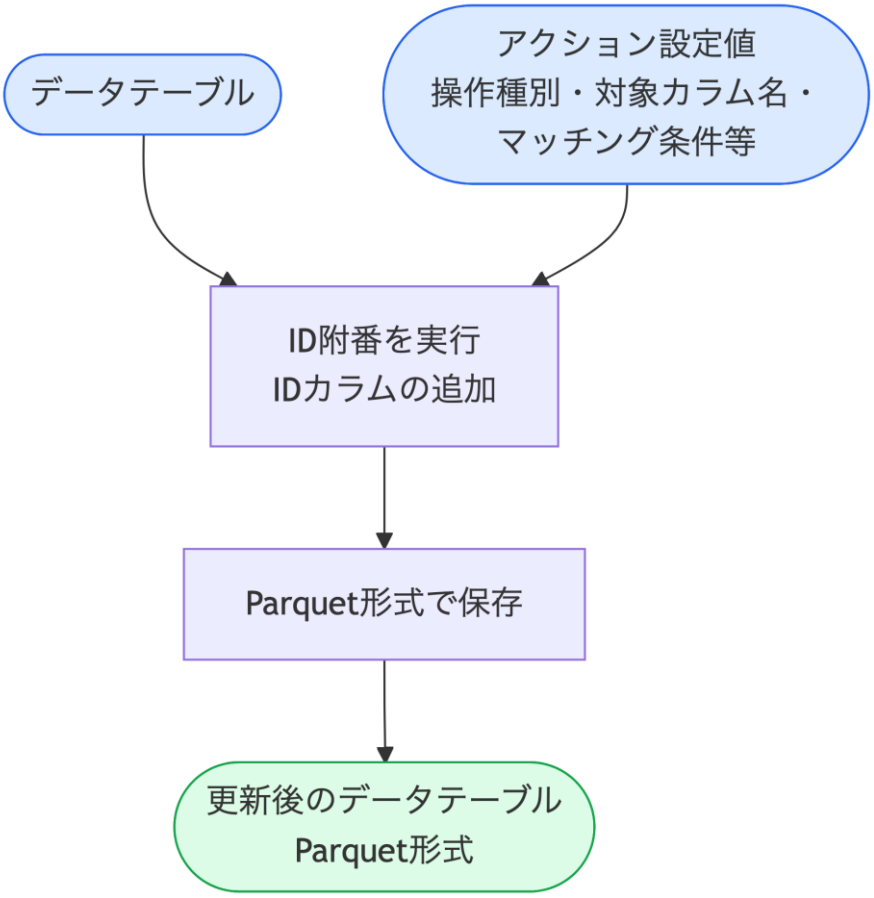


図 2.25　【FN024】のフローチャート

本システム機能の処理の詳細

● **単表 ID 附番**

■ 処理内容：対象カラム（1 つまたは複数）の値が完全一致するレコードを密ランク関数でグループ化し、連番 ID を付与する。類似度閾値が指定された場合は Jaro 類似度でクロス結合し、閾値以上の組をグループ化する。NULL /空文字列は個別 ID を付与する。

■ 利用するライブラリ：【SL009】DuckDB（node-api）

■ 利用するアルゴリズム：Jaro 類似度（DuckDB 組み込み関数）

● **複数表 ID 附番**

■ 処理内容：3 つのモードを持つ。（1）レガシー単表モード、（2）ペアベースモード（参照テーブル指定）、（3）グループベースモード（マッチングテーブル＋マッチンググループ指定）。グループベースモードでは、マッチング条件グラフの連結成分を再帰 CTE で探索し、同一連結成分に属するレコードに共通 ID を採番する。ソーステーブルだけでなく参照テーブル側の Parquet も更新する。

■ 利用するライブラリ：【SL009】DuckDB（node-api）

■ 利用するアルゴリズム：グラフ連結成分探索（再帰 CTE）、Jaro 類似度

本システム機能の入出力データの仕様

● **入力**

■ 処理対象のデータテーブル

▲ データの形式：DuckDB テーブル

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

■ 単表 ID 附番：対象カラム名（1 つまたは複数）、類似度閾値

■ 複数表 ID 附番：マッチングテーブル定義、マッチンググループ定義、参照テーブル

● **出力**

■ 単表 ID 附番：ID 新規カラムを追加し、Parquet 形式で保存する。

■ 複数表 ID 附番：クロステーブル ID 等の新規カラムをソース・参照両テーブルに付与し、Parquet 形式で保存する。

■ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN025】重複行の削除＜新規開発＞

本システム機能の概要

職員が、選択したカラムの値が一致する行を重複として削除することができる（残すレコードを最初のレコード・最新/最大・最古/最小から選択できる）。
本機能は、【FN003】データセット管理のテーブルアクション実行基盤上で実行する。

本システム機能のフローチャート

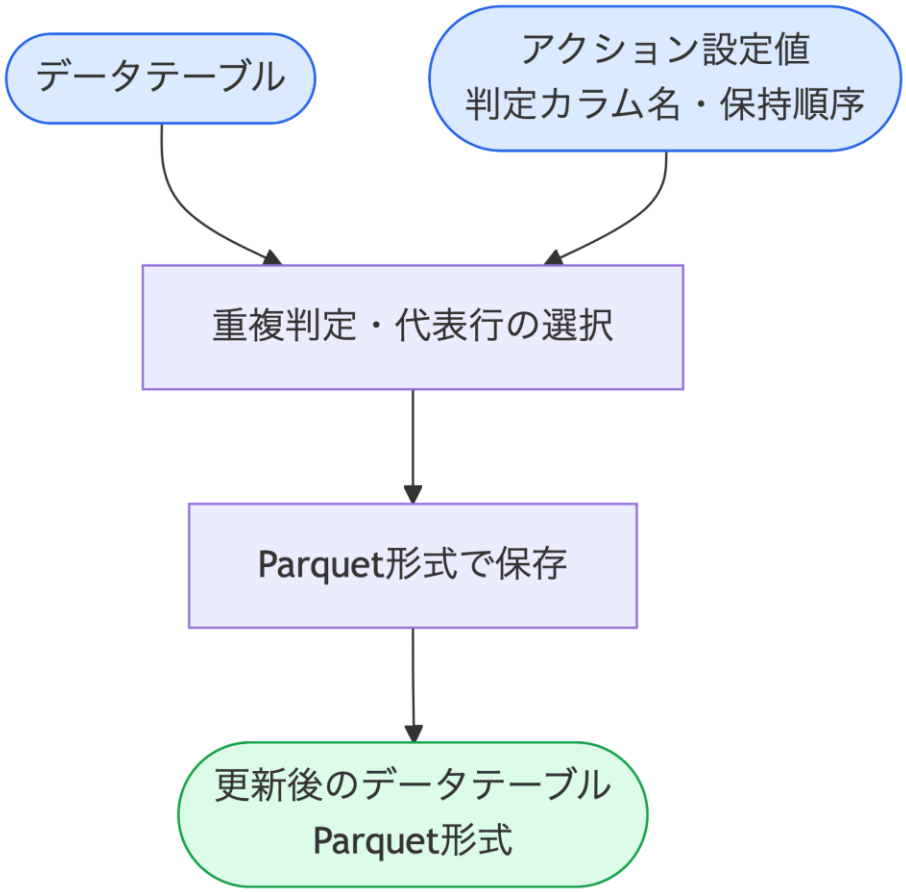


図 2.26　【FN025】のフローチャート

本システム機能の処理の詳細

- **重複排除処理**
  - ■ 処理内容：ユーザーが指定したカラムの組み合わせで各行を重複グループに分類し、グループ内で番号を付与して先頭の行のみを残す。保持順序として「最初の出現」「指定カラムの最小値」「指定カラムの最大値」を選択できる。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ データテーブル、重複判定カラム名、保持順序（最初の出現/最小値/最大値）、順序カラム名
- **出力**
  - ■ 重複除去後のデータテーブル（Parquet 形式で保存）

【FN026】テキスト結合＜新規開発＞

本システム機能の概要

職員が、類似度閾値を使用してテキストカラムをマッチングしテーブルを結合することができる（必須条件・部分条件を設定できる。結合タイプを左外部結合・内部結合・外部結合から選択できる）。

本機能は、2 つのデータテーブルをテキスト条件（完全一致または Jaro 類似度）で結合するテーブルアクションである。【FN003】データセット管理のテーブルアクション実行基盤上で実行する。

本システム機能のフローチャート

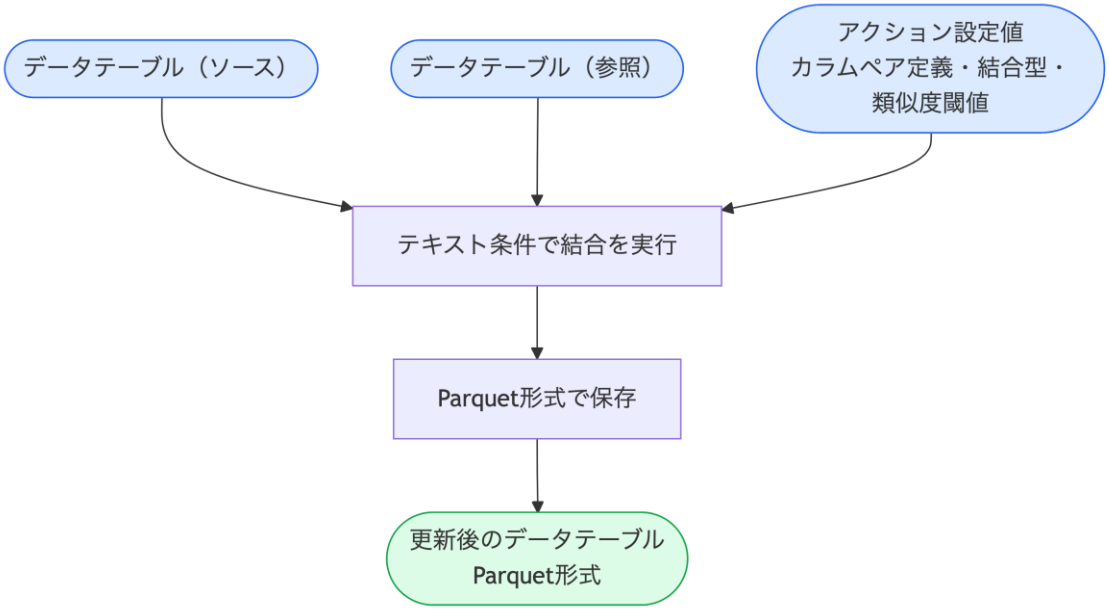


図 2.27 【FN026】のフローチャート

本システム機能の処理の詳細

- **テキスト条件結合処理**
  - 処理内容：カラムペア定義（左テーブルカラム・右テーブルカラム・類似度閾値・必須/任意フラグ）に基づき結合条件を構築する。必須条件はすべて一致する必要があり、部分条件は指定した最低一致数以上の一致で結合する。各条件で Jaro 類似度による閾値判定が可能。結合型として左外部結合・内部結合・完全外部結合を選択できる。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：Jaro 類似度（DuckDB 組み込み関数）

本システム機能の入出力データの仕様

- **入力**
  - ■ データテーブル（ソース）、データテーブル（参照）、カラムペア定義（左右のカラム名・類似度閾値・必須/任意フラグ）、結合型、部分条件の最低一致数
- **出力**
  - ■ 結合済みデータテーブル（参照テーブルのカラムがプレフィックス付きで追加、Parquet 形式で保存。）

【FN027】ジオコーディング＜新規開発＞

本システム機能の概要

職員が、選択したカラムの住所値を WKT 形式のジオメトリに変換し、新しいジオメトリカラムを作成することができる。

本機能では、外部ジオコーディング API（AWS Location Service）を使用して住所カラムから緯度・経度座標を取得し、WKT 形式のジオメトリカラムを生成する。
STRUCT / STRUCT 配列カラムのネストフィールドにも対応する。【FN003】データセット管理のテーブルアクション実行基盤上で実行する。

本システム機能のフローチャート

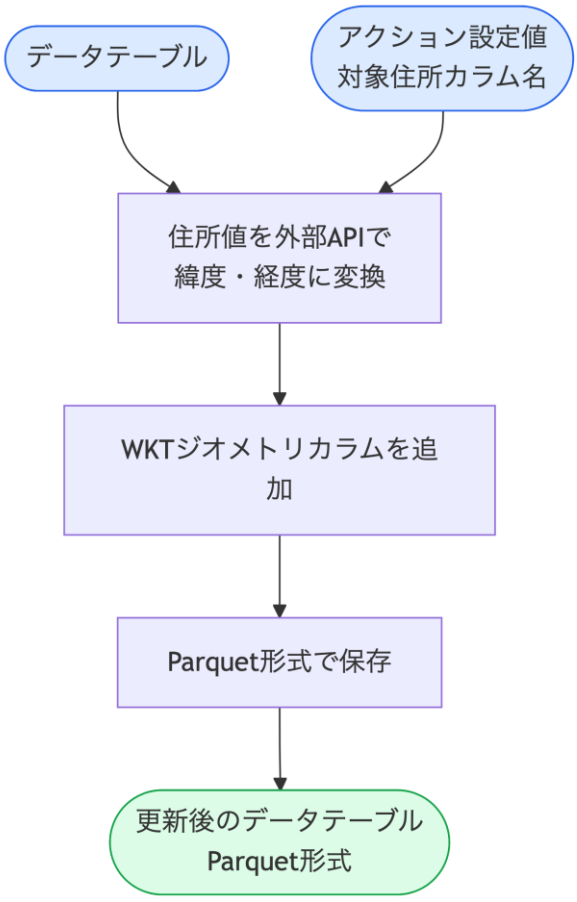


図 2.28　【FN027】のフローチャート

本システム機能の処理の詳細

- **ジオコーディング処理**
  - 処理内容：対象カラムの住所値をバッチ（10 件ずつ、100ms 間隔）で外部ジオコーディング API に送信し、緯度・経度を取得する。取得した座標を WKT 形式のジオメトリに変換し、新規ジオメトリカラムとして追加する。STRUCT 型の場合は子要素として追加、STRUCT 配列の場合は各要素に付与する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - データテーブル、対象住所カラム名
- **出力**
  - ジオメトリカラムが追加されたデータテーブル（WKT 形式、Parquet 形式で保存。）

【FN028】空間結合＜新規開発＞

本システム機能の概要

職員が、ジオメトリの交差または最近傍でテーブルを結合することができる（WKT 形式のジオメトリカラムをキーに使用する。最近傍検索では探索半径をメートル単位で指定する）。

本機能では、DuckDB の spatial 拡張を使い、2 つのデータテーブルをジオメトリ条件で結合する。交差結合と最近傍結合の 2 モードに対応する。【FN003】データセット管理のテーブルアクション実行基盤上で実行する。

本システム機能のフローチャート

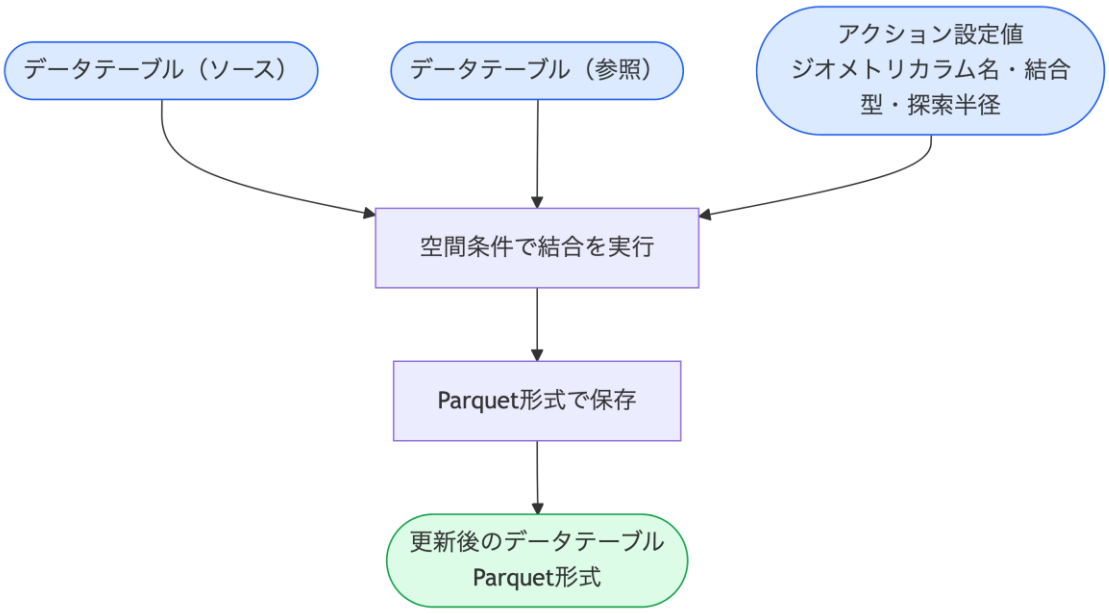


図 2.29 【FN028】のフローチャート

本システム機能の処理の詳細

● **空間結合処理**

■ 処理内容：交差モードでは空間的に重なる地物同士を結合する。最近傍モードでは 2 つのジオメトリ間の距離を算出し、探索半径内で最も近い地物 1 件を選択して結合する。

■ 利用するライブラリ：【SL009】DuckDB（node-api）、DuckDB spatial 拡張

■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ データテーブル（ソース）、データテーブル（参照）、ソースジオメトリカラム名、参照ジオメトリカラム名、結合型（交差/最近傍）、探索半径（メートル）
- **出力**
  - ■ 結合済みデータテーブル（参照テーブルのカラムがプレフィックス付きで追加、Parquet 形式で保存。）

【FN029】テキスト＋空間結合＜新規開発＞

本システム機能の概要

職員が、テキストマッチングと半径内の空間最近傍を組み合わせた結合を行うことができる（テキストと位置の両方を考慮した高精度な結合に対応する）。
本機能は、テキスト候補をまず抽出し、空間距離で絞り込み、最近傍 1 件を採用する複合結合のテーブルアクションである。【FN003】データセット管理のテーブルアクション実行基盤上で実行する。

本システム機能のフローチャート

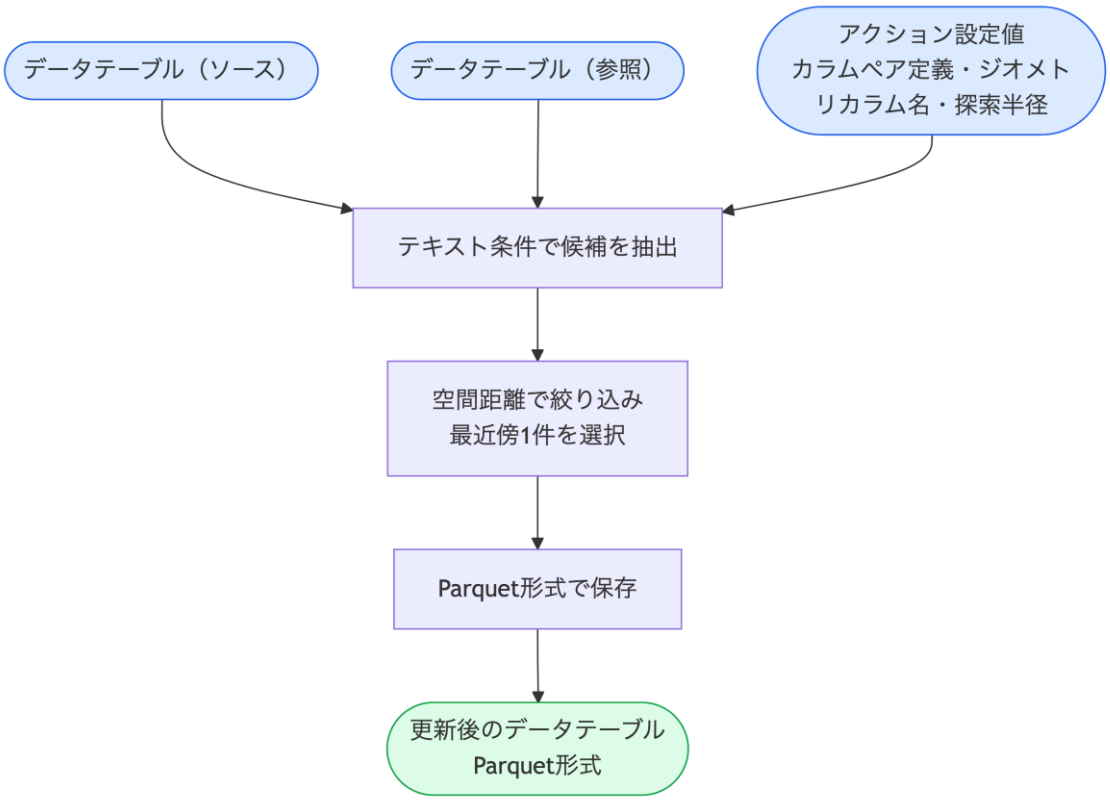


図 2.30　【FN029】のフローチャート

本システム機能の処理の詳細

- **複合結合処理**
  - 処理内容：（1）テキスト条件（完全一致/ Jaro 類似度）で候補レコードを抽出する、（2）候補のうち空間距離が探索半径内のものに絞り込む、（3）ソーステーブル 1 行につき最近傍 1 件を選択する。テキスト条件は必須条件と部分条件の組み合わせに対応する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）、DuckDB spatial 拡張
  - 利用するアルゴリズム：Jaro 類似度（DuckDB 組み込み関数）

本システム機能の入出力データの仕様

- **入力**
  - データテーブル（ソース）、データテーブル（参照）、カラムペア定義（左右のカラム名・類似度閾値・必須/任意フラグ）、ソース/参照ジオメトリカラム名、探索半径（メートル）、部分条件の最低一致数
- **出力**
  - 結合済みデータテーブル（参照テーブルのカラムがプレフィックス付きで追加、Parquet 形式で保存。）

【FN030】オープンデータ作成＜新規開発＞

本システム機能の概要

職員が、データセットに対して、住所秘匿化・数値階層化・数値偏差値化等、オープンデータ化に必要なテーブルアクションをノーコードで実行することができる。
本機能では、住所秘匿化（住所情報のマスキング）、数値階層化（実数値のマスキング）、数値偏差値化（元の数値を偏差値スコアに変換）等の処理を提供する。住所秘匿化（【FN039】）と数値階層化（【FN023】）と数値偏差値化（【FN040】）が具体的なテーブルアクションとして利用できる。

本システム機能のフローチャート

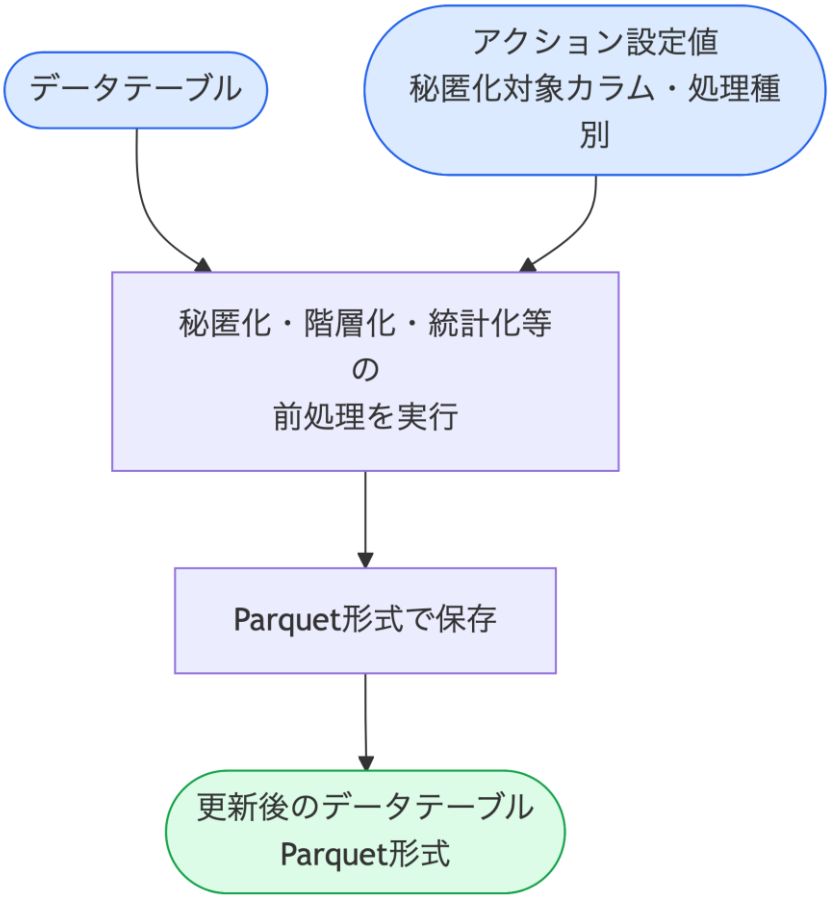


図 2.31　【FN030】のフローチャート

本システム機能の処理の詳細

- **データマスキング処理**
  - ■ 処理内容：指定されたカラムに対して、個人情報・機密情報のマスキング処理を実行する。マスキング対象のカラム値を不可逆に変換し、元データの復元が不可能な状態にする。

- **数値階層化処理**
  - ■ 処理内容：営業実績や財務情報などの数値に対して、実数を直接公開しないようにランク化処理を実行する。

- **数値偏差値化処理**
  - ■ 処理内容：数値カラムの元の値を偏差値（平均 50・標準偏差 10）に変換し、生データを直接公開せずに相対的な位置づけだけを示す処理を実行する。

本システム機能の入出力データの仕様

- **入力**
  - ■ データテーブル、秘匿化対象カラム名、処理種別（マスキング/数値階層化/数値偏差値化）
- **出力**
  - ■ オープンデータ化処理済みのデータテーブル（Parquet 形式で保存。）

【FN031】カラム名を翻訳する＜新規開発＞

本システム機能の概要

職員は、オープンデータ化や国際利用を想定し、LLM を活用してデータテーブルの日本語カラム名を英語に変換できる。

本機能は、【FN003】データセット管理のテーブルアクション実行基盤上で実行する。利用するライブラリは【SL020】boto3、利用するインターフェースは【IF012】AWS Bedrock API である。

本システム機能のフローチャート

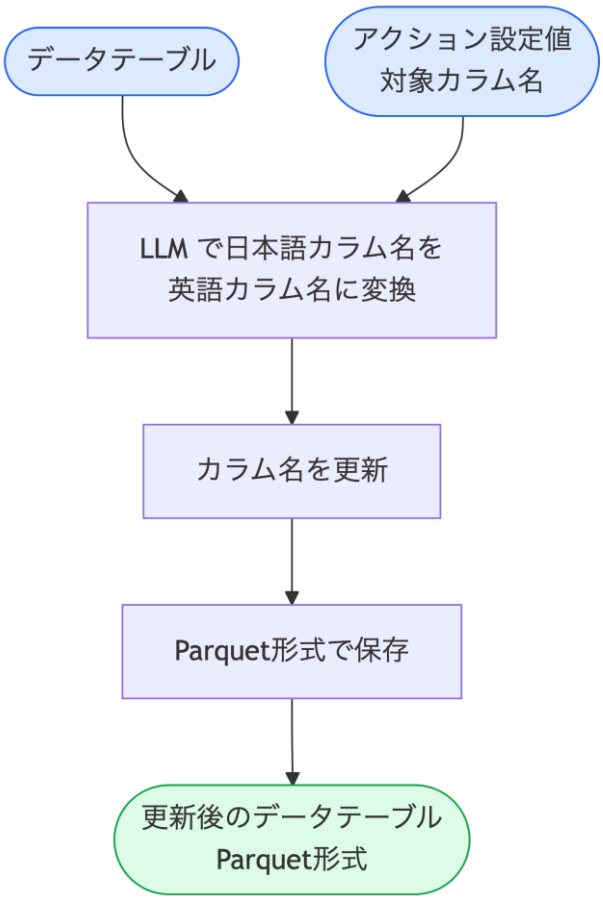


図 2.32　【FN031】のフローチャート

本システム機能の処理の詳細

- **英語カラム名変換処理**
  - ■ 処理内容：データテーブルの日本語カラム名を LLM に送信し、適切な英語カラム名への変換候補を取得する。変換結果でカラム名を更新し、Parquet 形式で保存する。
  - ■ 利用するライブラリ：【SL020】boto3
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ データテーブル、変換対象のカラム名一覧
- **出力**
  - ■ 英語カラム名に変換されたデータテーブル（Parquet 形式で保存。）

【FN032】キーワード分類＜新規開発＞

本システム機能の概要

職員が、データテーブルのテキストカラムに対して、キーワードを分類・割り当てることができる（手動モードでは事前定義したキーワードパターンに基づきルールベースで分類する。AI モードでは LLM がテキスト内容を分析して適切なキーワードを自動で割り当てる）。

本機能は、手動モードと AI モードの 2 つの分類方式を提供するテーブルアクションである。手動モードでは、キーワードごとに複数のパターンを定義し、対象カラムの値がいずれかのパターンに部分一致した場合にそのキーワードを新規カラムに出力する。大文字小文字は区別しない。AI モードでは、LLM がテキスト内容を分析し、適切なキーワードを自動的に割り当てる。

本システム機能のフローチャート

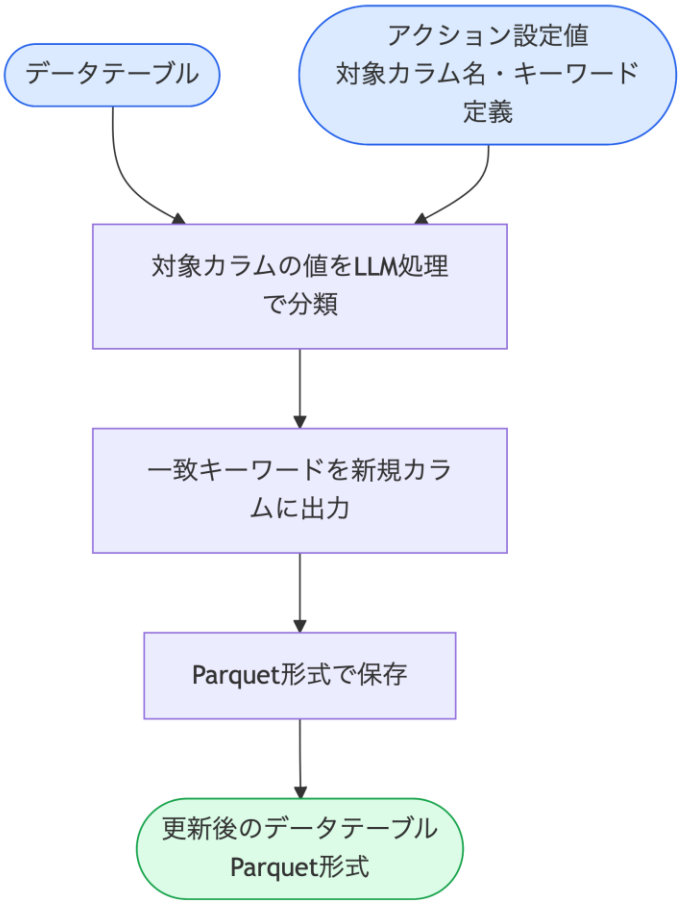


図 2.33　【FN032】のフローチャート

本システム機能の処理の詳細

- **ルールベース分類処理**
  - ■ 処理内容：キーワード定義（キーワード名とパターン配列の組）を受け取り、対象カラムの値を LLM 処理で分類させ、指定された最大キーワード数までカンマ区切りで新規カラムに出力する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）、【SL020】boto3
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 対象カラム名、キーワード定義（キーワード名とパターン配列の組）
    - ▲ データの内容：分類ルールの設定値
- **出力**
  - ■ 新規カラムにカンマ区切りのキーワード文字列を付与したデータテーブルを Parquet 形式で保存する
    - ▲ データの形式：Apache Parquet
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN033】縦結合＜新規開発＞

本システム機能の概要

職員が、テーブルを縦に積み重ねることができる（両テーブルのすべてのカラムを保持し、行を垂直に積み重ねる。結合条件はカラム名の完全一致とする）。

本機能は、カラム名が一致するもの同士を自動マッチングし、一方にのみ存在するカラムは NULL で埋めるテーブルアクションである。結合後に行 ID を再採番する。

本システム機能のフローチャート

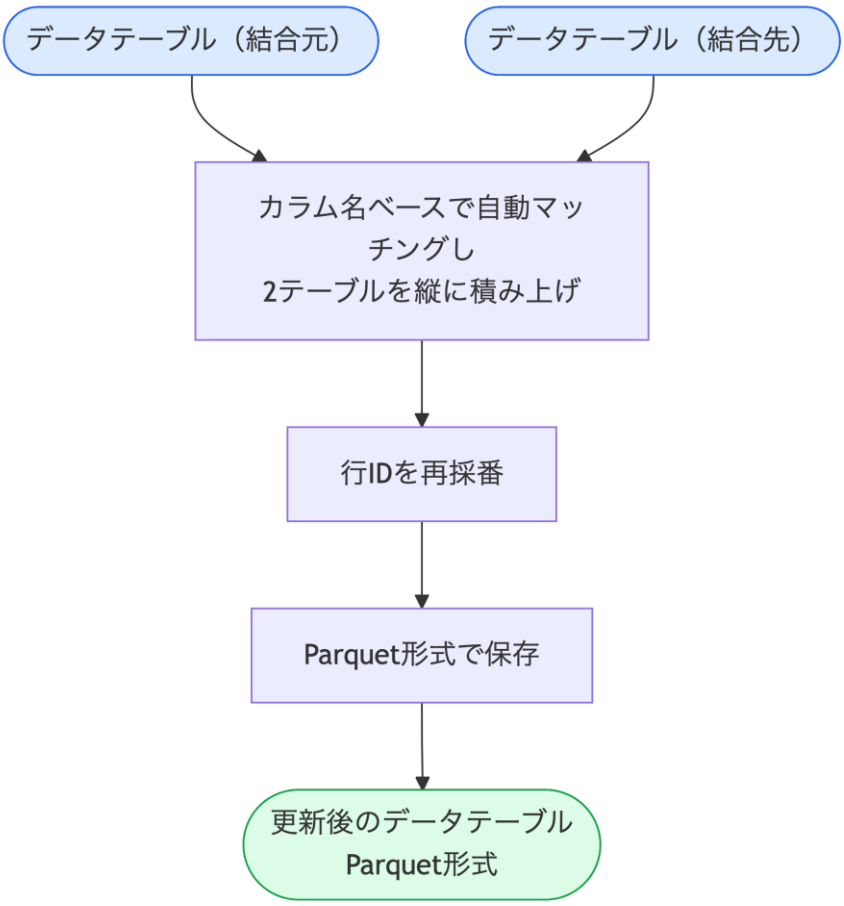


図 2.34　【FN033】のフローチャート

本システム機能の処理の詳細

- **縦結合処理**
  - ■ 処理内容：2 つのデータテーブルをカラム名ベースで縦に積み上げる。カラム名が一致するもの同士が自動マッチングされ、一方にのみ存在するカラムは NULL で埋められる。結合後に行 ID を再採番する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 結合元のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - ■ 結合先のデータテーブル（参照テーブル ID で指定）
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル
- **出力**
  - ■ 縦結合済みのデータテーブル（両テーブルの全行を統合、行 ID 再採番済み）を Parquet 形式で保存する。
    - ▲ データの形式：Apache Parquet
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN034】構造展開（テーブルの展開・グループの展開）＜新規開発＞

本システム機能の概要

職員が、テーブルカラムを複数行に展開することができる（グループカラムを個別のカラムに展開する）。

本機能は、配列カラムやグループカラムを展開するテーブルアクションである。配列カラムの場合は各要素が個別の行に展開され行数が増加し、グループカラムの場合は各フィールドが個別のカラムに展開されカラム数が増加する。

本システム機能のフローチャート

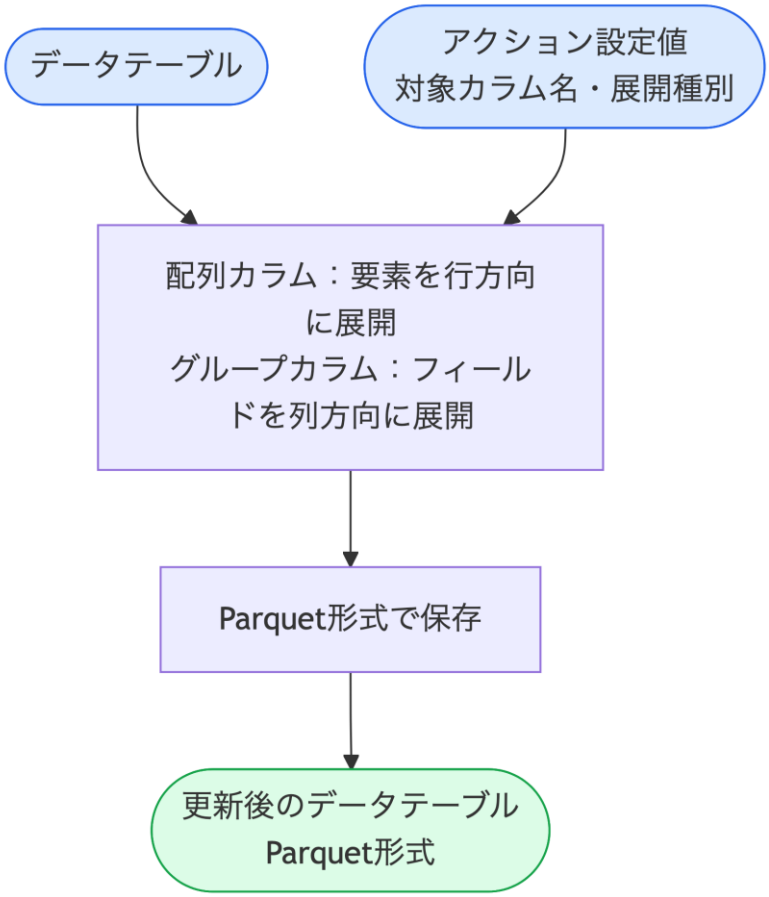


図 2.35　【FN034】のフローチャート

本システム機能の処理の詳細

- **テーブルの展開**
  - 処理内容：配列カラムの各要素を行方向に展開する。空配列や NULL の場合も 1 行を保持し、NULL 値で代替する。グループの配列の場合は展開後に子フィールドを個別カラムとして展開する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

- **グループの展開**
  - 処理内容：グループカラムの各フィールドを個別カラムに展開する。展開後のカラム名にはプレフィックスとしてフィールド名を付与する。元のグループカラムは除外する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - 対象カラム名、展開種別（LIST 展開または STRUCT 展開）
    - データの内容：展開対象のカラムおよび展開方法を指定。
- **出力**
  - LIST 展開の場合：行数が増加したデータテーブルを Parquet 形式で保存する。

■ STRUCT 展開の場合：カラム数が増加し元の STRUCT カラムを除外したデータテーブルを Parquet 形式で保存する。

▲ データの形式：Apache Parquet

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN035】テーブル列操作＜新規開発＞

本システム機能の概要

職員が、カラムの追加（空カラム）・名前変更・複製・削除・単一セル値更新の基本テーブル操作を行うことができる（GUI からのインライン編集に使用する）。
本機能はテーブルアクションとして実行され、GUI からのインライン編集や前処理に使用する。

本システム機能のフローチャート

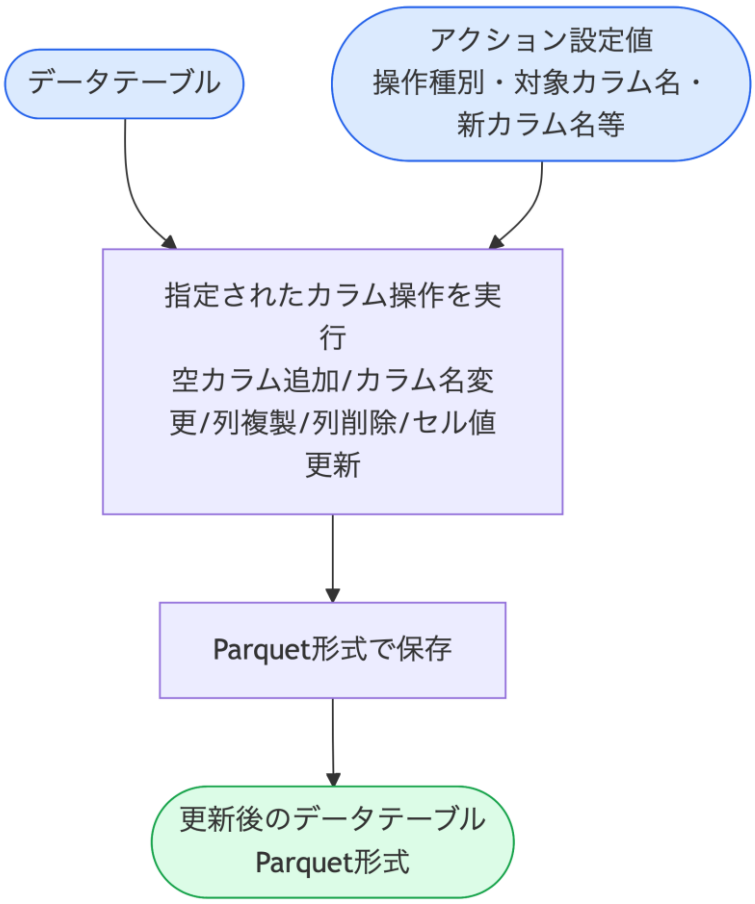


図 2.36　【FN035】のフローチャート

本システム機能の処理の詳細

- **空カラム追加**
  - 処理内容：指定名の空文字列カラムを追加する。既存カラム名との重複を禁止する。
- **カラム名変更**
  - 処理内容：対象カラムにエイリアスを付与し、カラム名を変更する。
- **列複製**
  - 処理内容：対象カラムのコピーを指定名で追加する。
- **列削除**
  - 処理内容：対象カラムを除外する。全カラム削除は禁止する。
- **単一セル値更新**
  - 処理内容：行 ID が一致する行のみ、指定カラムの値を新しい値に更新する。
- 利用するライブラリ：【SL009】DuckDB（node-api）
- 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - 操作種別、対象カラム名、新カラム名、行 ID、新しい値（操作に応じて必要なパラメータを指定。）
    - データの内容：カラム操作の設定値

● **出力**

■ 操作後のデータテーブルを Parquet 形式で保存する。

▲ データの形式：Apache Parquet

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN036】外れ値フィルター＜新規開発＞

本システム機能の概要

職員が、数値カラムの値を昇順・降順ソートし、上位・下位の外れ値候補を視覚的に確認することができる。

本機能は、データテーブル UI のカラムソート・フィルター機能として実装されており、データ自体は変更しない。

本システム機能の入力・処理・出力のフローチャート

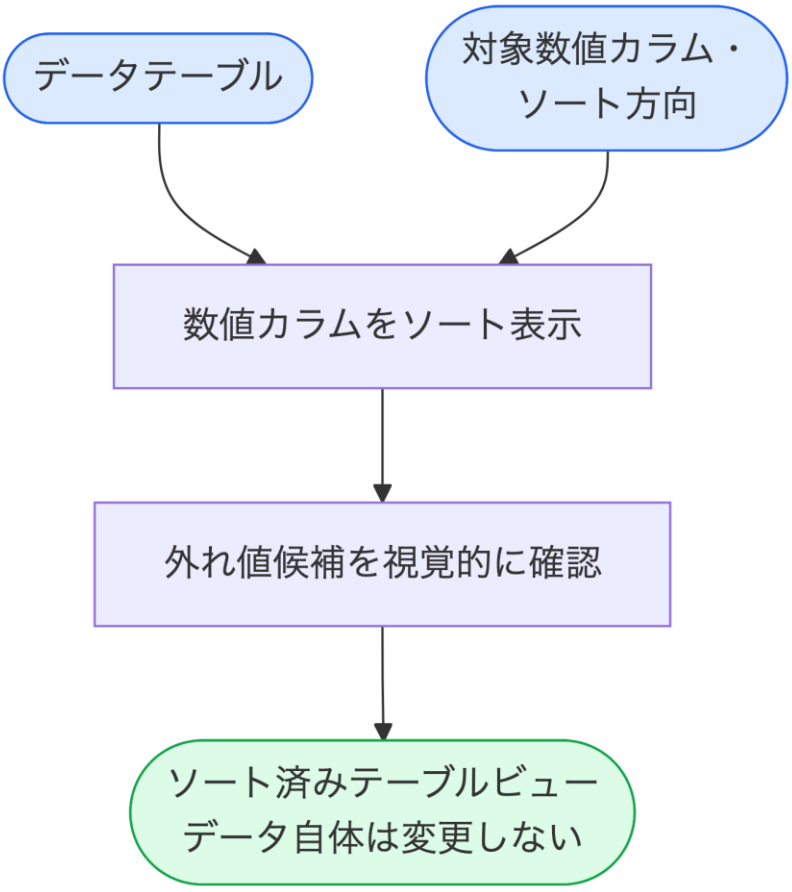


図 2.37　【FN036】のフローチャート

本システム機能の処理の詳細

- **ソート・フィルタリング処理**
  - ■ 処理内容：データテーブル画面のカラムヘッダーから昇順・降順ソートを適用し、外れ値が上位・下位に表示されるようにする。ユーザーはソート結果を目視確認し、必要に応じて【FN023】数値階層化・数値置換や【FN035】テーブル列操作で値を修正する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
  - ■ 対象数値カラム、ソート方向（昇順/降順）
- **出力**
  - ■ ソート済みテーブルビュー（データ自体は変更しない。）

【FN037】フィルター結果の新規テーブル保存＜新規開発＞

本システム機能の概要

職員が、データテーブルのフィルター機能を使用した状態のテーブルを新規派生テーブルとして保存することができる。

本機能では、フィルター適用後の結果を新しいデータテーブルとして Parquet 形式で保存する。

本システム機能の入力・処理・出力のフローチャート

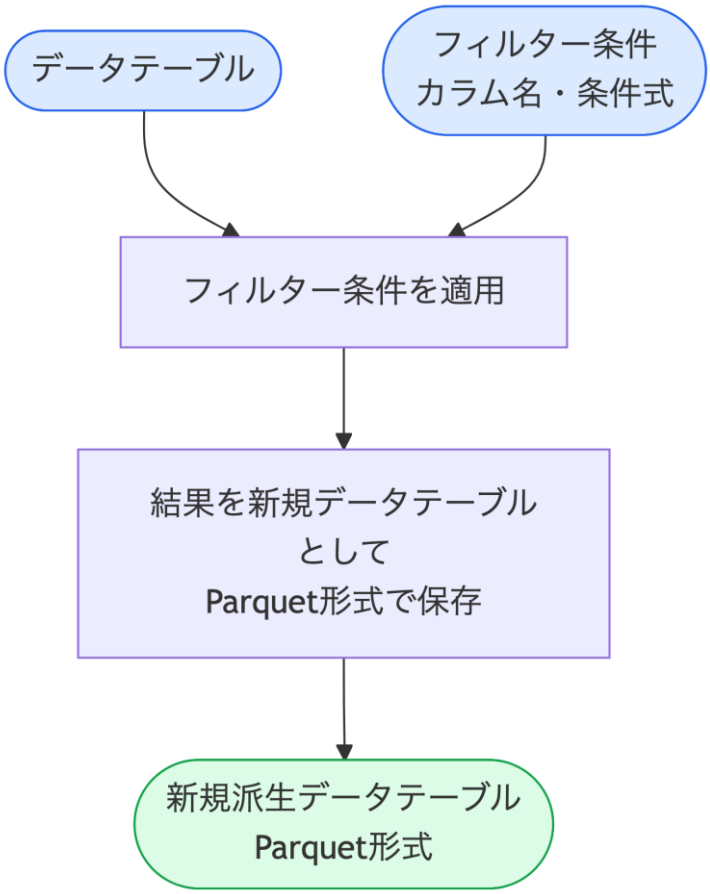


図 2.38　【FN037】のフローチャート

本システム機能の処理の詳細

- **フィルター結果の新規テーブル保存**
  - 処理内容：フィルター適用後の結果を新しいデータテーブルとして Parquet 形式で保存する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - フィルター条件（カラム名、条件式）
    - データの内容：フィルタリングの設定値
- **出力**
  - 新規データテーブル
    - データの形式：Apache Parquet
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN038】テーブルヘッダー操作＜新規開発＞

本システム機能の概要

職員が、データテーブルのカラムヘッダーに対するフィルター・重複行閲覧を行うことができる。

本機能は、データテーブル画面の UI 機能として実装されている。

本システム機能のフローチャート

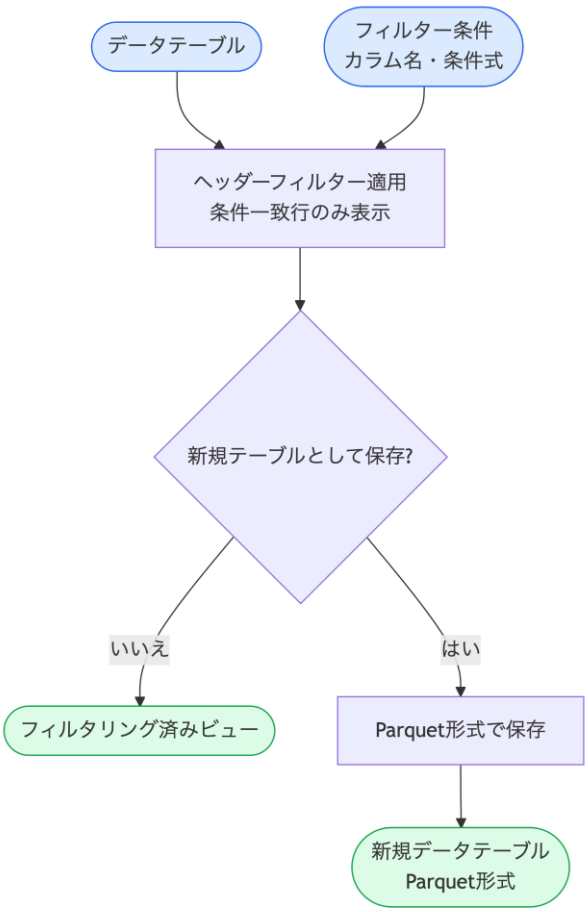


図 2.39　【FN038】のフローチャート

本システム機能の処理の詳細

- **ヘッダーフィルター**
  - 処理内容：カラムヘッダーのコンテキストメニューからフィルター条件を設定し、条件に一致する行のみを表示する。
- **重複値閲覧**
  - 処理内容：指定カラムのユニーク値一覧と出現回数を表示し、重複状況を確認できる。
- **フィルタ結果の新規テーブル保存**
  - 処理内容：フィルター適用後の結果を新しいデータテーブルとして Parquet 形式で保存する。
- 利用するライブラリ：【SL009】DuckDB（node-api）
- 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - 処理対象のデータテーブル
    - データの形式：DuckDB テーブル
    - 利用するデータインターフェース：【IF006】GCS Parquet ファイル
  - フィルター条件（カラム名、条件式）
    - データの内容：フィルタリングの設定値
- **出力**
  - フィルタリング済みビュー（画面表示のみの場合）
  - 新規データテーブルとして保存する場合は Parquet 形式で保存する。

▲ データの形式：Apache Parquet
▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN039】住所秘匿化＜新規開発＞

本システム機能の概要

職員が、住所文字列から都道府県または市区町村レベルまでを抽出し、それ以降の詳細住所を除去することができる（秘匿化レベルを都道府県または市区町村から選択できる）。

本機能は、オープンデータ公開時の個人情報保護を目的とし、住所の粒度を制御するテーブルアクションである。

本システム機能のフローチャート

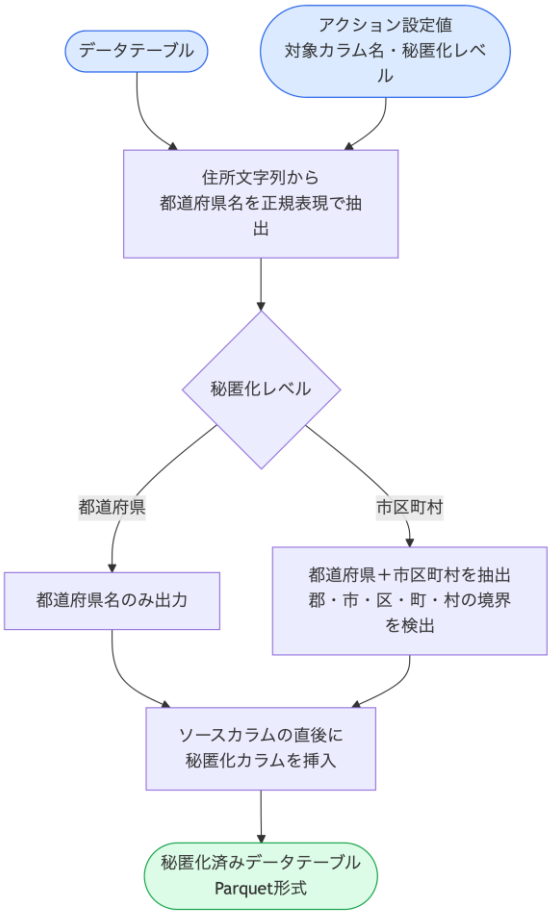


図 2.40　【FN039】のフローチャート

本システム機能の処理の詳細

- **都道府県抽出**
  - ■ 処理内容：47 都道府県の正規表現パターン（「^(北海道|青森県|…|沖縄県)」）で住所文字列の先頭から都道府県名を抽出する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

- **市区町村抽出（市区町村レベル選択時）**
  - ■ 処理内容：都道府県を除いた残りの文字列から、「市」・「区」・「町」・「村」 の境界文字を検出して市区町村名までを切り出す。「郡」 を含む住所（例：「○○郡○○町」）では郡名と町村名の両方を保持する。
  - ■ 利用するライブラリ：【SL009】DuckDB（node-api）
  - ■ 利用するアルゴリズム：なし

- **列挿入**
  - ■ 処理内容：元の住所カラムの直後に 「{カラム名}_秘匿化」 という名称の新規カラムを挿入する。NULL 値・空文字はそのまま保持する。

本システム機能の入出力データの仕様

- **入力**
  - ■ 処理対象のデータテーブル
    - ▲ データの形式：DuckDB テーブル
    - ▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

■ 秘匿化設定

▲ 対象カラム名：住所文字列を含むカラムを指定。

▲ 秘匿化レベル：「prefecture」（都道府県）または 「municipality」（市区町村）

● **出力**

■ 秘匿化済みデータテーブル（元カラムの直後に秘匿化カラムを追加。）

▲ データの形式：Apache Parquet

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN040】数値偏差値化＜新規開発＞

本システム機能の概要

職員が、数値カラムの偏差値（平均 50・標準偏差 10 の標準化スコア）を算出し、新規カラムとして追加することができる。

本機能は、データの相対的な位置付けの把握やオープンデータ公開時の数値統計化を目的とするテーブルアクションである。

本システム機能のフローチャート

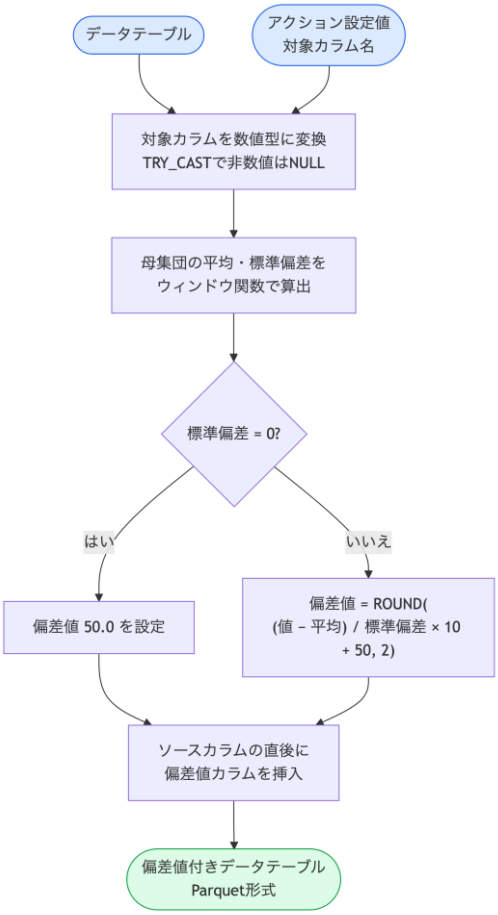


図 2.41　【FN040】のフローチャート

本システム機能の処理の詳細

● **偏差値算出**

■ 処理内容：対象カラムの値を 「TRY_CAST」 で DOUBLE 型に変換し、DuckDB のウィンドウ関数 「AVG()」 と 「STDDEV_POP()」（母集団標準偏差、ddof=0）で全行の平均と標準偏差を算出する。偏差値は 「(値 - 平均) / 標準偏差 × 10 + 50」 で計算し、「ROUND(..., 2)」 で小数第 2 位まで丸める。

■ 標準偏差が 0 の場合（全値同一）は偏差値 50.0 を返す。

■ 数値に変換できない値は NULL となり、統計計算から自動的に除外される。

■ 利用するライブラリ：【SL009】DuckDB（node-api）

■ 利用するアルゴリズム：なし

● **列挿入**

■ 処理内容：元の数値カラムの直後に 「{カラム名}_偏差値」 という名称の新規カラムを挿入する。

本システム機能の入出力データの仕様

● **入力**

■ 処理対象のデータテーブル

▲ データの形式：DuckDB テーブル

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

■ 対象カラム名：偏差値を算出する数値カラムを指定。

● **出力**

■ 偏差値付きデータテーブル（元カラムの直後に偏差値カラムを追加。）

▲ データの形式：Apache Parquet

▲ 利用するデータインターフェース：【IF006】GCS Parquet ファイル

【FN041】メタ情報自動生成＜新規開発＞

本システム機能の概要

職員が、スキーマ自動提案で提案されたフィールドの説明文等を活用し、メタ情報の入力を最小限に抑えることができる（フィールドの名称・型・説明文をシステムが自動補完する）。

本機能では、スキーマ自動提案（【FN007】）の過程で提案されたカラム名・データ型・カラムの説明文をメタデータとして保持し、データセットのメタデータ入力画面で自動的に補完される。G 空間情報センターへのデータ登録時に必要なメタデータ項目（タイトル・説明・カテゴリ・ライセンス・空間範囲・時間範囲等）についても、スキーマ情報から保持したカラムの説明文を自動で埋めることで入力補助を行う。

本システム機能の入力・処理・出力のフローチャート

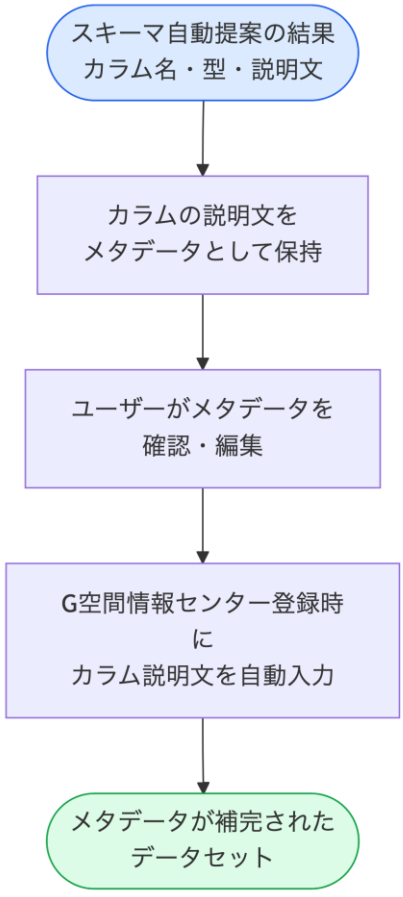


図 2.42　【FN041】のフローチャート

本システム機能の処理の詳細

- **スキーマ情報の引き継ぎ**
  - ■ 処理内容：スキーマ自動提案（【FN007】）で生成されたカラム名・データ型・説明文をデータセットのメタデータとして保持する。ユーザーがメタデータ入力画面を開いた際に、保持された説明文が自動的に表示される。
  - ■ 利用するアルゴリズム：なし

- **G 空間情報センター登録時の自動補完**
  - ■ 処理内容：G 空間情報センターへのデータ登録時に、保持されたカラムの説明文をメタデータ項目に自動で入力する。ユーザーは GUI 上で内容を確認・修正し、最終的なメタ情報として確定する。

【FN042】データエクスポート・ダウンロード＜新規開発＞

本システム機能の概要

職員が、データセット・データテーブル・アセットを CSV / JSON / GeoJSON 形式でエクスポート・ダウンロードすることができる。

本機能では、サーバーサイドで事前生成した CSV キャッシュ（PostgreSQL の「csv_cache」 カラムに保存）を利用して高速配信する。キャッシュが存在しない場合は DuckDB SQL を実行してリアルタイムに CSV を生成する。

本システム機能の入力・処理・出力のフローチャート

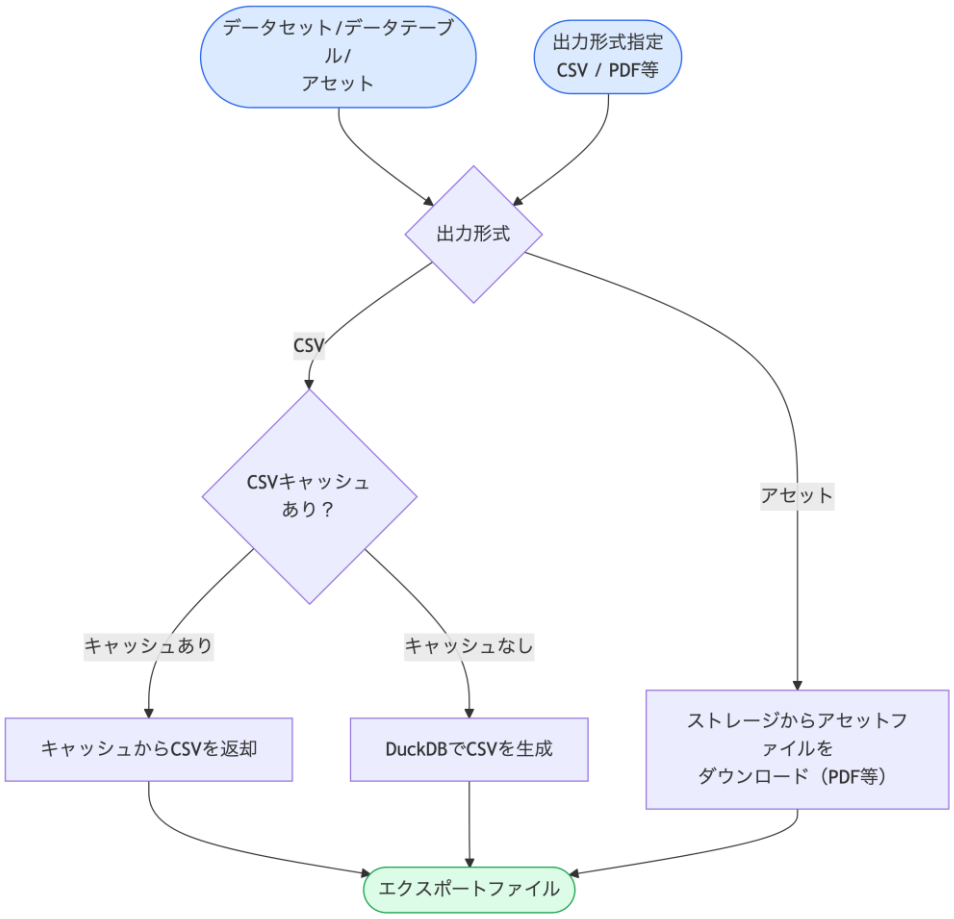


図 2.43 【FN042】のフローチャート

本システム機能の処理の詳細

- **CSV エクスポート**
  - 処理内容：データセットまたはデータテーブルを CSV 形式でエクスポートする。サーバーサイドで事前生成した CSV キャッシュが存在する場合はキャッシュを返し、存在しない場合は DuckDB SQL を実行して CSV を生成する。
  - 利用するライブラリ：【SL009】DuckDB（node-api）
- **アセットダウンロード**
  - 処理内容：GCS に保存されたアセットファイル（PDF 等）をダウンロードする。
  - 利用するライブラリ：【SL008】@google-cloud/storage

本システム機能の入出力データの仕様

- **入力**
  - データセット ID またはデータテーブル ID またはアセット ID・出力形式指定（CSV /アセット）
    - データの形式：リクエストパラメータ
    - 利用するデータインターフェース：【IF007】PostgreSQL データアクセス、【IF005】GCS アセットファイル
- **出力**
  - エクスポートファイル（CSV / PDF 等）
    - データの形式：CSV（UTF-8）またはアセットの元ファイル形式
    - 利用するデータインターフェース：HTTP レスポンスストリーム

【FN043】オープンデータ化＜新規開発＞

本システム機能の概要

職員が、G 空間情報センターへのオープンデータ化対応のステータス管理を行うことができる（内部連携や G 空間情報センターとの API 連携により、システムで管理されているメタデータやデータ本体を G 空間情報センターへ連携し、データ公開を行う）。

本機能では、ステータス管理として「公開準備」→「公開（LINKS Veda から G 空間情報センターへの連携）」→「G 空間情報センターでの再公開処理」の流れで進行する。LINKS Veda からの公開処理により、G 空間情報センター上にプライベートなページとしてデータが登録される。その後、G 空間情報センター側で公開状態を変更して一般公開にする処理が別途必要となる。編集者のロールを持つユーザーが公開処理を実行できる。

本システム機能の入力・処理・出力のフローチャート

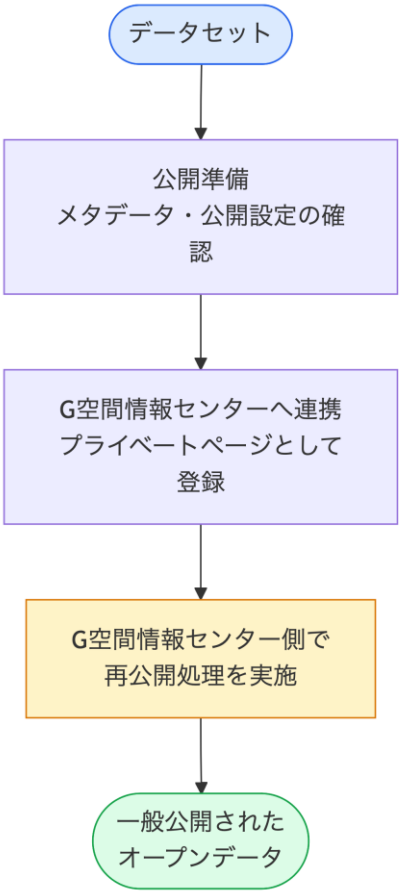


図 2.44　【FN043】のフローチャート

本システム機能の処理の詳細

- **公開準備**
  - 処理内容：データセットのメタデータ・公開設定を確認し、G 空間情報センターへの連携に必要な情報を整備する。編集者のロールを持つユーザーが実行する。
  - 利用するアルゴリズム：なし

- **G 空間情報センターへの連携**
  - 処理内容：LINKS Veda から G 空間情報センターの API を通じてデータとメタデータを連携する。連携されたデータは G 空間情報センター上でプライベートなページとして登録される。
  - 利用するアルゴリズム：なし

- **G 空間情報センターでの再公開処理**
  - 処理内容：G 空間情報センター側で管理者がプライベートページの公開状態を変更し、一般公開とする。本処理は G 空間情報センター（外部システム）側の操作である。

本システム機能の入出力データの仕様

- **入力**
  - データセット（メタデータ・公開設定を含む）
    - データの形式：JSON
    - 利用するデータインターフェース：【IF007】PostgreSQL データアクセス

- **出力**
  - ■ G 空間情報センターへの連携データ
    - ▲ データの形式：API リクエスト（JSON）
    - ▲ 利用するデータインターフェース：G 空間情報センター連携 API（外部システムインターフェース。詳細は IF セクションを参照）

【FN044】データ配信 API＜既存改修＞

本システム機能の概要

職員が、認証済みアクセストークンによるアクセス制御のもと、特定のチームに所属するデータテーブルを CSV または Parquet 形式で API 経由で取得することができる。

本機能では、API キー認証によるアクセス制御を行い、認証されたリクエストに対してデータテーブルを CSV または Parquet 形式で返却する。返却されたデータテーブルは QGIS 等の GIS ツールで連携し、分析・可視化に利用されることを想定している。

本システム機能の入力・処理・出力のフローチャート

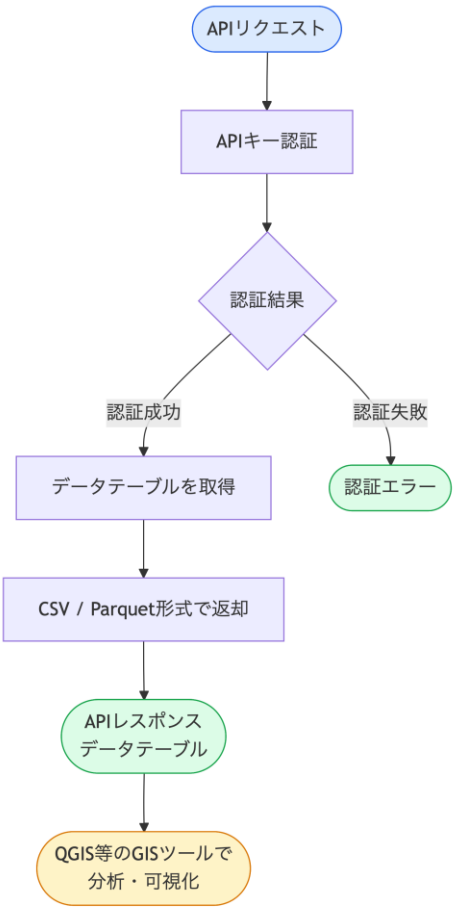


図 2.45 【FN044】のフローチャート

本システム機能の処理の詳細

● **API キー認証**

■ 処理内容：リクエストに含まれる API キーを検証し、対象チームへのアクセス権限を確認する。認証に失敗した場合はエラーを返却する。

■ 利用するアルゴリズム：なし

● **データテーブル返却**

■ 処理内容：認証されたリクエストに対して、指定されたデータテーブルを CSV または Parquet 形式で返却する。

■ 利用するライブラリ：【SL009】DuckDB（node-api）、【SL008】@google-cloud/storage

本システム機能の入出力データの仕様

● **入力**

■ API リクエスト（API キー・対象データテーブル指定）

▲ データの形式：HTTP リクエスト

● **出力**

■ データテーブル（CSV / Parquet 形式）

▲ データの形式：CSV（UTF-8）または Apache Parquet

▲ 利用するデータインターフェース：HTTP レスポンスストリーム

【FN045】ドキュメントインデキシング・ベクトル検索＜新規開発＞

本システム機能の概要

職員が、OCR 処理済みドキュメントに対して、キーワードだけでなく意味的な類似性でも検索することができる（ベクトル検索とキーワード検索を統合したハイブリッド検索を用いる）。

本機能では、データ構造化処理（【FN008】）の後段で OCR 処理済みテキストを自動的にインデキシングし、自然言語によるドキュメント検索を可能にする。具体的には、LLM によるメタデータ自動抽出（タイトル・要約・タグ・セクション要約）、階層的チャンキング（ドキュメント・セクション・パラグラフの 3 レベル）、Embeddings 処理によるベクトル生成、PostgreSQL + pgvector（【CO026】）へのベクトル・メタデータ格納、RRF（Reciprocal Rank Fusion）ハイブリッド検索（セマンティック検索+キーワード検索の統合）の処理を実行する。ファイル検索機能（【FN116】）が本機能のハイブリッド検索を利用する。なお、国会答弁機能（【FN110】）ではベクトル検索は使用せず、AI が自律的に答弁書データからテキスト検索を行う方式を採用している。

本システム機能の入力・処理・出力のフローチャート

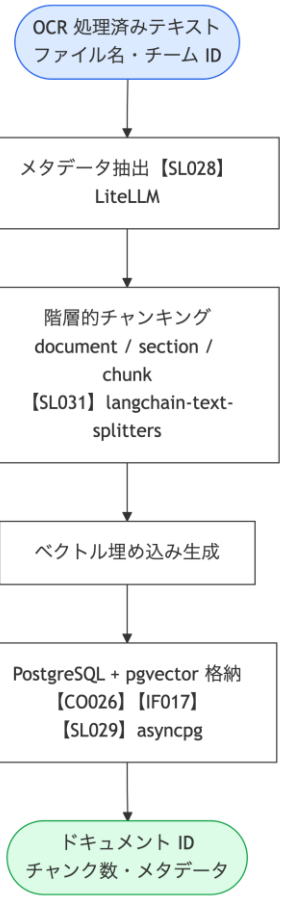


図 2.46 【FN045】のフローチャート（インデキシングパイプライン）

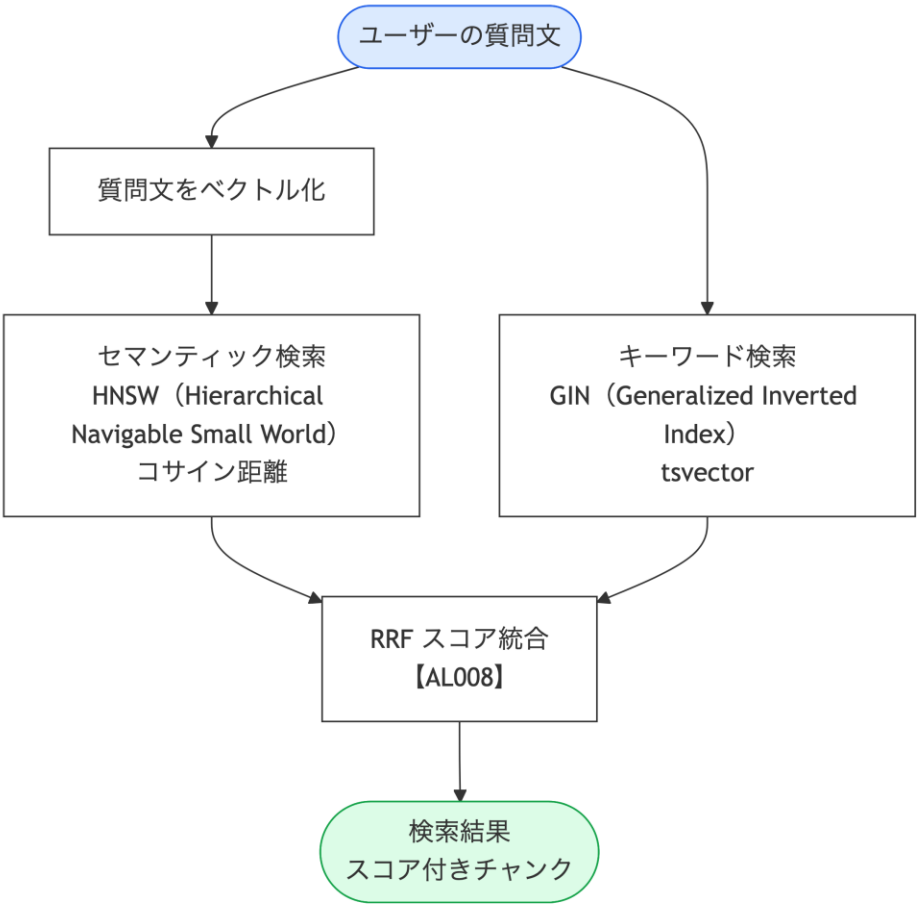


図 2.47　【FN045】のフローチャート（ハイブリッド検索）

本システム機能の処理の詳細

- **メタデータ抽出**
  - 処理内容：OCR テキスト（先頭 8000 文字）を LLM に送信し、ドキュメントタイトル・要約（200 字以内）・タグ（最大 10 個）・セクション要約を JSON 形式で抽出する。最大 3 回リトライする。
  - 利用するライブラリ：【SL028】LiteLLM
  - 利用するデータインターフェース：【IF012】AWS Bedrock API（Claude）

- **階層的チャンキング**
  - 処理内容：Markdown 見出し・日本語見出し（「第 N 章」「N.」等）を検出してセクションに分割し、各セクションを RecursiveCharacterTextSplitter（1000 文字/200 文字オーバーラップ）でチャンクに分割する。ドキュメント要約・セクション要約・パラグラフチャンクの 3 レベル階層を構築する。
  - 利用するライブラリ：【SL031】langchain-text-splitters
  - 利用するアルゴリズム：【AL008】階層的ドキュメントインデキシング・RRF ハイブリッド検索
- **ベクトル埋め込み生成**
  - 処理内容：全チャンクのテキストを Embeddings 処理を行いベクトルに変換する。20 件/バッチで処理し、レート制限・タイムアウト時は指数バックオフでリトライする（最大 5 回）。
  - 利用するライブラリ：【SL028】LiteLLM
  - 利用するデータインターフェース：【IF012】AWS Bedrock API

- **ベクトルストア格納**
  - ■ 処理内容：ドキュメントメタデータ・セクション情報・チャンクおよびベクトルを PostgreSQL + pgvector に格納する。既存ドキュメントの再インデキシング時は UPSERT で上書きする。
  - ■ 利用するライブラリ：【SL029】asyncpg、【SL030】pgvector
  - ■ 利用するデータインターフェース：【IF017】pgvector ベクトルストアインターフェース

- **ハイブリッド検索（検索時）**
  - ■ 処理内容：クエリテキストをベクトル化し、セマンティック検索（HNSW コサイン距離）とキーワード検索（GIN tsvector）の結果を RRF（Reciprocal Rank Fusion、k=60）で統合する。チーム ID・タグ・ドキュメント種別・日時範囲でフィルタリングできる。
  - ■ 利用するアルゴリズム：【AL008】階層的ドキュメントインデキシング・RRF ハイブリッド検索
  - ■ 利用するデータインターフェース：【IF017】pgvector ベクトルストアインターフェース

本システム機能の入出力データの仕様

- **入力**
  - OCR 処理済みテキスト（Markdown 形式）
    - データの内容：データ構造化処理（【FN008】）で OCR 済みの全文テキスト
    - データの形式：文字列（Markdown）
    - 利用するデータインターフェース：【IF002】VD009 構造化処理 API（structure-job 内で連続実行）
  - ファイルメタデータ（ファイル名・チーム ID・ワークフローID・ドキュメント種別）
    - データの内容：ファイルメタデータ
    - データの形式：JSON
- **出力**
  - インデキシング結果
    - データの内容：ドキュメント ID・チャンク数・抽出タイトル・要約・タグ・セクション情報
    - データの形式：JSON
    - 利用するデータインターフェース：【IF004】WorkflowRun コールバック API（document_id を含めてコールバック）
  - 検索結果（ハイブリッド検索 API 経由）
    - データの内容：スコア付きチャンク・ドキュメントメタデータ
    - データの形式：JSON

▲ 利用するデータインターフェース：【IF017】pgvector ベクトルストアインターフェース（検索 API としても利用。）

【FN046】ページ分割＜新規開発＞

本システム機能の概要

職員が、1 ファイル内に複数の文書が混在している場合に、ページ数またはキーワードで複数ファイルに分割することができる。

本機能は、主に PDF を対象とし、行政文書では複数種類の帳票が 1 つの PDF にまとめてスキャンされるケースが多く、ワークフローごとに異なるスキーマを適用するために事前の文書分割が必要となる。

本システム機能の入力・処理・出力のフローチャート

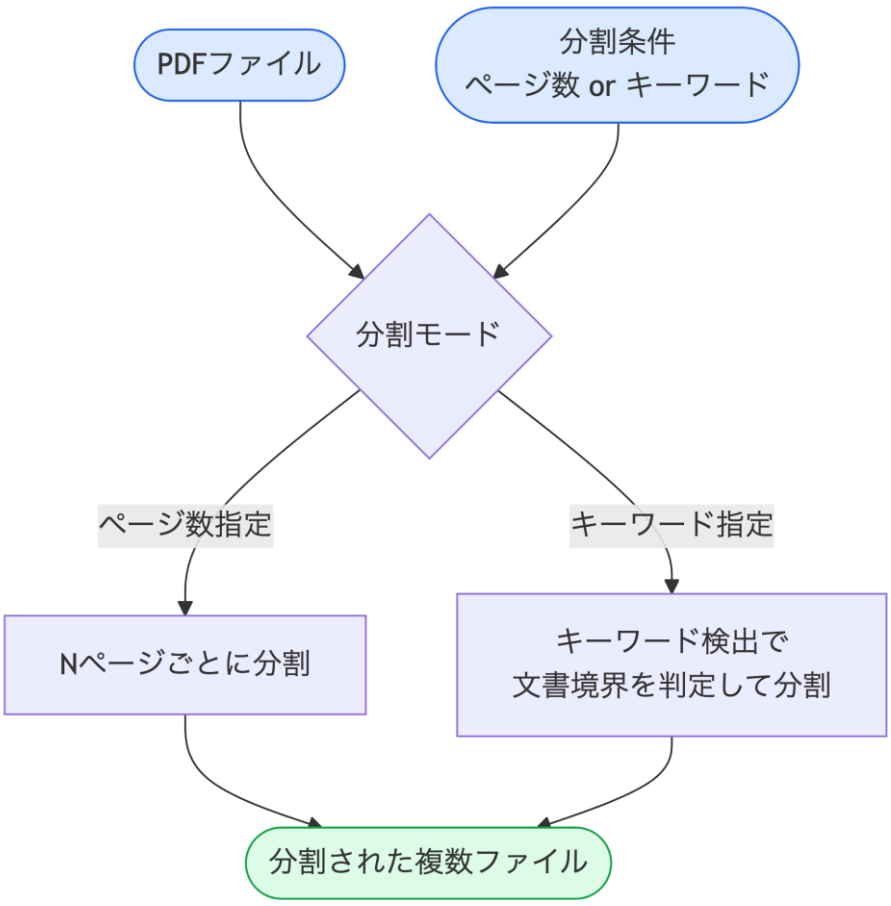


図 2.48　【FN046】のフローチャート

本システム機能の処理の詳細

- **ページ数指定分割**
  - ■ 処理内容：N ページごとに PDF を分割する。たとえば「3 ページごと」と指定すると、p.1-3 / p.4-6 / p.7-9 の各ファイルに分割される。

- **キーワード指定分割**
  - ■ 処理内容：指定したキーワード（表題・帳票名等）が含まれるページを区切りとして分割する。OCR テキストからキーワードの出現位置を検出し、文書境界を自動判定する。

【FN101】プロジェクト管理＜新規開発＞

本システム機能の概要

管理者が、BI のプロジェクトを管理することができる（データ構築モジュールと同じ認証基盤で動作し、同一チーム内でユーザーに割り振られた権限に基づく認可処理を行う）。

本機能では、データ構築モジュールのチーム管理（【FN002】）と同一の認証基盤を共有しており、データ構築モジュールで発行されたセッション Cookie を LINKS BI バックエンドが検証する委譲方式で認証を実現する。ユーザーはデータ構築モジュールでログイン済みであれば、追加の認証操作なしに LINKS BI にアクセスできる。ワークスペースはチーム単位で分離されており、チームメンバーのみがワークスペース内のダッシュボード・データソース・AI チャット履歴にアクセスできる。ロール（編集者・閲覧者）に応じた操作権限が適用される。

本システム機能の入力・処理・出力のフローチャート

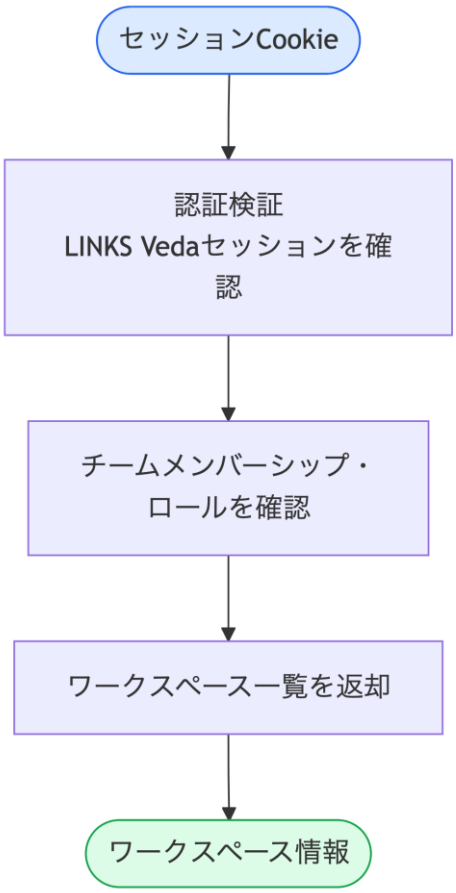


図 2.49　【FN101】のフローチャート

本システム機能の処理の詳細

- **認証連携**
  - ■ 処理内容：Hono バックエンドの認証ミドルウェアがリクエスト Cookie を検証し、データ構築モジュールの PostgreSQL（CO002）でチームメンバーシップとロールを確認する。認証に失敗した場合は 401 エラーを返す。
  - ■ 利用するライブラリ：【SL114】Hono、【SL116】Kysely

- **ワークスペース一覧取得**
  - ■ 処理内容：認証済みユーザーが所属するチームの一覧を返す。各チームについて、利用可能なデータソース数・ダッシュボード数を集計して返す。
  - ■ 利用するインターフェース：【IF010】LINKS BI 汎用 REST API

【FN102】データインポート・データソース管理＜新規開発＞

本システム機能の概要

職員が、データ構築モジュールで管理されているデータテーブルを選択し、分析対象のデータとしてインポートすることができる（複数ファイルのインポートにも対応する。インポートしたデータソースの一覧管理も行える）。

本機能では、インポートされたデータは Parquet ファイルとしてブラウザ内の DuckDB-Wasm に取り込まれ、OPFS（Origin Private File System）を使ってブラウザ内に一時保持される。ただし OPFS のデータはブラウザのリロード時に削除されるため、永続化ではない。OPFS 非対応環境ではオンメモリで動作する。データ構築モジュールで管理されているデータテーブルがデータソース候補として表示され、ユーザーがワンクリックでインポートできる。

本システム機能の入力・処理・出力のフローチャート

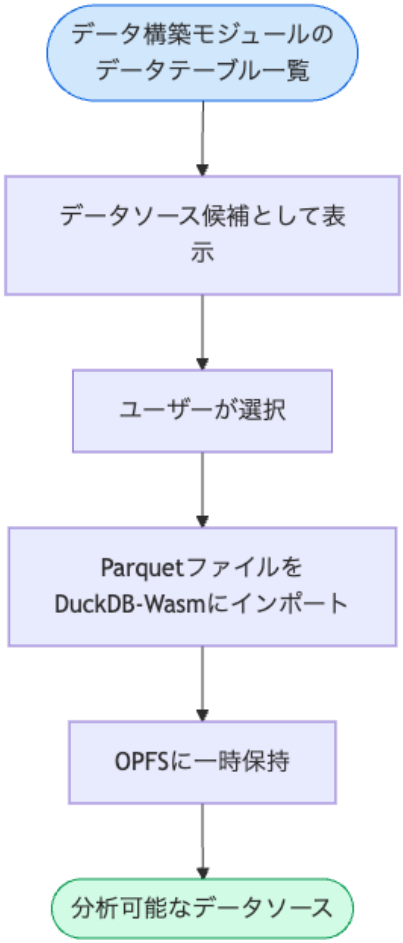


図 2.50 【FN102】のフローチャート

本システム機能の処理の詳細

- **Veda データソースの自動連携**
  - ■ 処理内容：データ構築モジュールの公開済みテーブル一覧を取得し、データソース候補として表示する。ユーザーが選択すると、GCS 上の Parquet ファイル URL を取得して DuckDB-Wasm にインポートする。
  - ■ 利用するライブラリ：【SL102】@duckdb/duckdb-wasm、【SL109】apache-arrow

- **OPFS 一時保持**
  - ■ 処理内容：インポート済みデータを OPFS に一時保持する。OPFS のデータはブラウザのリロード時に削除される。

- **バックエンド永続化**
  - ■ 処理内容：インポートしたデータソースのメタデータ（SQL 定義・テーブル名等）をバックエンド経由で PostgreSQL（CO014）に永続化する。ページを開いた際にバックエンドからメタデータを読み込み、DuckDB-Wasm 上のデータソースを復元する。
  - ■ 利用するインターフェース：【IF010】LINKS BI 汎用 REST API

【FN103】データセットプレビュー＜新規開発＞

本システム機能の概要

職員が、選択したデータセットをテーブル形式や地図等でプレビューすることができる。

本機能では、DuckDB-Wasm がブラウザ内で Parquet データに対して SQL クエリを実行し、結果を取得してテーブル UI に表示する。テーブルビューでは、カラム名・データ型・先頭 N 行のプレビューを表示する。マップビューでは、ジオメトリデータを地図上にプロットする。テーブルビュー・マップビューはデータの内容に関わらず両方利用可能である。

本システム機能の入力・処理・出力のフローチャート

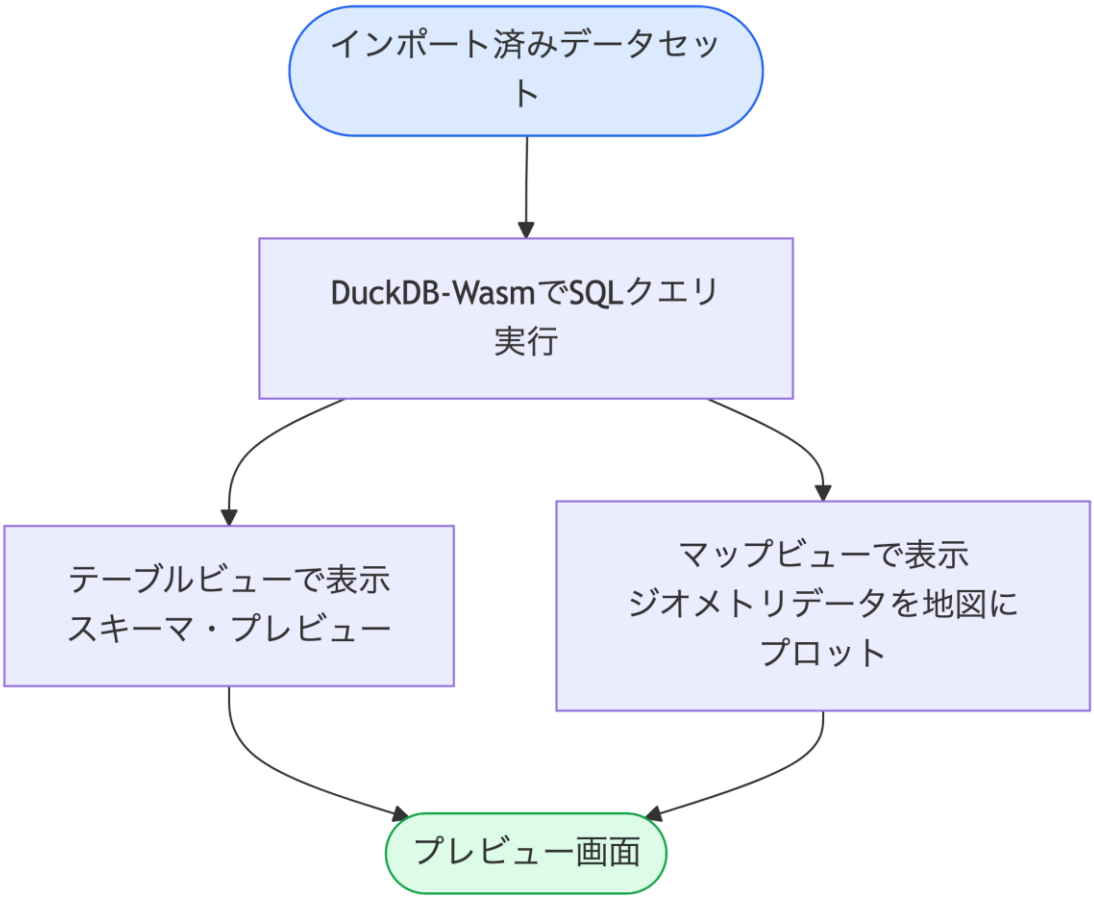


図 2.51　【FN103】のフローチャート

本システム機能の処理の詳細

- **テーブル情報取得**
  - 処理内容：DuckDB-Wasm を利用してテーブルの情報を取得する。スキーマ情報・プレビューデータを取得し、テーブル UI に表示する。
  - 利用するライブラリ：【SL102】@duckdb/duckdb-wasm、【SL109】apache-arrow

【FN104】ダッシュボード管理＜新規開発＞

本システム機能の概要

職員が、グリッドレイアウト上にウィジェット（グラフ・マップ・テキスト・テーブル）を配置・保存することができる。

本機能では、react-grid-layout によるドラッグ&ドロップ対応のグリッドレイアウトで、ウィジェットを自由に配置・リサイズ・並べ替えできる。ダッシュボード定義（ウィジェット配置・サイズ・設定）はデータベースに保存される。

本システム機能の入力・処理・出力のフローチャート

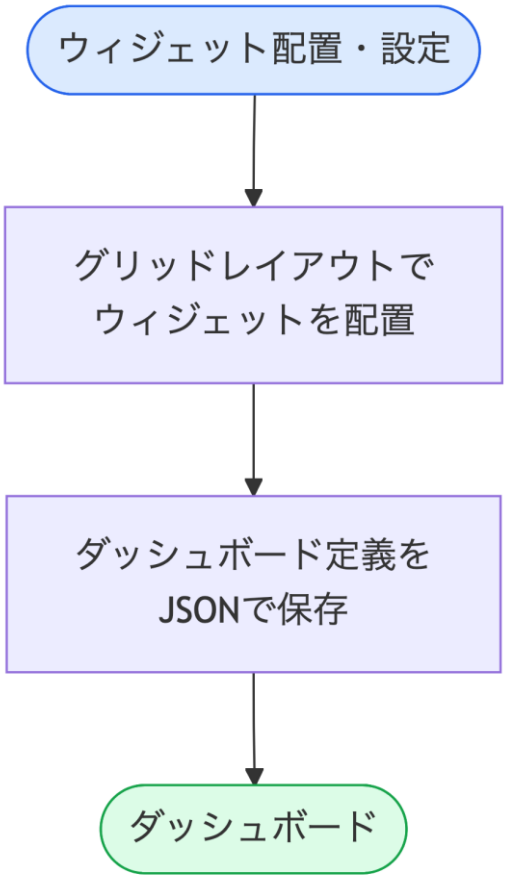


図 2.52　【FN104】のフローチャート

本システム機能の処理の詳細

- **ダッシュボードの作成・一覧表示・更新・削除**
  - ■ 処理内容：ダッシュボードの作成・更新・削除・一覧取得を行う。更新時にはウィジェット配置と各ウィジェットの設定を保存する。
  - ■ 利用するライブラリ：【SL104】react-grid-layout
  - ■ 利用するインターフェース：【IF010】LINKS BI 汎用 REST API

【FN105】グラフ可視化＜新規開発＞

本システム機能の概要

職員が、棒グラフ・折れ線グラフ・散布図・円グラフ・ヒストグラム・積み上げ棒グラフなど多様なグラフ形式でデータを可視化することができる（色やフォントサイズ等のスタイルを変更することも可能）。

本機能では、Vega-Lite（宣言的データビジュアライゼーション文法）を使ってグラフを作成する。グラフの定義は JSON 形式の Vega-Lite スペックとして記述され、AI チャット（【FN109】）から Function calling 経由で自動生成することも、ユーザーが GUI 上でノーコード編集することも可能である。Vega-Lite に自作の DuckDB ローダーを登録し、グラフ表示時に Vega が DuckDB-Wasm から JSON 形式でデータを取得してリアルタイムにグラフをレンダリングする。条件フィルタリング・ソート・凡例操作にも対応する。

本システム機能の入力・処理・出力のフローチャート

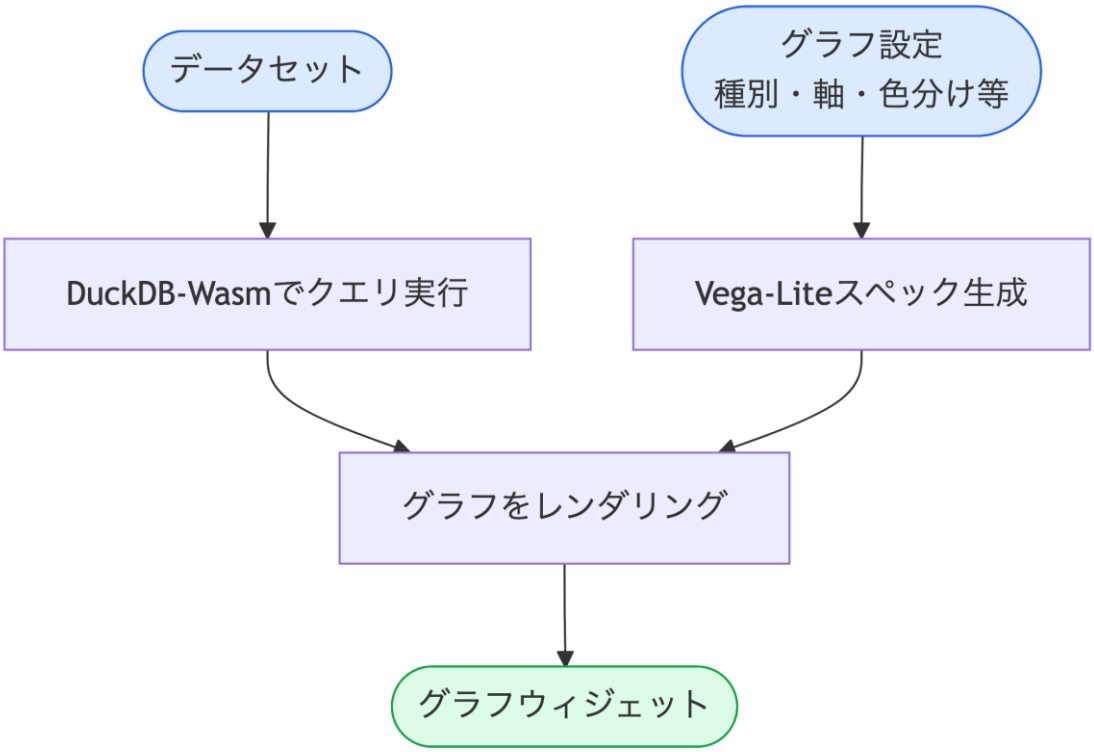


図 2.53　【FN105】のフローチャート

本システム機能の処理の詳細

- **グラフスペック生成**
  - 処理内容：AI チャットの 「update_vega_graph_spec_for_table」 ツールが、テーブルスキーマとユーザーの要求に基づいて Vega-Lite JSON スペックを生成する。生成されたスペックは PostgreSQL の 「dashboards」 テーブルに保存される。
  - 利用するライブラリ：【SL103】vega / vega-lite
- **ノーコード編集**
  - 処理内容：GUI 上でグラフ種別・軸設定・色分け・フィルター条件をドロップダウン・スライダー等で変更する。変更は Vega-Lite スペックに反映され、リアルタイムにグラフが更新される。
- **ソート順の特殊処理**
  - 処理内容：和暦（H30.11 等）を含むデータの場合、数値ソートキーを自動生成して正しい時系列順でグラフを描画する。

【FN106】マップ可視化＜新規開発＞

本システム機能の概要

職員が、地図ライブラリを利用して、【FN102】でインポートしたデータを地図上に可視化することができる（レイヤーの追加・スタイルの編集・複数レイヤーの重畳表示・凡例の表示・地物クリック時のポップアップ表示・フィルタリングの適用・時系列表示に対応する）。

本機能では、MapLibre GL JS を使ってインタラクティブな地図上にデータを描画する。DuckDB の空間拡張を利用してブラウザ内でベクタータイルを動的に生成し、MapLibre GL JS がタイルを描画する。メッシュコード（JIS X 0410）にも対応しており、メッシュのグリッドポリゴンを自動生成・描画できる。

本システム機能の入力・処理・出力のフローチャート

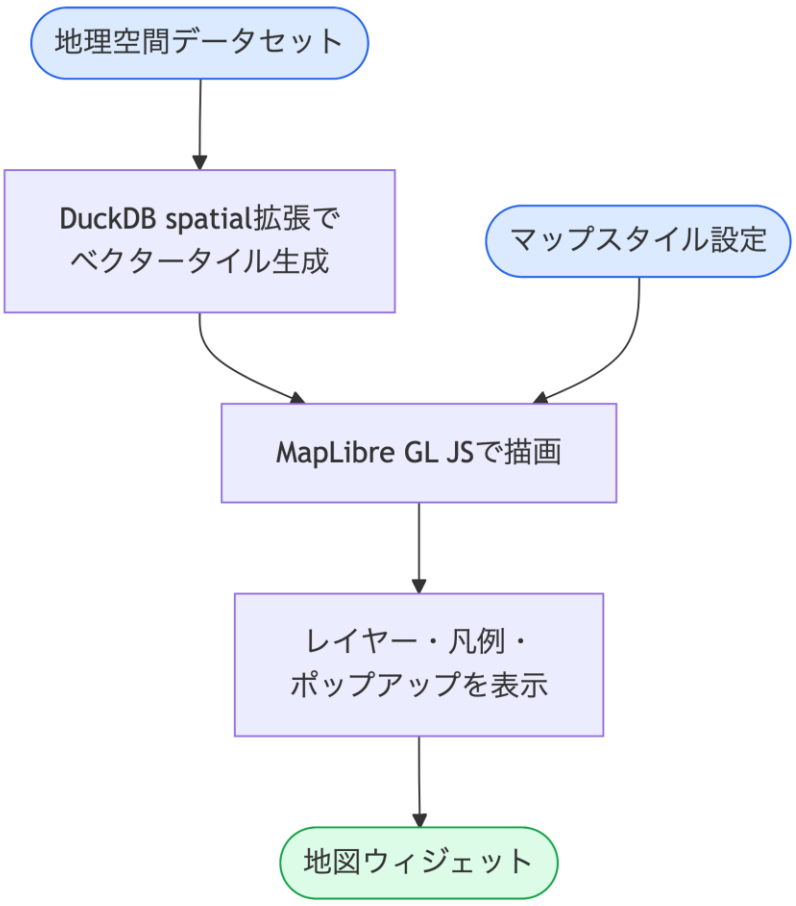


図 2.54　【FN106】のフローチャート

本システム機能の処理の詳細

- **ベクタータイル生成**
  - 処理内容：DuckDB-Wasm の空間拡張でジオメトリデータをベクタータイル形式に変換し、表示範囲に必要なデータのみを動的に生成する。
  - 利用するライブラリ：【SL102】@duckdb/duckdb-wasm（spatial 拡張）

- **スタイルシステム**
  - 処理内容：AI チャットまたはユーザーのノーコード操作により、地図のスタイル設定（色分け・線幅・透明度等）を編集する。凡例はスタイル設定から自動生成する。
  - 利用するライブラリ：【SL108】MapLibre GL JS（BI）

- **メッシュコード描画**
  - 処理内容：メッシュコード値の桁数から自動的にメッシュ次数（1 次〜4 次）を判定し、各メッシュの緯度経度範囲を SQL で計算してグリッドポリゴンを生成する。ジオメトリカラムが存在しないデータでも、メッシュコードカラムさえあれば地図上にグリッド表示できる。

【FN107】テーブルフィルタリング＜新規開発＞

本システム機能の概要

職員が、テーブルウィジェットに対して、ダッシュボード上でカラムの条件による並び替えや絞り込みを行うことができる。

本機能では、テキスト検索・数値範囲指定・日付範囲指定・カテゴリ選択等のフィルター条件を設定できる。DuckDB-Wasm の SQL クエリに WHERE 句・ORDER BY 句を動的に追加する方式で実装されており、ブラウザ内でリアルタイムにフィルタリング結果を反映する。

本システム機能の入力・処理・出力のフローチャート

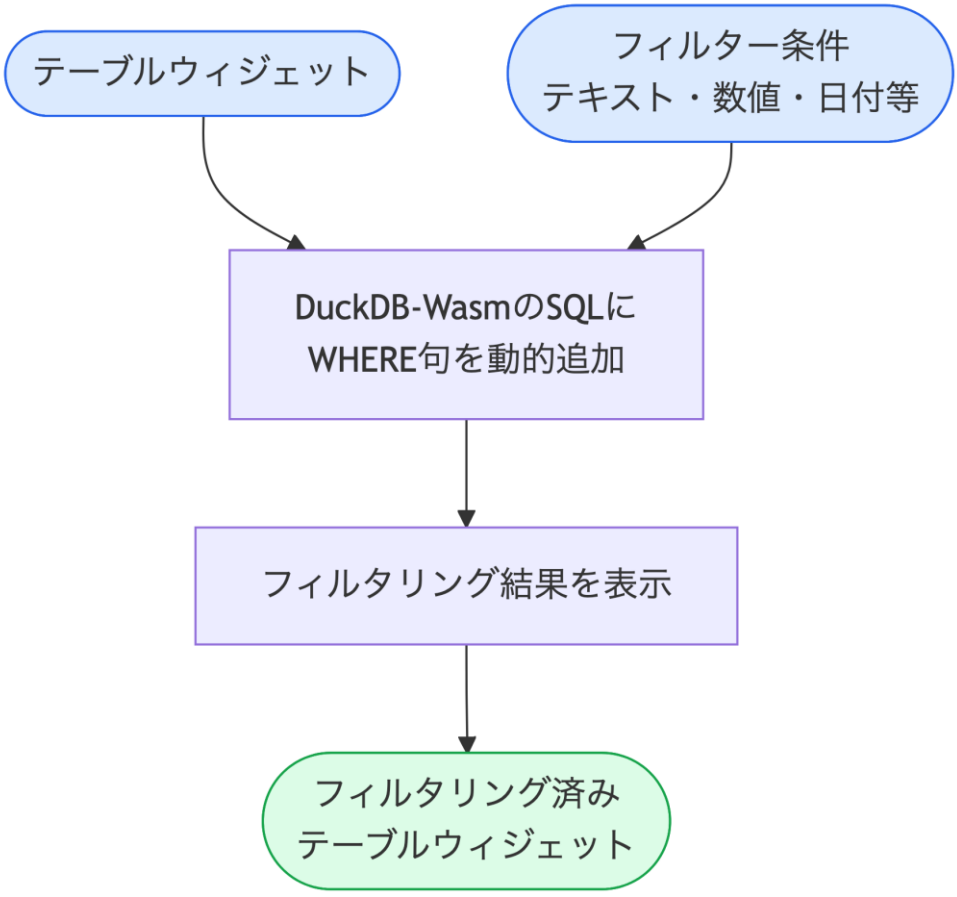


図 2.55　【FN107】のフローチャート

【FN108】演算モジュール＜新規開発＞

本システム機能の概要

職員が、インポートしたデータセットに対して、カラム同士の四則演算・集計・統計値の算出・空間集計などを、ノーコードで実行することができる。

本機能は、ブラウザ内の DuckDB-Wasm で SQL を実行し、サーバーサイド処理を伴わずに完結する。

本システム機能の入力・処理・出力のフローチャート

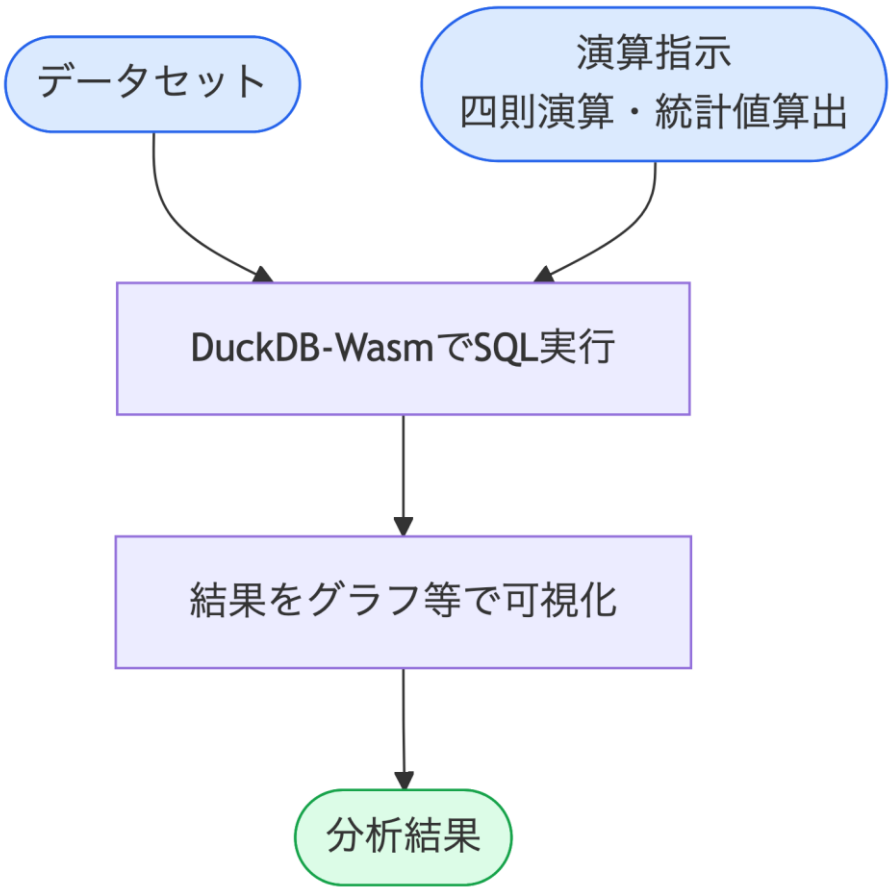


図 2.56　【FN108】のフローチャート

【FN109】AI チャット＜新規開発＞

本システム機能の概要

職員が、エージェント AI との対話を通して、自然言語でデータの分析・可視化等を実行することができる。

本機能では、LLM の Function Calling を活用し、DuckDB-Wasm と連携することで自然言語によるデータの分析・可視化を実現する。HTTP ストリーミング（ai-sdk 形式）でリアルタイムにレスポンスを表示する。

本システム機能の入力・処理・出力のフローチャート

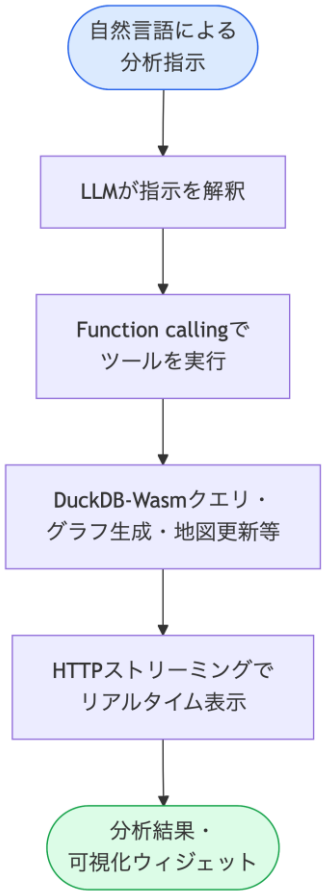


図 2.57　【FN109】のフローチャート

本システム機能の処理の詳細

- **スキルシステム**
  - ■ 処理内容：「get_skill」 ツールでスキルファイル（duckdb/vega/map/regression/estat 等）を動的取得し、AI の回答品質を向上させる。
  - ■ 利用するアルゴリズム：なし（スキルファイルに基づくプロンプト制御）

- **Function calling**
  - ■ 処理内容：LLM が Function Calling でツール（データクエリ・グラフ生成・地図スタイル更新等）を呼び出し、フロントエンドで実行する。
  - ■ 利用するライブラリ：【SL105】@ai-sdk/react、【SL106】ai（Vercel AI SDK）、【SL117】@ai-sdk/amazon-bedrock

- **回帰分析**
  - ■ 処理内容：「select_predictors」 → 「perform_regression」 → 「perform_segmented_regression」 の順で線形回帰・セグメント回帰を実行する。
  - ■ 利用するアルゴリズム：【AL006】線形回帰分析

【FN110】国会答弁機能＜新規開発＞

本システム機能の概要

職員が、過去答弁を読み込んだ状態で、自然言語で答弁案生成を依頼し、過去答弁の内容を踏まえた答弁案を作成することができる（答弁案の生成に際しては、過去答弁への参照情報も含めて提示する）。

本システム機能の入力・処理・出力のフローチャート

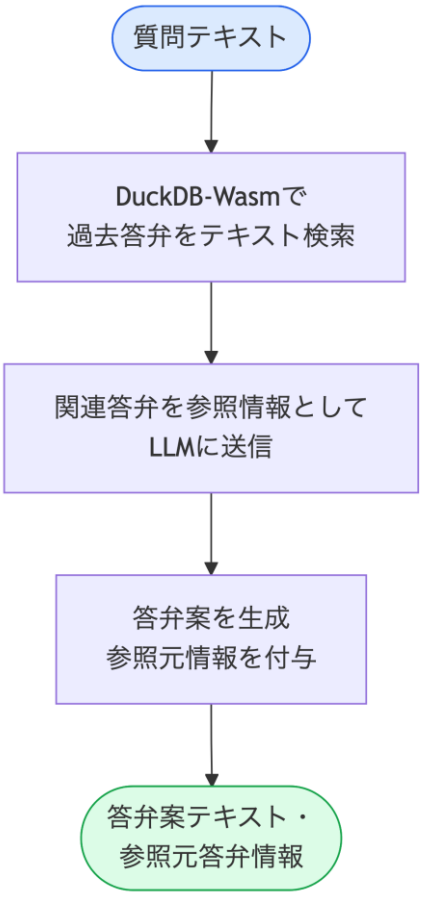


図 2.58　【FN110】のフローチャート

本システム機能の処理の詳細

- **過去答弁検索**
  - 処理内容：DuckDB-Wasm を利用して、インポート済みの国会答弁データに対してテキスト類似度検索を実行する。ユーザーが入力した質問に関連する過去答弁を検索して取得する。
  - 利用するライブラリ：【SL102】@duckdb/duckdb-wasm
  - 利用するアルゴリズム：なし

- **答弁案生成**
  - 処理内容：検索でヒットした過去答弁を参照情報として LLM に送信し、答弁案を生成する。参照元の答弁への出典情報を付与する。
  - 利用するアルゴリズム：なし

本システム機能の入出力データの仕様

- **入力**
  - ユーザーの質問テキスト
    - データの形式：テキスト
  - 国会答弁データ（DuckDB-Wasm にインポート済み。）
    - データの形式：DuckDB テーブル
- **出力**
  - 答弁案テキスト・参照元答弁情報
    - データの形式：Markdown テキスト

【FN111】統計分析処理＜新規開発＞

本システム機能の概要

職員が、K-Means クラスタリングや回帰分析等を使った発展的な統計処理・統計解析を実行することができる。

本機能は、AI チャット（【FN109】）のツール呼び出しとして実行される。

本システム機能の入力・処理・出力のフローチャート

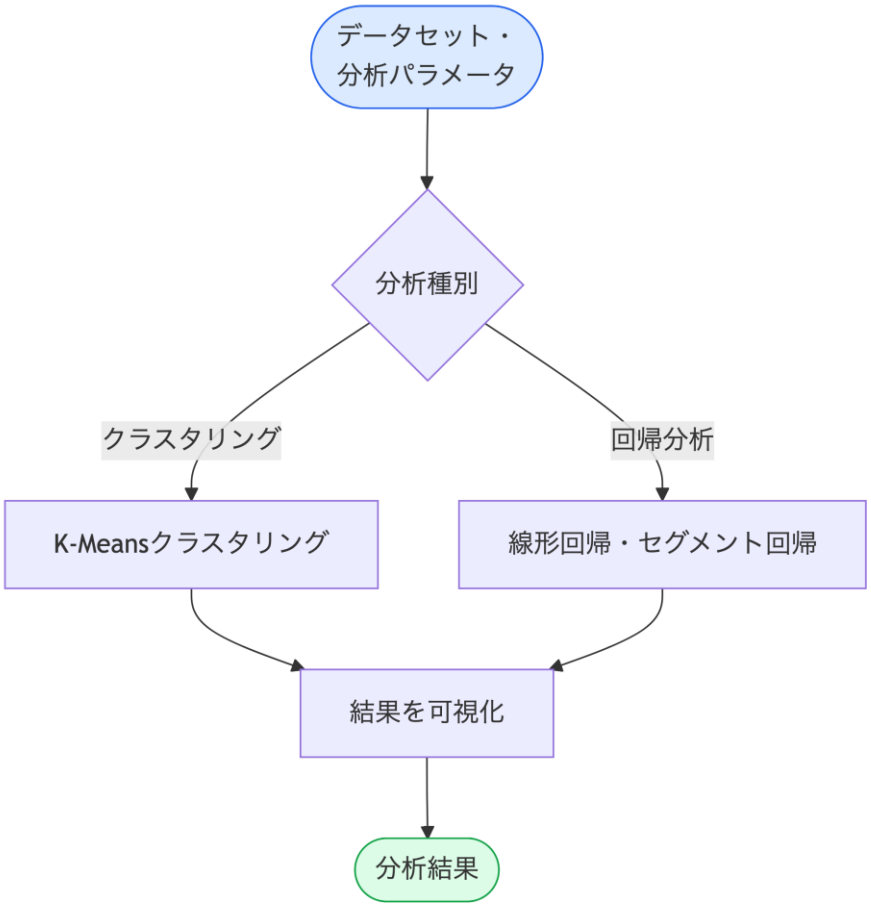


図 2.59　【FN111】のフローチャート

本システム機能の処理の詳細

- **クラスタリング分析**
  - 処理内容：K-Means アルゴリズムを用いて、指定カラムの数値データをクラスタに分類する。
  - 利用するライブラリ：ml-kmeans（7.0.0）
  - 利用するアルゴリズム：【AL005】K-Means クラスタリング

- **回帰分析**
  - 処理内容：線形回帰・セグメント回帰を実行し、データの傾向・予測を算出する。
  - 利用するライブラリ：jstat（1.9.6）、ml-matrix（6.12.1）
  - 利用するアルゴリズム：【AL006】線形回帰分析

本システム機能の入出力データの仕様

- **入力**
  - 分析対象のデータセット・分析パラメータ
    - データの形式：JSON
- **出力**
  - 分析結果（クラスタ割当・回帰係数・予測値等）
    - データの形式：JSON

【FN112】外部データソース連携＜新規開発＞

本システム機能の概要

職員が、政府統計 e-Stat 内の人口データとの連携により、外部統計データを取得し分析処理に利用することができる。

本機能では、AI チャット（【FN109】）を通じて人口データの検索・取得を行う。取得した統計データは DuckDB-Wasm にテーブルとして登録され、他のデータセットと同様にグラフ・地図・AI チャットで分析できる。

本システム機能の入力・処理・出力のフローチャート

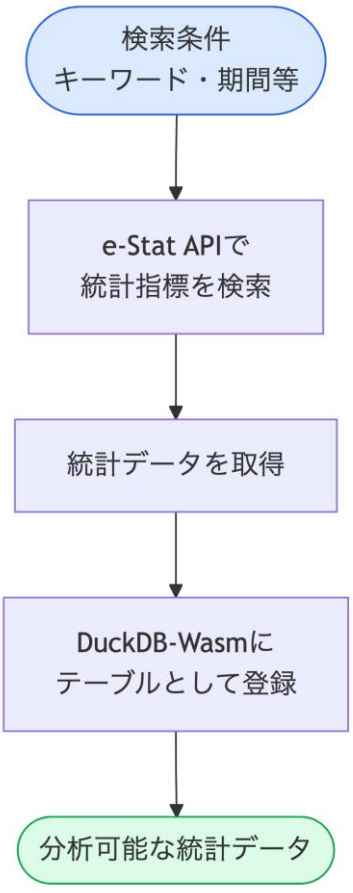


図 2.60　【FN112】のフローチャート

本システム機能の処理の詳細

- **統計指標検索**
  - ■ 処理内容：e-Stat の 「getStatsList」 API にキーワード・期間等の条件を送信し、該当する統計指標の一覧（統計 ID・名称・調査名・提供周期等）を取得する。
  - ■ 利用するインターフェース：【IF015】e-Stat API

- **統計データ取得**
  - ■ 処理内容：指定した統計 ID の 「getStatsData」 API を呼び出し、統計データ本体を JSON 形式で取得する。取得したデータを DuckDB-Wasm のテーブルとして登録する。
  - ■ 利用するインターフェース：【IF015】e-Stat API

【FN113】レポート出力＜新規開発＞

本システム機能の概要

職員が、ダッシュボードをレポートとして PNG 画像形式でダウンロードすることができる。

本機能では、html-to-image（【SL112】）ライブラリを使ってブラウザ内でダッシュボードの DOM をキャプチャし、PNG 画像としてダウンロードする。レポートテンプレート機能を備えており、表紙・目次・ヘッダー・フッター等の定型要素を含むフォーマットでレポートを出力できる。ダッシュボード上のすべてのウィジェット（グラフ・マップ・テーブル・テキスト）がレポートに含まれる。

本システム機能の入力・処理・出力のフローチャート

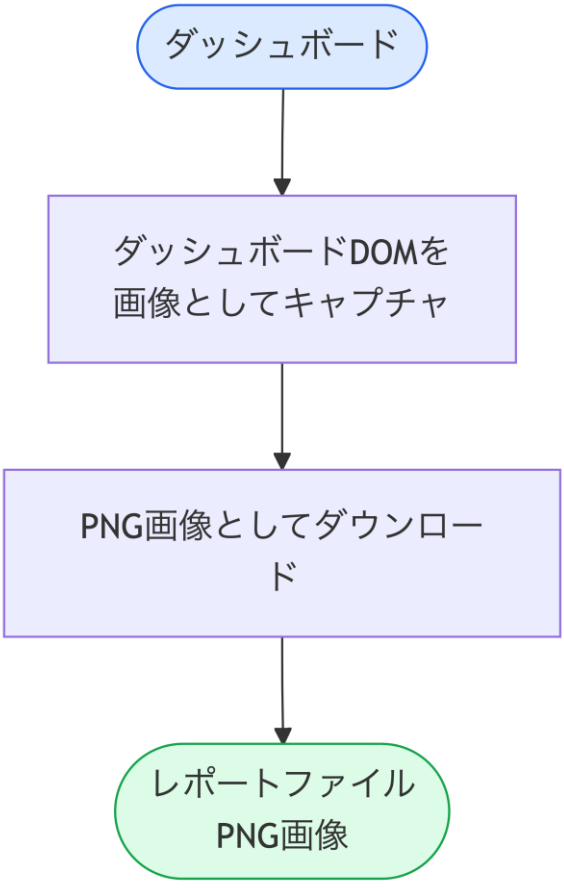


図 2.61　【FN113】のフローチャート

本システム機能の処理の詳細

- **ダッシュボードキャプチャ**
  - 処理内容：html-to-image ライブラリでダッシュボードの DOM を PNG 画像に変換し、ダウンロード用ファイルとして提供する。
  - 利用するライブラリ：【SL112】html-to-image

本システム機能の入出力データの仕様

- **入力**
  - ダッシュボード（ウィジェット配置・設定）
    - データの形式：JSON
- **出力**
  - レポートファイル
    - データの形式：PNG

【FN114】ダッシュボード共有＜新規開発＞

本システム機能の概要

職員が、ダッシュボードを URL で共有することができる（同一チーム内のユーザーには編集権限で、チーム外のユーザーには閲覧権限で URL を共有する）。

本機能では、共有 URL はアカウントを持つユーザーであれば、同一チームに所属していなくてもアクセスできる。共有されたダッシュボードは閲覧権限のみで表示され、ウィジェットの追加・編集・削除を行うことはできない。共有の解除も可能である。

本システム機能の入力・処理・出力のフローチャート

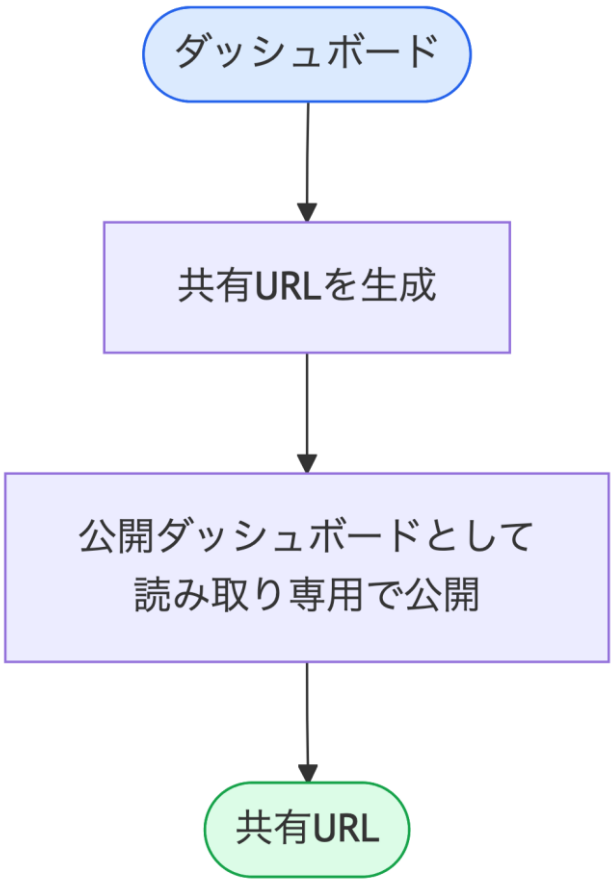


図 2.62　【FN114】のフローチャート

本システム機能の処理の詳細

- **共有 URL 生成**
  - ■ 処理内容：ダッシュボードの共有設定を有効にし、公開用 URL を生成する。
  - ■ 利用するインターフェース：【IF010】LINKS BI 汎用 REST API

- **公開ダッシュボード表示**
  - ■ 処理内容：共有 URL にアクセスしたユーザーの認証を確認し、ダッシュボードを読み取り専用モードで表示する。同一チームに所属していなくても、アカウントを持つユーザーであればアクセスできる。

【FN115】GIS 機能＜新規開発＞

本システム機能の概要

職員が、自然言語でのチャットを通して、バッファ処理や空間結合等の基本的な GIS 演算処理をノーコードで実行することができる。

本機能では、AI チャット（【FN109】）のツールシステムを通じて GIS 演算処理を実行し、結果を地図上に可視化する。ジオコーディング（住所→座標変換）、距離計算（2 点間距離）、空間フィルタリング（ある領域内のデータ抽出）、メッシュコードグリッド生成等の空間演算を自然言語で指示できる。DuckDB の spatial 拡張（「ST_Distance」・「ST_Intersects」・「ST_Buffer」 等の空間関数）をブラウザ内で実行し、結果をマップ可視化（【FN106】）のベクタータイルとして描画する。ルート探索機能では、Cloud SQL（links-veda-routing）上の pgRouting を活用する。

本システム機能の入力・処理・出力のフローチャート

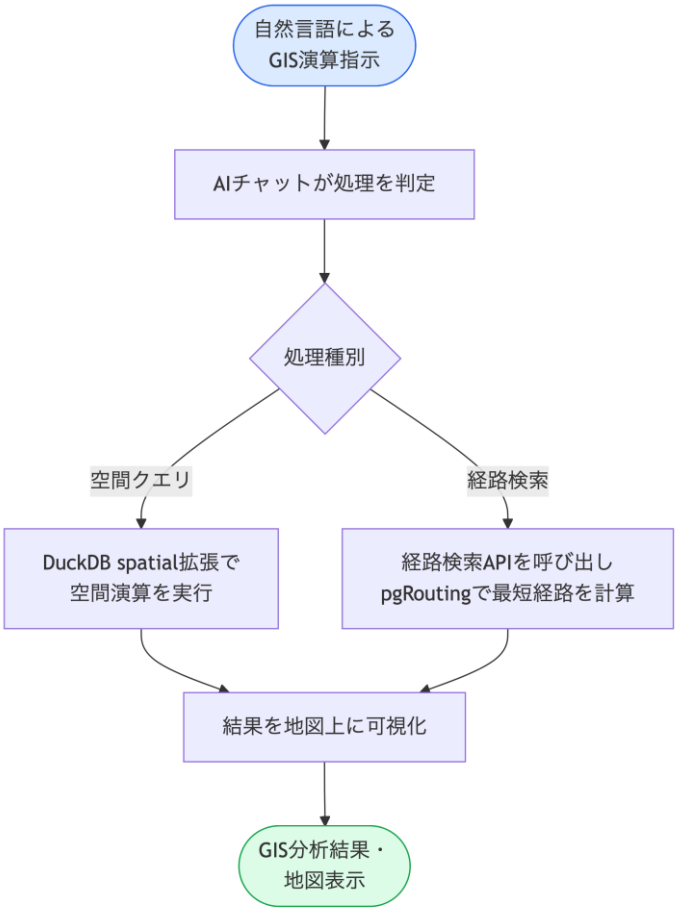


図 2.63 【FN115】のフローチャート

本システム機能の処理の詳細

- **空間クエリ実行**
  - 処理内容：AI チャットが 「duckdb_query」 ツール経由で DuckDB spatial SQL を生成・実行する。空間結合・バッファ分析・距離算出等の結果を新規テーブルとして作成し、マップ可視化に渡す。
  - 利用するライブラリ：【SL102】@duckdb/duckdb-wasm（spatial 拡張）

- **メッシュコード分析**
  - 処理内容：メッシュコードカラムを持つデータに対して、JIS X 0410 準拠のグリッドポリゴンを自動生成し、統計値を色分けして地図上に描画する。メッシュ次数の自動判定・集計・可視化を一連の処理として実行する。
  - 利用するライブラリ：【SL108】MapLibre GL JS（BI）

- **経路検索**
  - 処理内容：出発地・目的地を指定し、経路検索 API を呼び出して最適経路を取得する。経路検索は PostgreSQL 上の pgRouting（pgr_bdAstar、双方向 A* アルゴリズム）を使用し、OpenStreetMap ベースの道路ネットワークデータ上で最短経路を計算する。トラック単独モードとトラック＋船舶のマルチモーダル検索に対応する。検索結果には経路ジオメトリ・距離・所要時間・CO2 排出量が含まれる。
  - 利用するライブラリ：pgRouting（PostgreSQL 拡張）
  - 利用するデータインターフェース：経路検索 API（POST /api/search-route）、Cloud SQL（PostgreSQL / links-veda-routing）

【FN116】ファイル検索＜新規開発＞

本システム機能の概要

職員が、自然言語で AI とのチャットを通して検索条件を指定し、データ構築モジュールで構造化された PDF 等の元のアセットを検索することができる。

本機能では、ベクトル検索（【FN045】ドキュメントインデキシング・ベクトル検索）のインデキシング済みデータを活用し、ファイル検索 API を通じて結果を取得する。

本システム機能の入力・処理・出力のフローチャート

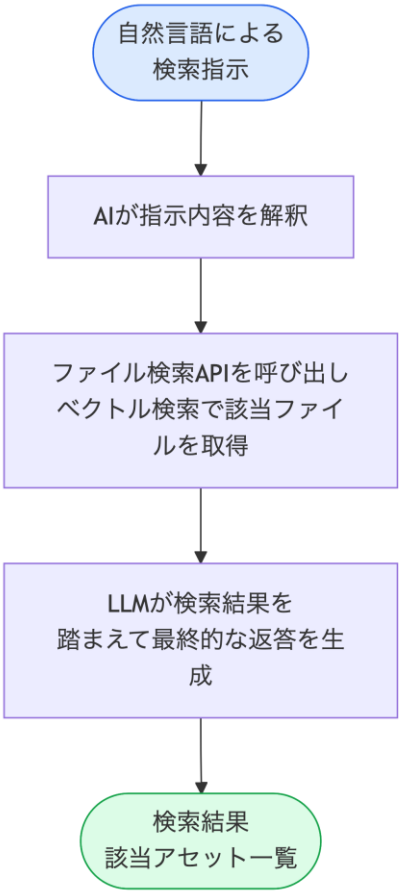


図 2.64　【FN116】のフローチャート

本システム機能の処理の詳細

- **ファイル検索**
  - 処理内容：ユーザーが自然言語で指定した検索条件を AI が解釈し、ファイル検索 API を呼び出してベクトル検索によるファイル検索を実行する。
  - 利用するアルゴリズム：【AL008】階層的ドキュメントインデキシング・RRF ハイブリッド検索
  - 利用するデータインターフェース：【IF017】pgvector ベクトルストアインターフェース

本システム機能の入出力データの仕様

- **入力**
  - 自然言語による検索クエリまたはサマリー指示
    - データの形式：テキスト
- **出力**
  - 検索結果（該当アセット一覧・メタデータ）またはサマリーテキスト
    - データの形式：JSON、Markdown テキスト
    - 利用するデータインターフェース：【IF017】pgvector ベクトルストアインターフェース

2.1.4　ソフトウェア・ライブラリ（SL）の詳細

本節では、システム開発に利用するソフトウェア・ライブラリの一覧と詳細を記載する。

表 2.3　ソフトウェア・ライブラリ一覧（データ構築モジュール）

ID	名称	バージョン	内容
SL001	Remix	2.15 以上 3.0 未満	React ベースのフルスタック Web フレームワーク。サーバーサイドレンダリング・ルーティング・API エンドポイントを統合し、データ構築モジュールの Web アプリケーション基盤として使用する。
SL002	React	18.3 以上 19.0 未満	UI コンポーネントライブラリ。仮想 DOM による効率的な画面更新を提供し、データ構築モジュールのフロントエンド全体で使用する。
SL003	Prisma	6.3 以上 7.0 未満	TypeScript 向け ORM。型安全なクエリ・スキーママイグレーションを提供し、PostgreSQL（Cloud SQL）への全データアクセスに使用する。

SL004	TypeScript	5.7 以上 6.0 未満	JavaScript に静的型付けを追加したプログラミング言語。型エラーのコンパイル時検出・IDE サポートにより、データ構築モジュール / LINKS BI 全体の型安全な実装言語として使用する。
SL005	Tailwind CSS	3.4 以上 4.0 未満	ユーティリティファーストの CSS フレームワーク。クラス名でスタイルを直接指定し、データ構築モジュール UI のスタイリング基盤として使用する。
SL006	Firebase（クライアント）	10.14 以上 11.0 未満	Google Firebase のクライアントサイド SDK。Cloud Identity Platform によるブラウザ上でのログイン・ID トークン取得に使用する。
SL007	firebase-admin	12.7 以上 13.0 未満	Firebase Admin SDK。サーバーサイドで Cloud Identity Platform の ID トークン検証・Cookie 発行・ユーザー管理に使用する。
SL008	@google-cloud/storage	7.15 以上 8.0 未満	Google Cloud Storage Node.js クライアントライブラリ。Bronze / Silver / Gold Layer のファイルアップロード・ダウンロード・署名付き URL 生成に使用する。

SL009	DuckDB（node-api）	1.4 以上 2.0 未満	DuckDB の Node.js ネイティブ API。サーバーサイドで高速 OLAP クエリを実行し、ファイルインポート・Silver / Gold Layer のデータ変換に使用する。
SL010	Zustand	5.0 以上 6.0 未満	軽量なグローバルステート管理ライブラリ。React の hooks API を活用したシンプルな store を提供し、データ構築モジュールフロントエンドの状態管理に使用する。
SL011	ulid	2.3 以上 3.0 未満	ULID（Universally Unique Lexicographically Sortable Identifier）生成ライブラリ。時刻ソート可能な ID を生成し、ワークフロー・ファイル・データセット等のリソース ID に使用する。

表 2.4　ソフトウェア・ライブラリ一覧（Cloud Run 処理サービス（processing/））

ID	名称	バージョン	内容
SL020	boto3	>=1.40	AWS SDK for Python。Bedrock（LLM）・S3 等の AWS サービスにアクセスし、Cloud Run 処理サービスから AWS Bedrock Claude を呼び出すために使用する。

SL021	azure-ai-documentintelligence	>=1.0	Azure Document Intelligence（旧 Form Recognizer）Python クライアントライブラリ。PDF・画像からのテキスト・レイアウト・テーブル抽出（OCR）に使用する。
SL022	redis（Python）	>=5.0	Python の Redis クライアントライブラリ。OCR 結果の MD5 ハッシュキーキャッシュに使用する（Redis / Memorystore 接続）。
SL023	PyMuPDF	>=1.24	MuPDF ベースの Python PDF ライブラリ。PDF ページの JPEG 変換・テキスト抽出を提供し、VLM モードの訂正線検知処理に使用する。
SL024	Pillow	>=11.3	Python Imaging Library（PIL）フォーク。画像の読み込み・リサイズ・形式変換等を提供し、PDF ページ画像のリサイズ・品質調整に使用する。
SL025	pydantic	>=2.10	Python のデータバリデーションライブラリ。型アノテーションベースで Cloud Run 処理サービスの API リクエスト・レスポンスのスキーマバリデーションに使用する。

SL026	tiktoken	>=0.11	OpenAI 製のトークンカウンターライブラリ。cl100k_base エンコーダーでテキストのトークン数を近似計算し、Bedrock Claude 呼び出し前のトークン数チェックに使用する。
SL027	langchain-aws / langchain-core	>=0.2	LangChain の AWS 統合パッケージ。スキーマ提案処理において LLM の出力（JSON）を構造化するための JSONOutputParser として使用する。
SL028	LiteLLM	>=1.81	複数の LLM プロバイダーを統一インターフェースで呼び出すゲートウェイライブラリ。DocumentIndexer で Titan Text Embeddings V2 および Claude Haiku 4.5 の呼び出しに使用する。
SL029	asyncpg	>=0.31	Python 向け高性能 PostgreSQL 非同期ドライバー。コネクションプールを内蔵し、DocumentIndexer のベクトルストアへの非同期アクセスに使用する。

SL030	pgvector（PostgreSQL 拡張）	>=0.4	PostgreSQL にベクトルデータ型と近似最近傍検索機能を追加する拡張。VECTOR 型・HNSW インデックス・コサイン距離演算子を提供し、DocumentIndexer のベクトルストアに使用する。
SL031	langchain-text-splitters	>=0.3	LangChain のテキスト分割ライブラリ。RecursiveCharacterTextSplitter で複数のセパレータを順に試し、DocumentIndexer の階層的チャンキングに使用する。
SL032	azure-cognitiveservices-vision-customvision	>=3.1	画像をアップロードしてラベル付けするだけで、画像分類や物体検出の AI モデルを簡単に作成・利用できる Azure のサービスである。
SL033	LibreOffice	>=25	Word・Excel・PowerPoint などの形式を PDF に変換できるオープンソースのオフィスソフト。
SL034	GhostScript	>=10.05	PDF ファイルの表示・変換・圧縮・処理を行うためのオープンソースツール。OCR 前処理で二値化として使用する。

表 2.5　ソフトウェア・ライブラリ一覧（LINKS BI）

ID	名称	バージョン	内容
SL101	React	19.2 以上 20.0 未満	UI コンポーネントライブラリ（最新版）。Concurrent Rendering に対応し、LINKS BI フロントエンド UI コンポーネントの構築に使用する。
SL102	@duckdb/duckdb-wasm	1.30 以上 2.0 未満	DuckDB の WebAssembly 版。ブラウザ内で SQL 分析クエリを実行し、OPFS による一時保持に対応する LINKS BI のデータ分析エンジンとして使用する。
SL103	vega / vega-lite	6.2 以上 7.0 未満	JavaScript データビジュアライゼーション宣言型文法ライブラリ。JSON 定義でグラフを生成し、LINKS BI のグラフ可視化ウィジェット（棒・折れ線・散布図・ヒストグラム等）に使用する。
SL104	react-grid-layout	2.2 以上 3.0 未満	React のドラッグ&ドロップグリッドレイアウトライブラリ。ウィジェット配置・サイズ変更・レスポンシブ対応を提供し、ダッシュボードのレイアウト管理に使用する。

SL105	@ai-sdk/react	3.0 以上 4.0 未満	Vercel AI SDK の React バインディング。useChat フックでストリーミング AI チャット UI を実装し、LINKS BI フロントエンドのチャット機能に使用する。
SL106	ai（Vercel AI SDK）	6.0 以上 7.0 未満	Vercel AI SDK 本体。SSE ストリーミング・Function calling・複数 LLM プロバイダー対応を提供し、LINKS BI フロントエンドの AI 処理に使用する。
SL107	@tanstack/react-router	1.141 以上 2.0 未満	TanStack 製のクライアントサイドルーティングライブラリ。型安全なルート定義・コード分割・データローディングを提供し、LINKS BI の画面遷移に使用する。
SL108	MapLibre GL JS	5.15 以上 6.0 未満	オープンソースのインタラクティブ Web マップライブラリ。メッシュコードグリッド描画等の高度な機能を備え、LINKS BI の地理空間データ可視化に使用する。
SL109	apache-arrow	21.1 以上 22.0 未満	Apache Arrow 形式データ処理ライブラリ。DuckDB-Wasm のクエリ結果を Arrow 形式で効率的に取り扱い、データの受け渡しに使用する。

SL110	ml-kmeans	7.0 以上 8.0 未満	JavaScript の K-Means クラスタリングライブラリ。AI チャットの perform_cluster_analysis ツールでブラウザ内クラスター分析に使用する。
SL111	jotai	2.16 以上 3.0 未満	アトミックな React ステート管理ライブラリ。プリミティブな atom を組み合わせて状態を管理し、LINKS BI フロントエンドのグローバルステート管理に使用する。
SL112	html-to-image	1.11 以上 2.0 未満	HTML/DOM 要素をキャンバスまたは画像に変換するライブラリ。ブラウザ内でダッシュボード全体をキャプチャし、PDF 形式でレポート出力する機能に使用する。
SL113	jstat	1.9 以上 2.0 未満	JavaScript の統計計算ライブラリ。回帰分析・t 検定・正規分布等の統計関数を提供し、AI チャットの線形回帰分析ツールに使用する。
SL114	Hono	4.10 以上 5.0 未満	エッジランタイム対応の軽量 TypeScript Web フレームワーク。OpenAPI 定義と型安全なルーティングを提供し、LINKS BI バックエンド API サーバーとして使用する。

SL115	@hono/zod-openapi	1.2 以上 2.0 未満	Hono フレームワークの OpenAPI / Zod インテグレーション。型安全な API ルート定義と OpenAPI スキーマ自動生成を提供し、LINKS BI バックエンド API の定義に使用する。
SL116	Kysely	0.28 以上 1.0 未満	TypeScript 向け型安全 SQL クエリビルダー。実行時に型検証せず SQL を直接構築し、LINKS BI バックエンドの PostgreSQL アクセスに使用する。
SL117	@ai-sdk/amazon-bedrock	4.0 以上 5.0 未満	Vercel AI SDK の Amazon Bedrock プロバイダー。Bedrock のストリーミング AI 推論を Vercel AI SDK 経由で利用し、AI チャット機能に使用する。
SL118	pino	10.1 以上 11.0 未満	高速な構造化ログライブラリ。JSON 形式でログを出力し、Google Cloud Logging での集約に対応する LINKS BI バックエンドのロギング基盤として使用する。

【SL001】Remix

- **概要：** フルスタック Web フレームワーク。ファイルベースルーティング・SSR・API サーバー機能を統合。
- **ベンダー/提供元：** Remix Software（現 Shopify）
- **バージョン：** ^2.15
- **ライセンス：** MIT
- **公式ドキュメント：** https://remix.run/docs
- **利用目的：** データ構築モジュールの Web アプリ全体（フロントエンド UI + API エンドポイント）を構築する基盤。
- **利用する機能：** 【FN002】〜【FN045】全機能の UI・API 実装

【SL002】React（データ構築モジュール）

- **概要：** Facebook 製の UI コンポーネントライブラリ。宣言的な UI 構築・仮想 DOM・コンポーネントベース開発を提供する。
- **ベンダー/提供元：** Meta（Facebook）
- **バージョン：** ^18.3
- **ライセンス：** MIT
- **公式ドキュメント：** https://react.dev/
- **利用目的：** データ構築モジュールフロントエンド UI コンポーネントの構築。
- **利用する機能：** 【FN002】〜【FN045】全機能の UI コンポーネント

【SL003】Prisma

- **概要：** TypeScript 向け ORM。型安全なクエリ・スキーママイグレーションを提供。
- **ベンダー/提供元：** Prisma Data Inc.
- **バージョン：** ^6.3
- **ライセンス：** Apache 2.0
- **公式ドキュメント：** https://www.prisma.io/docs
- **利用目的：** PostgreSQL（Cloud SQL）へのすべてのデータアクセス。
- **利用する機能：** 【FN002】〜【FN043】

【SL004】TypeScript

- **概要：** JavaScript に静的型付けを追加したプログラミング言語。型エラーのコンパイル時検出・IDE サポートを提供する。
- **ベンダー/提供元：** Microsoft
- **バージョン：** ^5.7
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://www.typescriptlang.org/docs/
- **利用目的：** データ構築モジュール / LINKS BI 全体の型安全な実装言語。
- **利用する機能：** 全機能の実装

【SL005】Tailwind CSS

- **概要：** ユーティリティファーストの CSS フレームワーク。クラス名でスタイルを直接指定する設計思想。
- **ベンダー/提供元：** Tailwind Labs
- **バージョン：** ^3.4
- **ライセンス：** MIT
- **公式ドキュメント：** https://tailwindcss.com/docs/
- **利用目的：** データ構築モジュール UI のスタイリング。
- **利用する機能：** 全 UI 画面

【SL006】Firebase（クライアント）

- **概要：** Google Firebase のクライアントサイドライブラリ。Cloud Identity Platform によるログイン・ID トークン取得を提供する。

- **ベンダー/提供元：** Google
- **バージョン：** ^10.14
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://firebase.google.com/docs/web/setup
- **利用目的：** ブラウザ上での Cloud Identity Platform ログイン・ID トークン取得。
- **利用する機能：** 【FN001】【IF011】

【SL007】firebase-admin

- **概要：** Firebase Admin SDK。サーバーサイドで Cloud Identity Platform のトークン検証・ユーザー管理を行う。
- **ベンダー/提供元：** Google
- **バージョン：** ^12.7
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://firebase.google.com/docs/admin/setup
- **利用目的：** サーバーサイドでの ID トークン検証・Cookie 発行・ユーザー管理。
- **利用する機能：** 【FN001】【IF011】

【SL008】@google-cloud/storage

- **概要：** Google Cloud Storage Node.js クライアントライブラリ。
- **ベンダー/提供元：** Google Cloud
- **バージョン：** ^7.15
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://cloud.google.com/storage/docs/reference/libraries
- **利用目的：** Bronze / Silver / Gold Layer のファイルアップロード・ダウンロード・署名付き URL 生成。
- **利用する機能：** 【FN005】【FN042】【IF001】

【SL009】DuckDB（node-api）

- **概要：** DuckDB の Node.js ネイティブ API（「@duckdb/node-api」）。サーバーサイドで高速 OLAP クエリを実行する。
- **ベンダー/提供元：** DuckDB Labs
- **バージョン：** ^1.4
- **ライセンス：** MIT
- **公式ドキュメント：** https://duckdb.org/docs/api/nodejs/overview
- **利用目的：** ファイルインポート（CSV / Excel / GeoJSON / Parquet）・Silver / Gold Layer のデータ変換・DuckLake スナップショット管理。
- **利用する機能：** 【FN005】【FN004】【FN003】【IF016】

【SL010】Zustand

- **概要：** 軽量なグローバルステート管理ライブラリ。React の hooks API を活用したシンプルな store を提供する。
- **ベンダー/提供元：** pmndrs
- **バージョン：** ^5.0
- **ライセンス：** MIT
- **公式ドキュメント：** https://zustand-demo.pmnd.rs/
- **利用目的：** データ構築モジュールフロントエンドのグローバルステート管理。
- **利用する機能：** 全機能の UI ステート

【SL011】ulid

- **概要：** ULID（Universally Unique Lexicographically Sortable Identifier）生成ライブラリ。時刻ソート可能な ID を生成する。
- **ベンダー/提供元：** ulid contributors
- **バージョン：** ^2.3
- **ライセンス：** MIT
- **公式ドキュメント：** https://github.com/ulid/spec
- **利用目的：** ワークフロー・ファイル・データセット等のリソース ID 生成。
- **利用する機能：** 全機能の ID 採番

【SL020】boto3

- **概要：** AWS SDK for Python。Bedrock（LLM）・S3 等の AWS サービスにアクセスする。
- **ベンダー/提供元：** Amazon Web Services
- **バージョン：** >=1.40
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://boto3.amazonaws.com/v1/documentation/api/latest/
- **利用目的：** Cloud Run 処理サービス（DataStructure）から AWS Bedrock Claude を呼び出す。
- **利用する機能：** 【FN008】【IF012】

【SL021】azure-ai-documentintelligence

- **概要：** Azure Document Intelligence（旧 Form Recognizer）Python クライアントライブラリ。
- **ベンダー/提供元：** Microsoft Azure
- **バージョン：** >=1.0
- **ライセンス：** MIT
- **公式ドキュメント：** https://learn.microsoft.com/azure/ai-services/document-intelligence/
- **利用目的：** PDF・画像からのテキスト・レイアウト・テーブル抽出（OCR）。
- **利用する機能：** 【FN008】【FN007】【IF013】

【SL022】redis（Python）

- **概要：** Python の Redis クライアントライブラリ。Redis / Memorystore 接続・キャッシュ操作を提供する。
- **ベンダー/提供元：** Redis Inc.
- **バージョン：** >=5.0
- **ライセンス：** MIT
- **公式ドキュメント：** https://redis-py.readthedocs.io/
- **利用目的：** OCR 結果の MD5 ハッシュキーキャッシュ（現在 production 環境では Redis 無効のため非稼働）。
- **利用する機能：** 【FN008】

【SL023】PyMuPDF

- **概要：** MuPDF をベースにした Python PDF ライブラリ。PDF ページの JPEG 変換・テキスト抽出・画像処理を提供する。
- **ベンダー/提供元：** Artifex Software
- **バージョン：** >=1.24
- **ライセンス：** AGPL-3.0 /商用ライセンス
- **ライセンス注記：** AGPL-3.0 は商用利用に制約があるため、本システムでは Artifex Software の商用ライセンスを取得して利用するか、AGPL-3.0 の条件（ソースコード公開義務等）を満たす形で利用する。NF008（商用利用可能なライセンスに限定）との整合を確保するため、利用形態を運用資料に明記すること。
- **公式ドキュメント：** https://pymupdf.readthedocs.io/
- **利用目的：** PDF → JPEG 変換（VLM モードの訂正線検知機能で使用）。

- **利用する機能：** 【FN007】

【SL024】Pillow

- **概要：** Python Imaging Library（PIL）フォーク。画像の読み込み・リサイズ・形式変換等を提供する。
- **ベンダー/提供元：** Pillow contributors
- **バージョン：** >=11.3
- **ライセンス：** HPND
- **公式ドキュメント：** https://pillow.readthedocs.io/
- **利用目的：** PDF ページ画像のリサイズ・品質調整（VLM モード処理）。
- **利用する機能：** 【FN008】【FN007】

【SL025】pydantic

- **概要：** Python のデータバリデーションライブラリ。型アノテーションベースでリクエスト・レスポンスを検証する。
- **ベンダー/提供元：** pydantic developers
- **バージョン：** >=2.10
- **ライセンス：** MIT
- **公式ドキュメント：** https://docs.pydantic.dev/
- **利用目的：** Cloud Run 処理サービスの API リクエスト・レスポンスのスキーマバリデーション。
- **利用する機能：** 【FN008】【FN007】

【SL026】tiktoken

- **概要：** OpenAI 製のトークンカウンターライブラリ。cl100k_base エンコーダーでテキストのトークン数を近似計算する。Claude（Anthropic）は独自のトークナイザーを使用しており、cl100k_base との計算結果は近似値である。
- **ベンダー/提供元：** OpenAI
- **バージョン：** >=0.11
- **ライセンス：** MIT
- **公式ドキュメント：** https://github.com/openai/tiktoken
- **利用目的：** Bedrock Claude 呼び出し前のトークン数チェック・コンテキスト制限超過時の自動トリミング。
- **利用する機能：** 【FN008】

【SL027】langchain-aws / langchain-core

- **概要：** LangChain の AWS 統合パッケージ。スキーマ提案処理において LLM の出力（JSON）を構造化するための 「JSONOutputParser」 として使用する。
- **ベンダー/提供元：** LangChain
- **バージョン：** >=0.2
- **ライセンス：** MIT
- **公式ドキュメント：** https://python.langchain.com/docs/integrations/platforms/aws/
- **利用目的：** スキーマ提案処理（SchemaSuggestion）における LLM 出力 JSON のパースのみに使用。LLM の呼び出し自体は boto3 が担当する。
- **利用する機能：** 【FN007】

【SL028】LiteLLM

- **概要：** 複数の LLM プロバイダー（AWS Bedrock、OpenAI、Azure 等）を統一インターフェースで呼び出すゲートウェイライブラリ。「aembedding()」 で埋め込み生成、「acompletion()」 でチャット補完を非同期実行する。
- **ベンダー/提供元：** BerriAI
- **バージョン：** >=1.81
- **ライセンス：** MIT
- **公式ドキュメント：** https://docs.litellm.ai/
- **利用目的：** DocumentIndexer で Titan Text Embeddings V2（埋め込み生成）および Claude Haiku 4.5（メタデータ抽出）の呼び出しに使用。
  「aws_region_name」 パラメータで東京リージョンを指定する。
- **利用する機能：** 【FN045】

【SL029】asyncpg

- **概要：** Python 向け高性能 PostgreSQL 非同期ドライバー。コネクションプール（「create_pool」）を内蔵し、「asyncio」 ベースの非同期 I/O で並列クエリを実行する。
- **ベンダー/提供元：** MagicStack
- **バージョン：** >=0.31
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://magicstack.github.io/asyncpg/

- **利用目的：** DocumentIndexer のベクトルストア（CO015）への非同期アクセス。テーブル初期化・ドキュメント UPSERT・チャンク一括挿入・ハイブリッド検索クエリの実行に使用する。
- **利用する機能：** 【FN045】

【SL030】pgvector（PostgreSQL 拡張）

- **概要：** PostgreSQL にベクトルデータ型と近似最近傍検索機能を追加する拡張。「VECTOR(N)」 型、HNSW / IVFFlat インデックス、コサイン距離演算子（「<=>」）を提供する。
- **ベンダー/提供元：** Andrew Kane / pgvector contributors
- **バージョン：** >=0.4
- **ライセンス：** PostgreSQL License
- **公式ドキュメント：** https://github.com/pgvector/pgvector
- **利用目的：** DocumentIndexer のベクトルストア（CO015）で 1024 次元ベクトルの格納・HNSW インデックスによるコサイン類似度検索に使用する。
- **利用する機能：** 【FN045】

【SL031】langchain-text-splitters

- **概要：** LangChain のテキスト分割ライブラリ。「RecursiveCharacterTextSplitter」 で複数のセパレータ（段落・改行・句点・スペース）を順に試し、指定サイズのチャンクに分割する。
- **ベンダー/提供元：** LangChain
- **バージョン：** >=0.3
- **ライセンス：** MIT
- **公式ドキュメント：** https://python.langchain.com/docs/how_to/recursive_text_splitter/

- **利用目的：** DocumentIndexer の HierarchicalChunker で、セクション内テキストを 1000 文字/ 200 文字オーバーラップのチャンクに分割する。未インストール時はフォールバックの自前分割ロジックで動作する。
- **利用する機能：** 【FN045】

【SL032】azure-cognitiveservices-vision-customvision

- **概要：** Azure Custom Vision は、画像を使った AI モデル（画像分類・物体検出）を簡単に作成できる Microsoft Azure の機械学習サービスで、プログラミングや AI の専門知識が少なくても、画像をアップロードしてラベル付けするだけでモデルを学習し、API として利用できる。
- **ベンダー/提供元：** Azure
- **バージョン：** >=3.1
- **ライセンス：** MIT
- **公式ドキュメント：** https://learn.microsoft.com/ja-jp/azure/ai-services/custom-vision-service/
- **利用目的：** 画像をアップロードしてラベル付けするだけで、画像分類や物体検出の AI モデルを簡単に作成・利用できる Azure のサービスである。
- **利用する機能：** 【FN008】

【SL033】LibreOffice

- **概要：** 文書作成、表計算、プレゼンテーションなどのオフィス文書を作成・編集し、Microsoft Office 形式を含む各種ファイルの表示・変換を行うことができるオープンソースのオフィスソフトウェア。
- **ベンダー/提供元：** The Document Foundation
- **バージョン：** >=25
- **ライセンス：** Mozilla Public License v2.0
- **公式ドキュメント：** https://documentation.libreoffice.org/ja/documentation-in-japanese/
- **利用目的：** Word・Excel・PowerPoint などの形式を PDF に変換できるオープンソースのオフィスソフト。
- **利用する機能：** 【FN008】

【SL034】GhostScript

- **概要：** PDF ファイルの表示・変換・圧縮・処理を行うためのオープンソースツール。
- **ベンダー/提供元：** Artifex Software, Inc.
- **バージョン：** >=10.05
- **ライセンス：** GNU Affero GPL or commercial license
- **公式ドキュメント：** https://www.ghostscript.com/releases/index.html
- **利用目的：** OCR 前処理で二値化として使用する。
- **利用する機能：** 【FN008】

【SL101】React（LINKS BI）

- **概要：** Facebook 製の UI コンポーネントライブラリ（最新版）。Concurrent Rendering に対応する。
- **ベンダー/提供元：** Meta（Facebook）
- **バージョン：** ^19.2
- **ライセンス：** MIT
- **公式ドキュメント：** https://react.dev/
- **利用目的：** LINKS BI フロントエンド UI コンポーネントの構築。
- **利用する機能：** 【FN101】〜【FN115】全 LINKS BI 機能の UI

【SL102】@duckdb/duckdb-wasm

- **概要：** DuckDB の WebAssembly 版。ブラウザ内で SQL 分析クエリを実行する。OPFS による一時保持対応。
- **ベンダー/提供元：** DuckDB Labs
- **バージョン：** ^1.30
- **ライセンス：** MIT
- **公式ドキュメント：** https://duckdb.org/docs/api/wasm/overview
- **利用目的：** LINKS BI のブラウザ内データ分析エンジン。
- **利用する機能：** 【FN102】【FN103】【FN104】【FN105】【FN106】【FN107】【FN109】

【SL103】vega / vega-lite

- **概要：** JavaScript データビジュアライゼーション宣言型文法ライブラリ。JSON 定義でグラフを生成する。
- **ベンダー/提供元：** UW Interactive Data Lab
- **バージョン：** vega ^6.2 / vega-lite ^6.4
- **ライセンス：** BSD-3-Clause
- **公式ドキュメント：** https://vega.github.io/vega-lite/
- **利用目的：** LINKS BI のグラフ可視化ウィジェット（棒・折れ線・散布図・ヒストグラム等）。
- **利用する機能：** 【FN105】

【SL104】react-grid-layout

- **概要：** React のドラッグ&ドロップグリッドレイアウトライブラリ。ウィジェット配置・サイズ変更・レスポンシブ対応を提供する。
- **ベンダー/提供元：** STRML
- **バージョン：** ^2.2
- **ライセンス：** MIT
- **公式ドキュメント：** https://github.com/react-grid-layout/react-grid-layout
- **利用目的：** ダッシュボードのグリッドレイアウト管理（ウィジェット配置・サイズ変更）。
- **利用する機能：** 【FN104】

【SL105】@ai-sdk/react

- **概要：** Vercel AI SDK の React バインディング。「useChat」 フックでストリーミング AI チャット UI を簡単に実装できる。
- **ベンダー/提供元：** Vercel
- **バージョン：** ^3.0
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://sdk.vercel.ai/docs/reference/ai-sdk-ui/use-chat
- **利用目的：** LINKS BI フロントエンドの AI チャット UI（ストリーミング表示・メッセージ管理）。
- **利用する機能：** 【FN109】

【SL106】ai（Vercel AI SDK）

- **概要：** Vercel AI SDK 本体。SSE ストリーミング・Function calling・複数 LLM プロバイダー対応を提供する。
- **ベンダー/提供元：** Vercel
- **バージョン：** ^6.0
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://sdk.vercel.ai/docs
- **利用目的：** LINKS BI フロントエンドでの Function calling 実行・ツール結果処理。
- **利用する機能：** 【FN109】

【SL107】@tanstack/react-router

- **概要：** TanStack 製のクライアントサイドルーティングライブラリ。型安全なルート定義・コード分割・データローディングを提供する。
- **ベンダー/提供元：** TanStack
- **バージョン：** ^1.141
- **ライセンス：** MIT
- **公式ドキュメント：**
  https://tanstack.com/router/latest/docs/framework/react/overview
- **利用目的：** LINKS BI フロントエンドのクライアントサイドルーティング（ダッシュボード/チャット画面間の遷移）。
- **利用する機能：** 【FN101】〜【FN115】全 LINKS BI 機能の画面遷移

【SL108】MapLibre GL JS（LINKS BI）

- **概要：** オープンソースのインタラクティブ Web マップライブラリ（最新版）。メッシュコードグリッド描画・プロパティ名ファジーマッチング等の高度な機能を使用する。
- **ベンダー/提供元：** MapLibre contributors
- **バージョン：** ^5.15
- **ライセンス：** BSD-3-Clause
- **公式ドキュメント：** https://maplibre.org/maplibre-gl-js/docs/
- **利用目的：** LINKS BI の地理空間データ可視化・メッシュコード（JIS X 0410）ポリゴングリッド描画。
- **利用する機能：** 【FN106】【FN115】

【SL109】apache-arrow

- **概要：** Apache Arrow 形式データ処理ライブラリ。DuckDB-Wasm のクエリ結果を Arrow 形式で効率的に取り扱う。
- **ベンダー/提供元：** Apache Software Foundation
- **バージョン：** ^21.1
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://arrow.apache.org/docs/js/
- **利用目的：** DuckDB-Wasm のクエリ結果受け取り・データの受け渡し。
- **利用する機能：** 【FN102】～【FN109】

【SL110】ml-kmeans

- **概要：** JavaScript の K-Means クラスタリングライブラリ。
- **ベンダー/提供元：** mljs
- **バージョン：** ^7.0
- **ライセンス：** MIT
- **公式ドキュメント：** https://mljs.github.io/kmeans/
- **利用目的：** AI チャットのクラスター分析ツール（「perform_cluster_analysis」）で使用する。
- **利用する機能：** 【FN109】【AL005】

【SL111】jotai

- **概要：** アトミックな React ステート管理ライブラリ。プリミティブな atom を組み合わせて状態を管理する。
- **ベンダー/提供元：** jotai contributors
- **バージョン：** ^2.16
- **ライセンス：** MIT
- **公式ドキュメント：** https://jotai.org/
- **利用目的：** LINKS BI フロントエンドのグローバルステート管理（チャートスペック・マップスペック等）。
- **利用する機能：** 【FN104】～【FN109】

【SL112】html-to-image

- **概要：** HTML/DOM 要素をキャンバスまたは画像に変換するライブラリ。ブラウザ内でダッシュボード全体をキャプチャする。
- **ベンダー/提供元：** bubkoo
- **バージョン：** ^1.11
- **ライセンス：** MIT
- **公式ドキュメント：** https://github.com/bubkoo/html-to-image
- **利用目的：** ダッシュボードを画像として取得し、PDF 形式でレポート出力する（【FN113】）。
- **利用する機能：** 【FN113】

【SL113】jstat

- **概要：** JavaScript の統計計算ライブラリ。回帰分析・t 検定・正規分布等の統計関数を提供する。
- **ベンダー/提供元：** jstat contributors
- **バージョン：** ^1.9
- **ライセンス：** MIT
- **公式ドキュメント：** https://jstat.github.io/jstat/
- **利用目的：** AI チャットの線形回帰分析ツール（「perform_regression」）における最小二乗法（OLS）の計算。
- **利用する機能：** 【FN109】【AL006】

【SL114】Hono

- **概要：** エッジランタイム対応の軽量 TypeScript Web フレームワーク。OpenAPI 定義と型安全なルーティングを提供。
- **ベンダー/提供元：** Hono プロジェクト
- **バージョン：** ^4.10
- **ライセンス：** MIT
- **公式ドキュメント：** https://hono.dev/
- **利用目的：** LINKS BI バックエンド API サーバー（認証・チャット・ダッシュボード・e-Stat 等）。
- **利用する機能：** 【FN101】〜【FN115】全 LINKS BI 機能

【SL115】@hono/zod-openapi

- **概要：** Hono フレームワークの OpenAPI / Zod インテグレーション。型安全な API ルート定義と OpenAPI スキーマ自動生成を提供する。
- **ベンダー/提供元：** Hono contributors
- **バージョン：** ^1.2
- **ライセンス：** MIT
- **公式ドキュメント：** https://hono.dev/examples/zod-openapi
- **利用目的：** LINKS BI バックエンド API の OpenAPI 定義・Zod バリデーション。
- **利用する機能：** 【FN101】〜【FN115】全 LINKS BI 機能

【SL116】Kysely

- **概要：** TypeScript 向け型安全 SQL クエリビルダー。実行時に型検証せず SQL を直接構築する。
- **ベンダー/提供元：** Kysely プロジェクト
- **バージョン：** ^0.28
- **ライセンス：** MIT
- **公式ドキュメント：** https://kysely.dev/
- **利用目的：** LINKS BI バックエンドの PostgreSQL アクセス（チャット・ダッシュボード管理）。
- **利用する機能：** 【FN101】【FN104】【FN107】【IF007】

【SL117】@ai-sdk/amazon-bedrock

- **概要：** Vercel AI SDK の Amazon Bedrock プロバイダー。Bedrock のストリーミング AI 推論を Vercel AI SDK 経由で利用する。
- **ベンダー/提供元：** Vercel
- **バージョン：** ^4.0
- **ライセンス：** Apache-2.0
- **公式ドキュメント：** https://sdk.vercel.ai/providers/ai-sdk-providers/amazon-bedrock
- **利用目的：** LINKS BI バックエンドから AWS Bedrock Claude を呼び出して AI チャットを実装する。
- **利用する機能：** 【FN109】【IF012】

【SL118】pino

- **概要：** 高速な構造化ログライブラリ。JSON 形式でログを出力し、Google Cloud Logging での集約に対応する。
- **ベンダー/提供元：** pino contributors
- **バージョン：** ^10.1
- **ライセンス：** MIT
- **公式ドキュメント：** https://getpino.io/
- **利用目的：** LINKS BI バックエンドの構造化ログ出力（リクエスト ID・チーム ID・処理時間等を含む）。
- **利用する機能：** 全 LINKS BI 機能のロギング

2.1.5　数理モデル・アルゴリズム（AL）の詳細

本節では、システムで利用する数理モデル・アルゴリズムの一覧と詳細を記載する。

表 2.6　数理モデル・アルゴリズム一覧

ID	名称	説明	アルゴリズムを利用した機能
AL009	LLM / VLM（大規模言語モデル/視覚言語モデル）	Transformer アーキテクチャに基づく深層ニューラルネットワーク。非構造データからの構造化出力生成、自然言語理解・生成の基盤技術。	【FN007】 【FN008】 【FN010】 【FN012】 【FN016】 【FN041】 【FN109】
AL001	OCR + LLM データ構造化アルゴリズム	Azure OCR でテキスト抽出後、LLM でスキーマに沿った JSON を抽出する。バッチ処理・マージ戦略を含む。	【FN008】

AL002	スキーマ自動提案アルゴリズム	PDF の OCR テキストを LLM に送信し、データ項目・型・説明を自動推定する。前バッチ結果を prior_items として累積する。	【FN007】
AL003	バウンディングボックス抽出アルゴリズム	LLM 抽出結果と OCR レイアウトの element_id を照合し、各フィールド値のバウンディングボックス座標を特定する。	【FN008】
AL004	不正検知アルゴリズム	抽出されたフィールドに 0.0〜1.0 の不正検知スコアを付与する（Bedrock Claude によるセカンドパス評価）。	【FN008】
AL005	K-Means クラスタリング	ml-kmeans ライブラリを使ったフロントエンド内クラスター分析。BI チャットの「perform_cluster_analysis」ツールから呼び出す。	【FN109】

AL006	線形回帰分析	フロントエンド内で実行する線形回帰・セグメント回帰分析。BI チャットの「perform_regression」・「perform_segmented_regression」 ツールから呼び出す。	【FN109】
AL007	プロンプトエンジニアリング	データ構造化精度向上のためのプロンプト構成最適化。Few-Shot Learning・Chain of Thought 等の手法を適用する。	【FN016】
AL008	階層的ドキュメントインデキシング・RRF ハイブリッド検索	OCR テキストを階層的チャンキング（document/section/chunk）し、Titan Text Embeddings V2 で 1024 次元ベクトルに変換後、RRF でセマンティック検索とキーワード検索を統合する。	【FN045】

AL010	訂正線の検知アルゴリズム	深層学習モデルおよび VLM を使って文書中の訂正線を検知し、訂正箇所を特定するアルゴリズム。	【FN012】
AL011	LLM モデルの自動選定アルゴリズム	文書の複雑さに応じて最適な LLM モデルを自動選定するアルゴリズム。	【FN014】
AL012	Office ファイルのデータ構造化アルゴリズム	Word・Excel・PowerPoint 等の Office ファイルを構造化データに変換するアルゴリズム。	【FN008】
AL013	エージェンティック AI（Function Calling）の実装	LLM の Function Calling を活用し、自律的にツールの選択・実行・結果評価を繰り返しながらタスクを達成するアルゴリズム。	【FN109】 【FN110】 【FN115】 【FN116】

【AL009】LLM / VLM（大規模言語モデル/視覚言語モデル）

本アルゴリズムの概要

- LLM（Large Language Model、大規模言語モデル）は、Transformer アーキテクチャに基づく深層ニューラルネットワークであり、本システムにおける AI 処理全般の基盤技術である。大量のテキストデータで事前学習されたモデルが、プロンプト（指示文）に従って自然言語の理解・生成を行い、非構造データから構造化された出力を生成する。
- VLM（Vision Language Model、視覚言語モデル）は、LLM の言語理解能力を画像・映像などの視覚情報に拡張したモデルである。テキストと画像の両方を入力として受け取り、画像内の文字・レイアウト・図表構造を理解した上で、構造化出力を生成できる。
- 本システムでは AWS Bedrock 経由で Anthropic Claude（Claude Sonnet 4.5）を主要 LLM / VLM として使用し、データ構造化・スキーマ提案・信頼度評価・メタ情報生成・AI チャット等の処理で活用する。

本アルゴリズムを利用した機能

- 【FN007】スキーマ自動提案
- 【FN008】データ構造化処理
- 【FN010】信頼度評価・不正生成検知
- 【FN012】訂正線検知
- 【FN016】LLM プロンプト最適化
- 【FN041】メタ情報自動生成
- 【FN109】AI チャット

本アルゴリズムの計算・処理の詳細

1. Transformer アーキテクチャの概要と構成

Transformer は、2017 年に Google が発表した"Attention Is All You Need"論文で提案されたニューラルネットワークアーキテクチャである。Self-Attention（自己注意）により、入力系列中の各トークン（単語・サブワード）が他のすべてのトークンとの関連度を動的に計算し、文脈に応じた表現を獲得する。

本システムで用いる LLM（Claude Sonnet / Haiku）および埋め込みモデル（Titan Text Embeddings V2）は、いずれも Transformer 系アーキテクチャの考え方を基盤としている（用途により Encoder のみ、Decoder のみ、Encoder-Decoder を採る）。

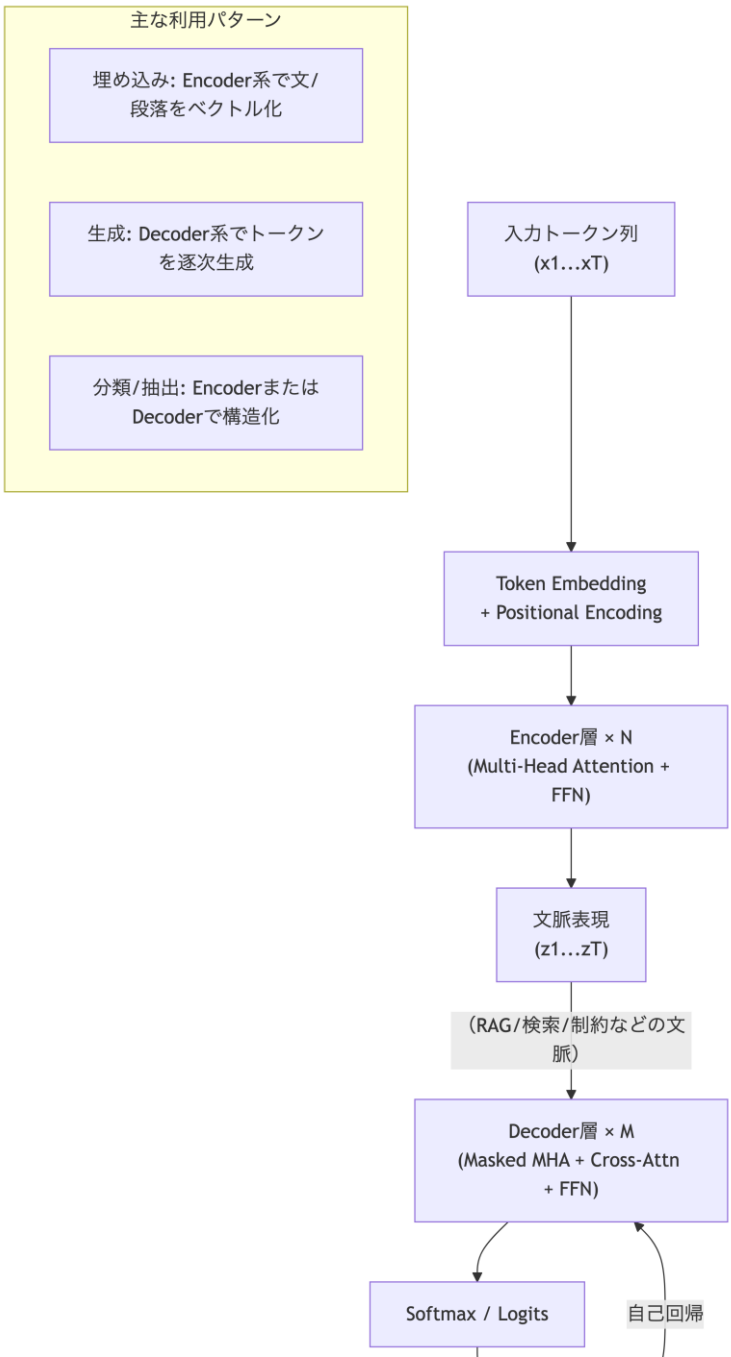

主な利用パターン
埋め込み: Encoder系で文/段落をベクトル化
生成: Decoder系でトークンを逐次生成
分類/抽出: EncoderまたはDecoderで構造化
入力トークン列 (x1…xT)
Token Embedding + Positional Encoding
Encoder層 × N (Multi-Head Attention + FFN)
文脈表現 (z1…zT)
（RAG/検索/制約などの文脈）
Decoder層 × M (Masked MHA + Cross-Attn + FFN)
Softmax / Logits
自己回帰

図 2.65 Transformer アーキテクチャ

表 2.7　本システムで使用するモデル一覧

モデル	**カテゴリ**	**主な用途（本システム）**	**強み**	**注意点**
Claude Sonnet 4.5	LLM（高性能）	複雑な推論・長文/複雑レイアウト文書の理解、難易度の高い抽出・整形、仕様生成、整合チェック。	推論精度と指示追従が強い。多段推論や制約があるタスクに向く。	相対的にコストとレイテンシが大きくなりやすい。大量バッチは分割・非同期との併用が重要。
Claude Haiku 4.5	LLM（軽量・高速）	軽量な抽出・整形、前処理（要約、分類、簡易スキーマ候補生成）、対話の即応。	高速・低コストでスループット確保に向く。簡易タスクの大量処理に適する。	複雑な推論や厳密な整合が必要な処理は Sonnet へフォールバックする運用が望ましい。

Titan Text Embeddings V2	埋め込み（ベクトル化）	RAG 検索用ベクトル DB 作成、類似検索（質問↔根拠文書、カラム/定義の類似度）	テキストを固定次元ベクトルへ変換し、近傍探索で高速に関連文を抽出できる。 幻覚抑制と根拠提示に寄与。	生成は行わない。検索品質は前処理（分割粒度、メタデータ付与）とインデックス設計に依存する。

2. Scaled Dot-Product Attention

入力トークン系列 $X = (x_1, x_2, \ldots, x_n)$ に対して、各トークンから Query・Key・Value の 3 つのベクトルを線形変換で生成し、Scaled Dot-Product Attention を計算する。

$$Attention(Q, K, V) = softmax\left(\frac{QK^{¥top}}{\sqrt{d_k}}\right)V$$

ここで d_k は Key ベクトルの次元数であり、スケーリングで勾配の安定化を図る。Decoder の自己注意（Masked Self-Attention）やパディング除外等では、マスク M を加算して計算する。

$$Attention(Q, K, V) = softmax\left(\frac{QK^{¥top}}{\sqrt{d_k}} + M\right)V$$

3. Multi-Head Attention

h 個のヘッドに分割し、各ヘッドで独立に注意を計算する。

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$
$$MHA(Q, K, V) = Concat(head_1, \ldots, head_h)W^O$$

通常 $d_k = d_v = d_{model}/h$ とし、異なる投影により、多様な関係性（局所・長距離、列見出しと値の対応など）を同時に捉える。

4.構造化出力の生成

- 本システムでは、LLM に対してスキーマ定義（フィールド名・データ型・説明）をプロンプトで指示し、非構造データ（OCR テキスト・画像）から JSON 形式の構造化出力を生成させる。
- 推論パラメータは出力のばらつきを抑える目的で 「temperature=0」 を基本とする。ただし AWS Bedrock では 「temperature=0」 でも完全な決定論性は保証さ

れないため、再現性が求められる場面ではバリデーションや再実行戦略を併用する。

5. VLM による視覚理解

- VLM モードでは、PDF ページを画像（JPEG 等）に変換して Claude に送信する。Claude は画像内のテキスト・表・図・レイアウトを直接視覚的に理解し、OCR を介さずに構造化データを抽出する。

【AL001】OCR + LLM データ構造化アルゴリズム

本アルゴリズムの概要

- PDF・画像から OCR テキストを取得し、LLM（AWS Bedrock Claude）によって事前定義スキーマに従った構造化 JSON を抽出するアルゴリズムである。
- Precision 優先の方針を採用し、曖昧なフィールドには null を返す設計とすることで、ハルシネーション（不正生成）のリスクを低減する。
- 大規模文書に対応するため、ページをバッチ分割して処理し、バッチ結果をマージする戦略を備える。

本アルゴリズムを利用した機能

- 【FN008】データ構造化処理

本アルゴリズムの計算・処理の詳細

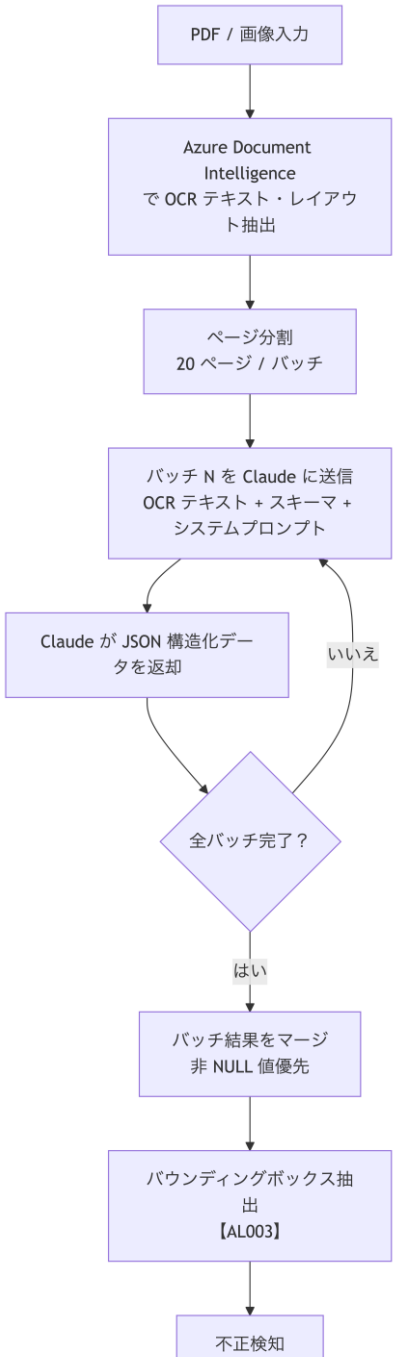

PDF / 画像入力
Azure Document Intelligence で OCR テキスト・レイアウト抽出
ページ分割 20 ページ / バッチ
バッチ N を Claude に送信 OCR テキスト + スキーマ + システムプロンプト
Claude が JSON 構造化データを返却
いいえ
全バッチ完了？
はい
バッチ結果をマージ 非 NULL 値優先
バウンディングボックス抽出 【AL003】
不正検知

図 2.66 OCR + LLM データ構造化アルゴリズムのフローチャート

1. **OCR テキスト抽出:**
   Azure Document Intelligence（「prebuilt-layout」 モデル）で PDF のテキスト・レイアウト・テーブル構造を抽出する。抽出結果は MD5 ハッシュでキャッシュし、同一ファイルの再処理を回避する。
2. **ページ分割:**
   抽出テキストを 「---PAGE BREAK---」 で分割し、20 ページ/バッチで LLM に送信する。これにより LLM のコンテキストウィンドウ制限を管理する。
3. **LLM 推論:**
   各バッチで Claude に OCR テキスト・スキーマ定義・システムプロンプトを送信し、スキーマに準拠した JSON を生成させる。
4. **バッチ結果マージ:**
   「_merge_batch_results()」 により、複数バッチの結果を統合する。同一フィールドに複数バッチから値が得られた場合は非 NULL 値を優先し、配列型フィールドは結合する。
5. **後処理:**
   バウンディングボックス抽出（【AL003】）で各フィールドの文書上の座標を特定し、不正検知（【AL004】）でスコアを付与する。

【AL002】スキーマ自動提案アルゴリズム

本アルゴリズムの概要

- PDF の OCR テキストを LLM に送信し、データ項目（フィールド名・型・説明）を自動推定して提案するアルゴリズムである。
- 前バッチの推定結果を次バッチの入力に含めることで、文書全体を通じたスキーマの一貫性を確保する累積マージ方式を採用する。

本アルゴリズムを利用した機能

- 【FN007】スキーマ自動提案

本アルゴリズムの計算・処理の詳細

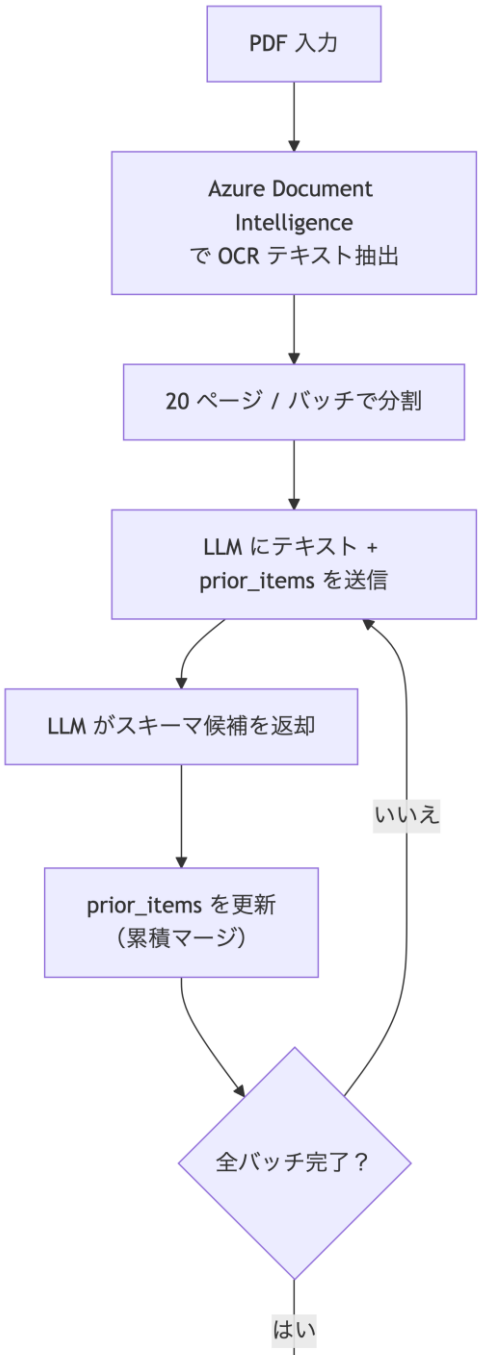

PDF 入力
Azure Document Intelligence で OCR テキスト抽出
20 ページ / バッチで分割
LLM にテキスト + prior_items を送信
LLM がスキーマ候補を返却
prior_items を更新（累積マージ）
全バッチ完了？
いいえ
はい

図 2.67 スキーマ自動提案アルゴリズムのフローチャート

1. **OCR 前処理**:
   OCR でページのテキストを抽出し、ページテキストマップに変換する。
2. **バッチ処理**:
   20 ページ/バッチで処理する。
3. **LLM 推論**:
   LLM に「このページからデータ構造を推定してください」というプロンプトとともにテキスト/画像を送信する。
4. **累積マージ**:
   前バッチの結果（「prior_items」）を次バッチの入力に含め、文書全体のスキーマを累積的に構築する。
5. **スキーマ返却**:
   最終的な 「SchemaItem[]」（key・type・description・fields）を同期的に返す。

【AL003】バウンディングボックス抽出アルゴリズム

本アルゴリズムの概要

- LLM 抽出結果と OCR レイアウトデータの element_id を照合し、各フィールド値が文書のどの位置（バウンディングボックス座標）に存在するかを特定するアルゴリズムである。
- バウンディングボックス情報を付与することで、UI 上で抽出結果の根拠箇所をハイライト表示し、ユーザーによる確認・修正を支援する。

本アルゴリズムを利用した機能

- 【FN011】バウンディングボックス抽出・構造化結果確認

本アルゴリズムの計算・処理の詳細

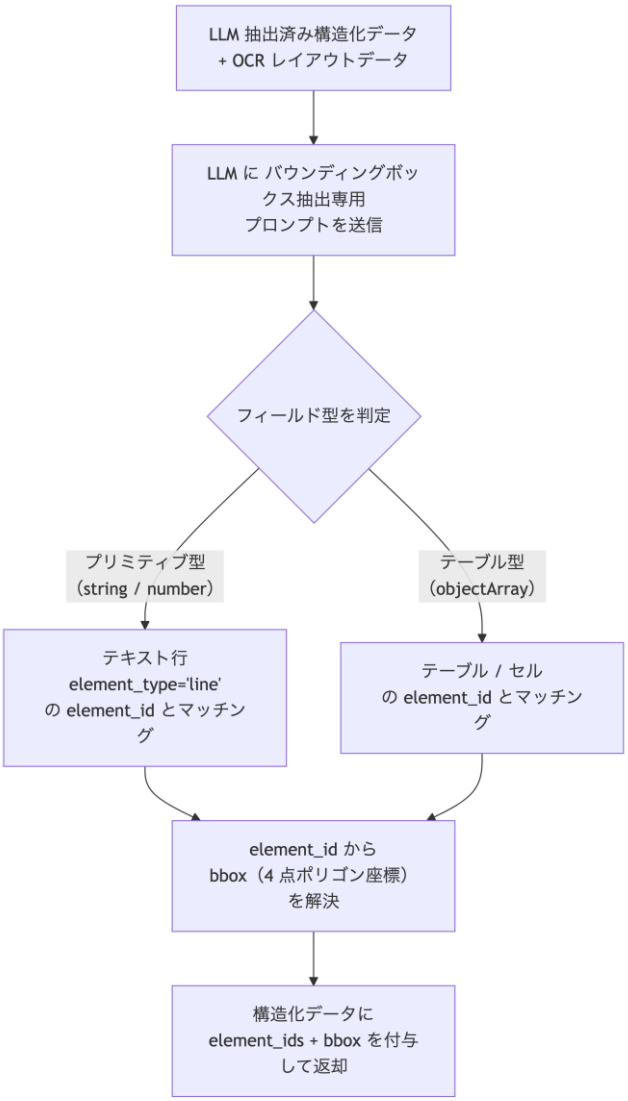

LLM 抽出済み構造化データ
+ OCR レイアウトデータ
LLM に バウンディングボックス抽出専用
プロンプトを送信
フィールド型を判定
プリミティブ型
（string / number）
テーブル型
（objectArray）
テキスト行
element_type='line'
の element_id とマッチング
テーブル / セル
の element_id とマッチング
element_id から
bbox（4 点ポリゴン座標）
を解決
構造化データに
element_ids + bbox を付与
して返却

図 2.68 バウンディングボックス抽出アルゴリズムのフローチャート

1. LLM 抽出済みの構造化データ（値のみ）と、OCR が付与した「element_id」・「bbox」・テキストを含むレイアウトデータを入力として受け取る。
2. LLM に「この値はどの element_id に対応するか」を問い合わせる（バウンディングボックス抽出専用プロンプト）。
3. プリミティブ型（string / number）はテキスト行（「element_type = "line"」）の element_id とマッチングする。
4. テーブル型（objectArray）はテーブル element_id または特定セルの element_id とマッチングする。
5. element_id から実際の 「bbox」（4 点ポリゴン座標）を解決して返す。

【AL004】不正検知アルゴリズム

本アルゴリズムの概要

- LLM で抽出した各フィールドに対して、0.0〜1.0 の不正検知スコアを付与する二段階評価アルゴリズム（セカンドパス評価）である。
- 一次抽出とは別の LLM 呼び出しで評価を行うことで、抽出結果の客観的な品質指標を提供し、ハルシネーション検知に活用する。

本アルゴリズムを利用した機能

- 【FN010】不正生成検知

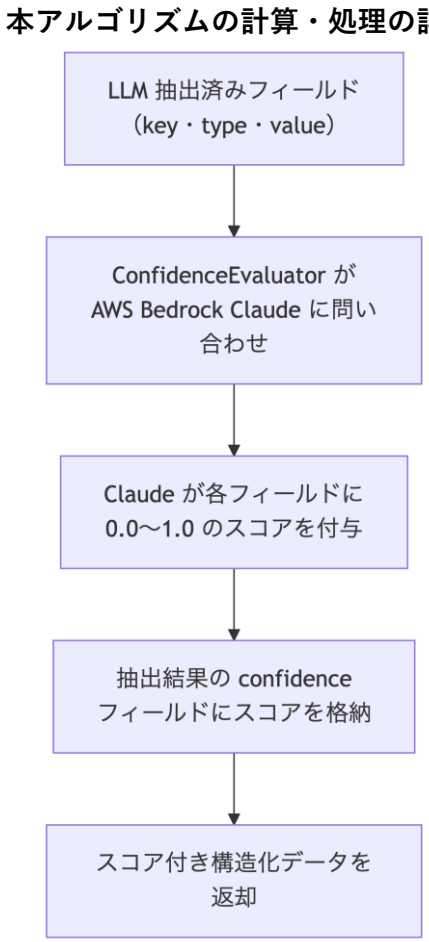


図 2.69 不正検知アルゴリズムのフローチャート

1. LLM 抽出結果の各フィールド（key・value）を入力として受け取る。

2. 「ConfidenceEvaluator」 が LLM に別途呼び出しを行い、「この値は正確に抽出されているか？」を問い合わせる。
3. LLM が各フィールドに 0.0〜1.0 のスコアを付与する。スコアが低いフィールドはハルシネーションの可能性があることを示す。
4. 抽出結果の 「confidence」 フィールドにスコアを格納して返す。

【AL005】K-Means クラスタリング

本アルゴリズムの概要

- 数値データを k 個のクラスターに分類する教師なし学習アルゴリズムである。
- LINKS BI フロントエンドで 「ml-kmeans」 ライブラリを使用してブラウザ内で実行し、AI チャットの分析結果として可視化する。

本アルゴリズムを利用した機能

- 【FN109】AI チャット

本アルゴリズムの計算・処理の詳細

1. AI チャットの 「perform_cluster_analysis」 ツールが DuckDB-Wasm クエリで数値データを取得する。

2. 「ml-kmeans」 ライブラリにデータと k（クラスター数）を渡す。

3. K-Means アルゴリズムの反復処理を実行する。

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} ¥|x - \mu_i¥|^2$$

ここで C_i は第 i クラスター、μ_i は第 i クラスターの重心、J は最小化すべき目的関数（クラスター内二乗和誤差）である。

4. 各データポイントに最近傍の重心に基づくクラスター番号を付与する。

5. 結果を Vega-Lite チャート（散布図+クラスター色分け）で可視化する。

【AL006】線形回帰分析

本アルゴリズムの概要

- 変数間の線形関係をモデル化し、予測・影響分析を行うアルゴリズムである。
- LINKS BI フロントエンドで JavaScript 実装（「jstat」 ライブラリ）を使用してブラウザ内で実行する。通常の線形回帰に加えて、区分線形回帰（セグメント回帰）にも対応する。

本アルゴリズムを利用した機能

- 【FN109】AI チャット

本アルゴリズムの計算・処理の詳細

1. 「select_predictors」 ツールでユーザーが説明変数と被説明変数を選択する。

2. 「perform_regression」 ツールで DuckDB-Wasm から数値データを取得し、最小二乗法（OLS: Ordinary Least Squares）で回帰係数を計算する。

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

ここで X は説明変数の行列、y は被説明変数のベクトル、$\hat{\beta}$ は推定回帰係数である。

3. 決定係数 R^2 および各係数の p 値を算出する。

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

4. 「perform_segmented_regression」 ツールでデータを複数セグメントに分割して区分線形回帰を実行する。

5. 結果を Vega-Lite チャート（散布図+回帰直線）で可視化する。

【AL007】プロンプトエンジニアリング

本アルゴリズムの概要

- データ構造化処理の精度向上を目的としたプロンプト構成最適化手法である。
- Few-Shot Learning（少数の入出力例示）、Chain of Thought（段階的推論の促進）等のプロンプトエンジニアリング技法を適用し、LLM の構造化出力品質を向上させる。

本アルゴリズムを利用した機能

- 【FN016】LLM プロンプト最適化

本アルゴリズムの計算・処理の詳細

1. **Few-Shot Learning**:
   ユーザーが定義したスキーマに対して、期待される入出力ペアの例をプロンプト内に含めることで、LLM が出力形式・抽出精度を学習する。
2. **Chain of Thought（CoT）**:
   複雑な抽出タスクにおいて、LLM に段階的な推論プロセスを促すプロンプト構造を適用する。中間推論ステップを明示的に出力させることで、最終的な抽出精度を向上させる。
3. **プロンプトテンプレート管理**:
   構造化処理・スキーマ提案・信頼度評価等のタスクごとに最適化されたプロンプトテンプレートを管理する。テンプレートにはシステムプロンプト・タスク指示・出力フォーマット指定・Few-Shot 例が含まれる。

【AL008】階層的ドキュメントインデキシング・RRF ハイブリッド検索

本アルゴリズムの概要

- OCR 処理済みテキストを階層的にチャンキングし、Titan Text Embeddings V2 で 1024 次元ベクトルに変換後、PostgreSQL + pgvector に格納するインデキシングアルゴリズムである。
- 検索時にはセマンティック検索（ベクトル近似最近傍）とキーワード検索（全文検索）を Reciprocal Rank Fusion（RRF）で統合し、高精度な文書検索を実現する。

本アルゴリズムを利用した機能

- 【FN045】ドキュメントインデキシング・ベクトル検索

本アルゴリズムの計算・処理の詳細

インデキシングフェーズ：

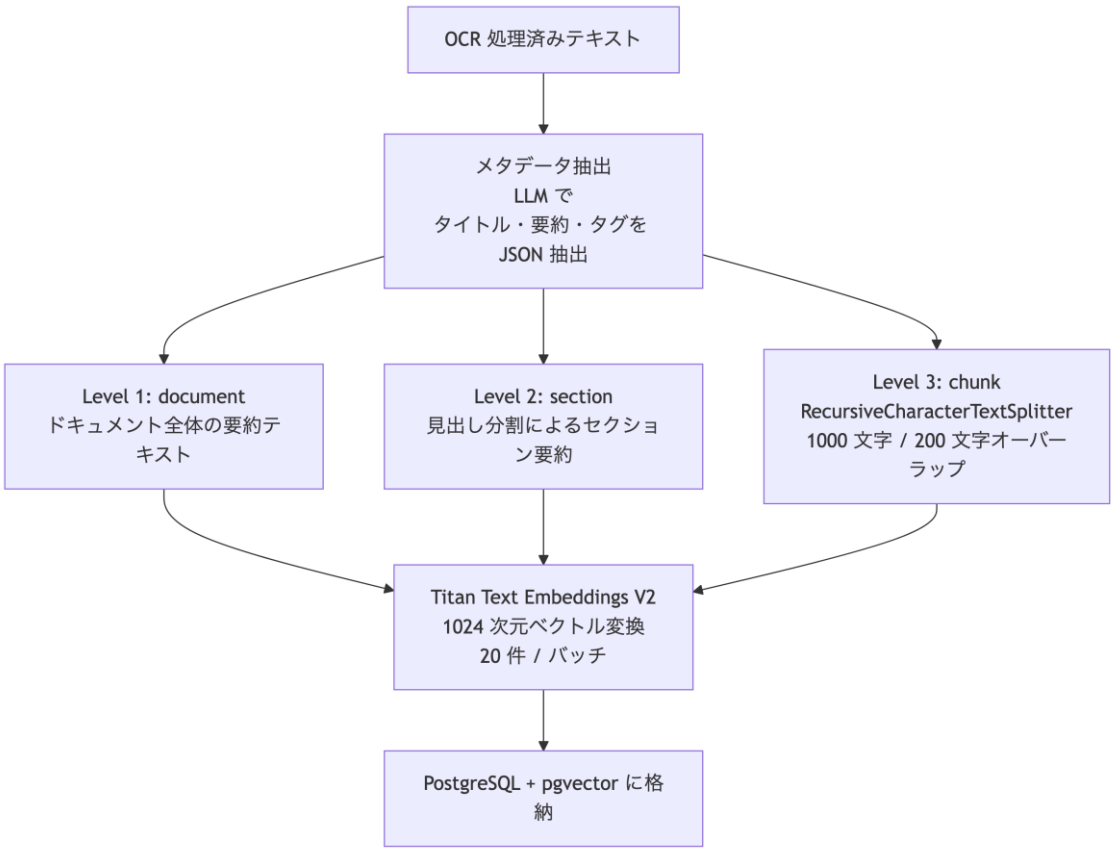


図 2.70 インデキシングフェーズのフローチャート

1. **メタデータ抽出**:
   OCR テキスト（先頭 8000 文字）を LLM に送信し、タイトル・要約・タグ・セクション要約を JSON 形式で抽出する。
2. **階層的チャンキング**:
   テキストを 3 レベルの階層に分割する。
   - Level 1（document）:ドキュメント全体の要約テキスト。
   - Level 2（section）: Markdown 見出し・日本語見出し（第 N 章・N.等）で分割されたセクションごとの要約テキスト。
   - Level 3（chunk）:各セクションを RecursiveCharacterTextSplitter（1000 文字/ 200 文字オーバーラップ）で分割したパラグラフチャンク。
3. **ベクトル埋め込み**:
   全チャンクを Titan Text Embeddings V2（「amazon.titan-embed-text-v2:0」）で 1024 次元ベクトルに変換する。20 件/バッチで処理し、指数バックオフリトライ（最大 5 回）を適用する。
4. **永続化**:
   ドキュメントメタデータ・セクション情報・チャンク（テキスト＋ベクトル）を PostgreSQL + pgvector に格納する。

検索フェーズ（RRF ハイブリッド検索）：

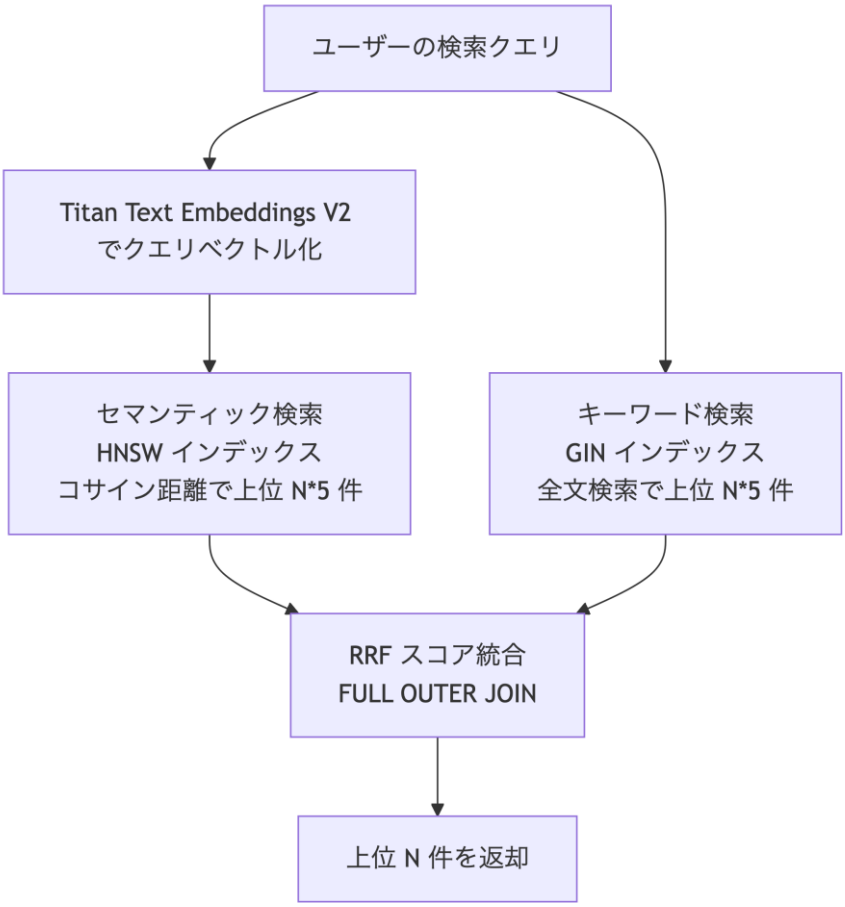


図 2.71 RRF ハイブリッド検索フェーズのフローチャート

1. **クエリベクトル化**:
   ユーザーの質問文を同一の Titan Text Embeddings V2 モデルでベクトルに変換する。
2. **セマンティック検索**:
   HNSW インデックスを使い 「embedding <=> query_vector」（コサイン距離）で上位 $N \times 5$ 件を取得し、ランクを付与する。
3. **キーワード検索**:
   GIN インデックスを使い 「to_tsvector('simple', content) @@ plainto_tsquery('simple', query_text)」 で上位 $N \times 5$ 件を取得し、「ts_rank_cd」 でランクを付与する。
4. **RRF 統合**:
   両方の検索結果を FULL OUTER JOIN し、以下の RRF スコアで上位 N 件を返却する。

$$RRF(d) = \sum_{r \in R} \frac{1}{k + rank_r(d)}$$

▲ R:セマンティック検索とキーワード検索の 2 つの結果セット。

▲ $rank_r(d)$:検索手法 r におけるドキュメント d のランク（結果に含まれない場合は $N \times 5 + 1$）。

▲ $k = 60$（RRF 定数）

【AL010】訂正線の検知アルゴリズム

本アルゴリズムの概要

深層学習モデルを使用して文書中の訂正線（一重取り消し線）や二重線（二重取り消し線）を検知し、VLM（Vision-Language Model）により訂正箇所を特定・修正するアルゴリズムである。
行政文書では手書きや印刷による訂正線で値が修正されるケースがあり、訂正前後の値を正確に識別・抽出することがデータ構造化の品質に直結する。

本アルゴリズムを利用した機能

【FN012】訂正線検知

本アルゴリズムの計算・処理の詳細

1. **ページ単位の訂正線有無判定**:データ構造化処理の過程で、各ページの画像を深層学習推論エンジンに送信し、ページ内に訂正線による訂正箇所が存在するかを判定する。
2. **訂正箇所の特定**:訂正線が検知されたページについて、VLM が訂正箇所の位置と訂正前の値（訂正された値）および訂正後の値（新しい値）を抽出する。
3. **構造化データへの反映**:抽出結果の該当フィールドに 「is_edited」 フラグを付与し、訂正前後の値をメタデータとして保持する。

VLM による訂正線検知の詳細ロジック

1.入力データ

訂正線が検知されたページの画像データを入力として VLM に送信する。
対象となるデータは、文書画像の視覚情報を含む画像データである。

2.訂正線の検出および種類判定

VLM は画像内の線情報を解析し、以下の基準に基づき訂正線を判定する。

- テキスト上を横断する直線または複数の平行線であること。
- 一重線（single strike-through）または二重線（double strike-through）として認識できる形状であること。
- テキスト内容と重なっている位置関係であること。

これらの特徴を基に、訂正線の種類および該当テキスト領域を特定する。

3.訂正前後の値の抽出

訂正線が検出されたテキスト領域について、VLM に対して以下のような観点のプロンプトを用いて情報抽出を行う。

- 訂正線によって取り消されている値（訂正前の値）。
- 近傍に記載されている修正後の値（訂正後の値）。
- 該当するフィールドまたはデータ項目。

VLM は視覚情報と文脈情報を組み合わせて解析し、訂正前後の値を抽出する。

4.出力データ

検出結果として以下の情報を出力する。

- 訂正前の値
- 訂正後の値

これらの情報は構造化データ処理に渡され、対象フィールドに反映される。

5.精度評価

訂正線検知および訂正前後値抽出の精度は、検証用データセットを用いて評価する。

評価指標として以下を使用する。

- 訂正線検出の正解率
- 訂正前後値抽出の一致率
- 文書全体に対する訂正検出の再現率

これらの指標によりアルゴリズムの性能を定量的に評価する。

【AL011】LLM モデルの自動選定アルゴリズム

本アルゴリズムの概要

- 文書の複雑さ・処理内容に応じて最適な LLM モデルを自動選定するアルゴリズムである。
- 上位モデルと下位モデルを文書特性に応じて使い分け、処理コストと精度のバランスを最適化する。

本アルゴリズムを利用した機能

- 【FN014】LLM モデル自動選定

本アルゴリズムの計算・処理の詳細

本アルゴリズムは、VD025（LLM モデル自動選定機能）の方針に基づき、入力文書の特性に応じて **高性能モデル（Sonnet）** と **低コスト高速モデル（Haiku）** を自動で振り分けるための判定処理である。
判定に用いる軸は次の 2 点とする。

- **一致率（Similarity）**：同一条件で実行した 2 モデルの構造化結果の一致度。
- **丸囲み（図形）有無**：入力文書に丸囲みが含まれるかどうか。

1.前提情報（判定入力）の取得

- ページ数・テキスト量などの入力サイズ情報。
- OCR 等の前処理結果（比較に用いる入力テキスト）
- 丸囲み（図形）検知結果（有／無）
- 出力スキーマ（フィールド数、データ型、ネスト構造の有無）

2.モデル選定ロジック

- **丸囲み（図形）が検知された場合**
  - ■ 丸囲みを含む資料ではモデル間で精度差が出やすい前提のため、比較判定を行わず **Sonnet を採用**する。
- **丸囲み（図形）が検知されない場合**
  - ■ Sonnet と Haiku を同一条件（同一 OCR テキスト、同一スキーマ、同一プロンプト、同一推論パラメータ）で実行し、構造化出力の **一致率（Similarity）**を算出する。
  - ■ 一致率を所定のしきい値と比較し、採用モデルを決定する。
    - ▲ 一致率がしきい値以上の場合：**Haiku を採用**する。
    - ▲ 一致率がしきい値未満の場合：**Sonnet を採用**する。

3.選定結果の適用

- 決定したモデル種別（モデル ID）を構造化処理パイプラインへ引き渡し、以降の LLM 呼び出しで使用する。

【AL012】Office ファイルのデータ構造化アルゴリズム

本アルゴリズムの概要

- Word・Excel・PowerPoint 等の Office ファイルを構造化データに変換するアルゴリズムである。
- ファイル形式に応じて最適な変換パイプラインを選択し、最終的に LLM による構造化処理（【AL001】）に接続する。

本アルゴリズムを利用した機能

● 【FN008】データ構造化処理

本アルゴリズムの計算・処理の詳細

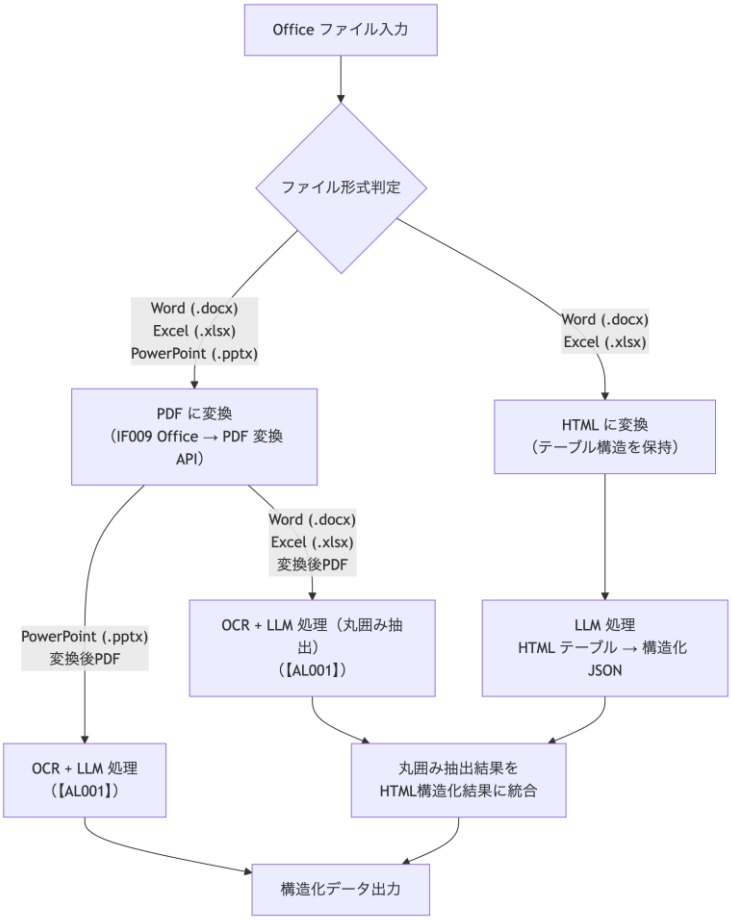


図 2.72 Office ファイルのデータ構造化フローチャート

1. **ファイル形式判定**:
   入力ファイルの形式（Word / Excel / PowerPoint）を判定する。
2. **PowerPoint の処理**:
   PDF に変換した上で、通常の OCR + LLM パイプライン（【AL001】）で処理する。PDF 変換には IF009（Office → PDF 変換 API）を使用する。
3. **Word / Excel の処理**:
   テーブル構造を保持したまま HTML に変換し、HTML テーブルを LLM に直接入力して構造化 JSON を抽出する。HTML 上では丸囲み抽出できなく、PowerPoint と同様に PDF に変換し、丸囲み抽出できるように構造化させ、最後に HTML で構造化 JSON 結果に統合し、最終結果とする。

【AL013】エージェンティック AI（Function Calling）の実装

本アルゴリズムの概要

- LLM の Function Calling（関数呼び出し）を活用し、AI が自律的にツールの選択・実行・結果評価を繰り返しながら複雑なタスクを達成するアルゴリズムである。
- LINKS BI の AI チャット機能の中核技術であり、自然言語による指示からデータ分析・可視化・外部 API 連携まで、一連の処理を AI が自律的に遂行する。
- スキルシステムによる段階的開示（Progressive Disclosure）を組み合わせ、ドメイン固有の知識を効率的にコンテキストウィンドウに読み込む設計を採用する。

本アルゴリズムを利用した機能

- 【FN109】AI チャット
- 【FN110】国会答弁機能
- 【FN115】GIS 機能
- 【FN116】ファイル検索

本アルゴリズムの計算・処理の詳細
Function Calling ループ：

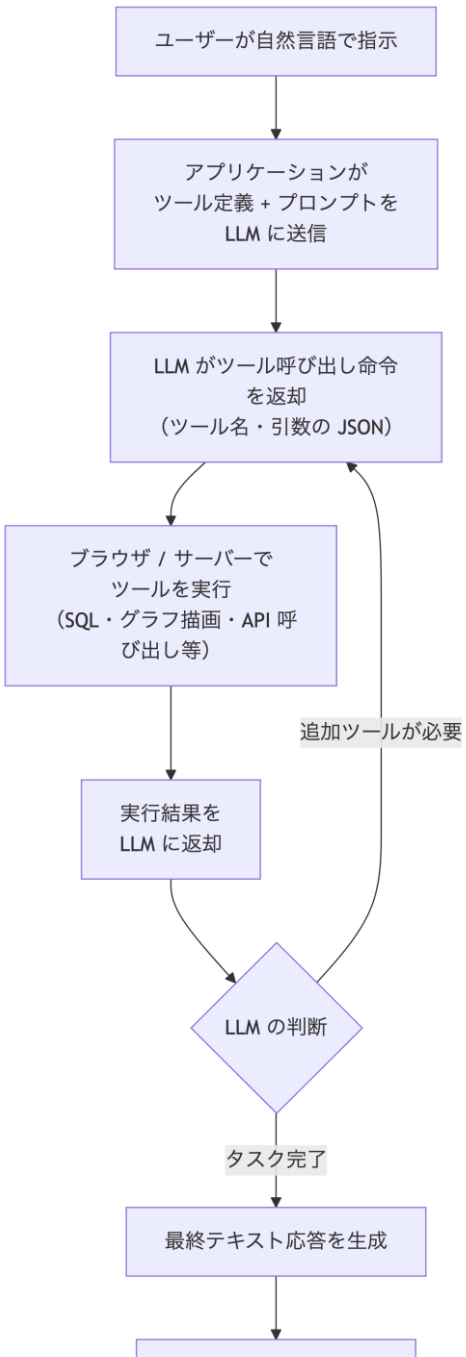

ユーザーが自然言語で指示
アプリケーションが
ツール定義 + プロンプトを
LLM に送信
LLM がツール呼び出し命令
を返却
（ツール名・引数の JSON）
ブラウザ / サーバーで
ツールを実行
（SQL・グラフ描画・API 呼
び出し等）
実行結果を
LLM に返却
追加ツールが必要
LLM の判断
タスク完了
最終テキスト応答を生成

図 2.73 Function Calling ループのフローチャート

Function Calling の動作は以下の 5 ステップで構成される。

1. **ツール定義の提供:**
   アプリケーションがツールの名前・説明・引数の JSON Schema をシステムプロンプトとともに LLM に送信する。
2. **モデルによる判断:**
   LLM は会話の文脈からどのツールをどの引数で呼び出すべきかを判断し、ツール呼び出し命令を含むレスポンスを返す。
3. **アプリケーションによる実行:**
   ブラウザまたはサーバー側がその命令を受け取り、実際の処理（DuckDB-Wasm による SQL 実行・Vega-Lite によるグラフ描画・MapLibre GL JS による地図描画・外部 API 呼び出し等）を実行する。
4. **結果の返却:**
   実行結果をツール実行結果メッセージとして LLM に送り返す。
5. **最終応答の生成/ループ継続:**
   LLM はツール実行結果をもとに、追加のツール呼び出しが必要か判断する。必要な場合はステップ 2 に戻りループを繰り返す。タスクが完了した場合は最終テキスト応答を生成する。

スキルシステムによる段階的開示（Progressive Disclosure）：
スキルシステムは、AI の汎用的な能力をドメイン固有のタスクに特化させるための拡張モジュールであり、以下の 3 段階で情報を読み込む。

1. **メタデータ（常時読み込み）：**

スキルの名前・説明をシステムプロンプトに含め、AI がどのスキルが利用可能かを常に把握する。1 スキルあたり約 100 トークンと軽量であるため、コンテキストウィンドウを圧迫しない。

2. **インストラクション（トリガー時に読み込み）：**
   ユーザーの要求がスキルに合致した場合にのみ、詳細な手順・制約・出力形式を記述したスキルファイルを読み込む。
3. **リソース（必要時に読み込み）：**
   スクリプト・テンプレート・参照資料等の補足ファイルをインストラクションから参照する機能。本機能は LINKS BI では現時点で未実装である。

プロンプトエンジニアリングによる動作制御：

- **ワンショット実行の強制**:複数ステップを要する処理では「全ステップを 1 レスポンスで完了せよ」という制約をシステムプロンプトに明示し、AI が自律的に最後まで処理を完了するよう誘導する。
- **構造化出力の強制**: JSON 形式での出力をプロンプトで強制し、AI の応答から必要な情報を確実に抽出する。
- **バリデーションループの組み込み**:生成結果の品質基準（文字数・段落数等）をツールで検証し、基準を満たさない場合は AI に修正を指示して再生成させるループを設ける。

エージェンティック AI 技術の設計思想・技術選定の背景・実装の詳細については技術検証レポート第 3 章 3.5.3 節を参照。

2.2. システムコンポーネント（CO）

2.2.1　システムコンポーネント図

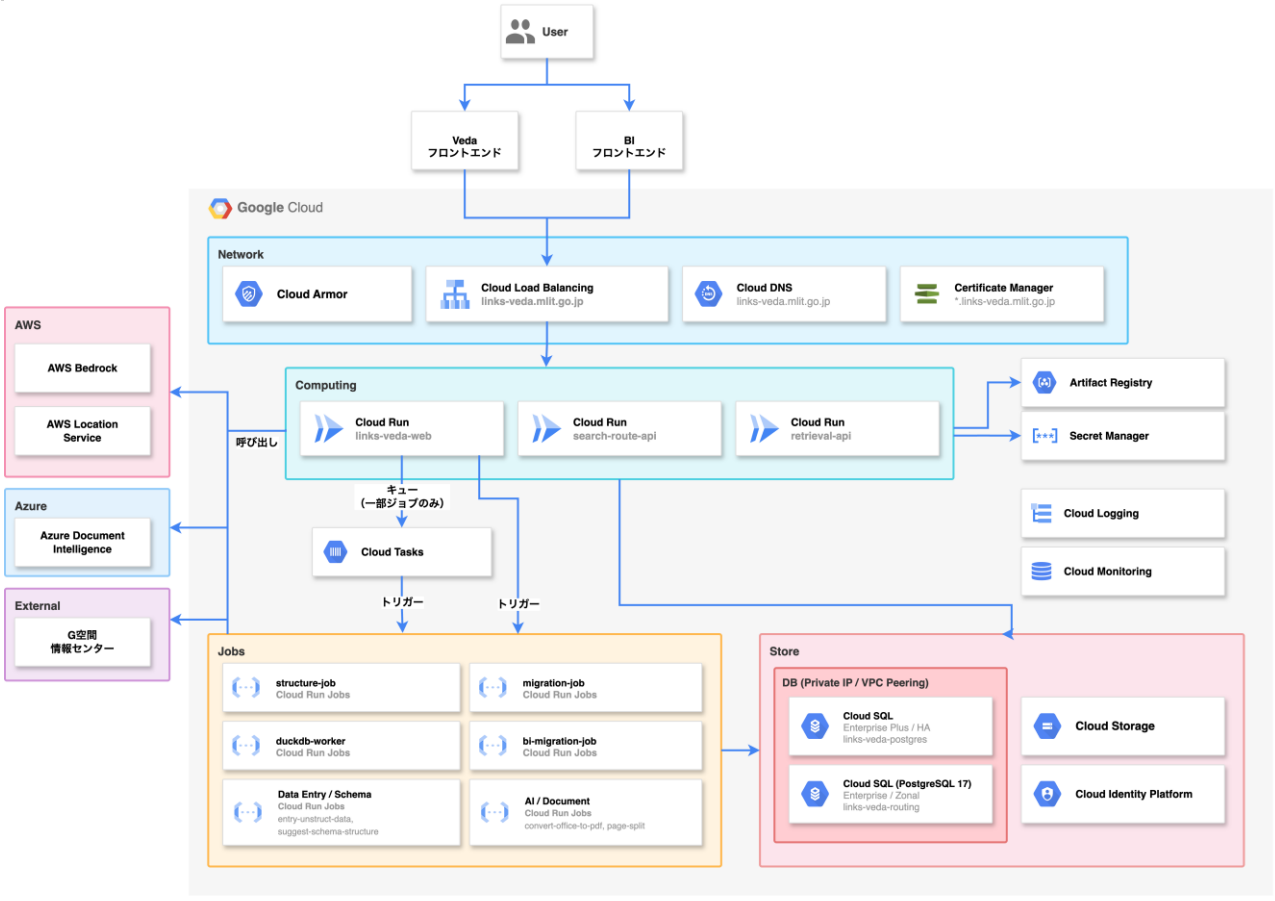


図 2.74 システムコンポーネント図

2.2.2　システムコンポーネント一覧

本節ではシステムコンポーネントの一覧を示し、各コンポーネントの詳細（バージョン・設定・デプロイ先）は 2.2.3 に記載する。

表 2.8　システムコンポーネント一覧

ID	種別	コンポーネント名	用途
CO001	ブラウザアプリケーション	データ構築モジュールフロントエンド（Remix SSR）	データ構造化・テーブルアクション・オープンデータ化の UI
CO002	ブラウザアプリケーション	LINKS BI フロントエンド（React + DuckDB-Wasm）	ブラウザ内分析・ダッシュボード・AI チャット UI
CO003	WAF	Cloud Armor	Web アプリケーションファイアウォール。DDoS 防御・WAF ルール適用
CO004	ロードバランサー	Cloud Load Balancing	HTTPS トラフィックのロードバランシング
CO005	DNS サービス	Cloud DNS	ドメイン名解決（links-veda.mlit.go.jp）

CO006	証明書管理	Certificate Manager	TLS/SSL 証明書の管理（*.links-veda.mlit.go.jp）
CO007	アプリケーションサーバー	データ構築モジュール Server（Cloud Run）	Remix で動作するサーバー。フロントエンド静的ファイルの配信、バックエンド処理の実行、データレイヤーの管理を行う
CO008	API サーバー	経路探索 API サーバー（Cloud Run）	ネットワーク経路探索処理（search-route-api）
CO009	API サーバー	検索 API サーバー（Cloud Run）	ドキュメント検索・RAG 処理（retrieval-api）
CO010	コンテナレジストリ	Artifact Registry	コンテナイメージの保存・管理
CO011	シークレット管理	Secret Manager	API キー・認証情報の安全な管理
CO012	AI サービス	AWS Bedrock Claude（Sonnet 4.5）	LLM 推論（構造化・スキーマ提案・BI チャット）
CO013	外部連携	AWS Location Service	地理空間処理・ジオコーディング・経路探索

CO014	AI サービス	Azure Document Intelligence	OCR 処理（PDF・画像テキスト・レイアウト抽出）
CO015	タスクキュー	Cloud Tasks	非同期ジョブのキュー管理・ディスパッチ
CO016	バッチ処理基盤	Cloud Run Jobs / Functions	データ構造化・スキーマ提案・クレンジング・エクスポート・マイグレーション等のバッチ処理実行基盤
CO017	データベース	PostgreSQL（Cloud SQL）	ワークフロー・ファイルメタデータ・ユーザー・チーム管理（links-veda-postgres）
CO018	データベース	PostgreSQL（Cloud SQL /経路探索用）	経路探索用データベース（links-veda-routing）
CO019	オブジェクトストレージ	Google Cloud Storage	アセットファイル・Parquet データの保存基盤
CO020	認証サービス	Cloud Identity Platform	ユーザー認証・ID トークン発行。データ構築モジュール・LINKS BI 共通

CO021	ログ収集	Cloud Logging	アプリケーション・インフラのログ収集・分析基盤
CO022	モニタリング	Cloud Monitoring	システムメトリクス監視・アラート
CO023	外部連携	G 空間情報センター	オープンデータ公開先の外部システム

2.2.3　システムコンポーネントの詳細

【CO007】データ構築モジュール Server

- **種別：** アプリケーションサーバー
- **技術スタック：** Remix ^2.15 / React ^18.3 / TypeScript ^5.7 / Prisma ^6.3
- **デプロイ先：** Cloud Run（links-veda-web）
- **役割：** Remix で動作するサーバー。フロントエンド静的ファイルの配信、バックエンド処理の実行（API エンドポイント・Remix ルート）、データレイヤーの管理を行う。DuckDB をインメモリ処理エンジンとしてプロセス内で使用し、CSV / Excel / GeoJSON / Parquet のファイルインポート・データ変換・クエリ実行を行う。
- **本番環境スペック：**
  - CPU：4 vCPU /メモリ：16 GiB
  - 最大インスタンス数：4 /同時接続数：300
  - タイムアウト：600 秒

■ リージョン：asia-northeast1（東京）

【CO017】PostgreSQL（Cloud SQL / links-veda-postgres）

- **種別：** マネージドデータベース
- **バージョン：** PostgreSQL 16
- **デプロイ先：** Cloud SQL
- **役割：** ワークフロー定義・実行履歴・ファイルメタデータ・ユーザー・チーム管理（データ構築モジュール側のデータ）。同一インスタンス上で LINKS BI のデータ（CO014）も管理するが、アクセスするアプリケーション・ORM が異なるため論理的に分離して定義する。
- **接続方式：** Prisma（Veda）経由の ORM 接続
- **CO014 との関係：** 同一の Cloud SQL インスタンス（links-veda-postgres）上で動作する。詳細は CO014 を参照。
- **本番環境スペック：**
  - ■ インスタンス名：links-veda-postgres
  - ■ バージョン：PostgreSQL 16
  - ■ マシンタイプ：db-perf-optimized-N-4（4 vCPU）
  - ■ ストレージ：SSD / 10 GB
  - ■ 可用性：REGIONAL（リージョナル高可用性・自動フェイルオーバー）
  - ■ バックアップ：毎日 11:00（自動バックアップ有効）
  - ■ 最大接続数：240
  - ■ リージョン：asia-northeast1（東京）

【CO019】Google Cloud Storage

- **種別：** オブジェクトストレージ

- **本番バケット：** 「links-veda-production-duckdb」（ASIA マルチリージョン/MULTI_REGIONAL）
- **役割：** アセットファイル・Parquet データの保存基盤。LINKS BI の DuckDB-Wasm 用 Parquet ファイルも格納。
- **ファイルパス体系：** Bronze 「bronze/{name}_{id}.{ext}」・Silver 「silver/{workflowId}/{tableId}/snapshot_N.parquet」・Gold 「gold/{datasetId}/」
- **本番環境スペック：**
  - ■ バケット名：links-veda-production-duckdb
  - ■ リージョン：ASIA（マルチリージョン）
  - ■ ストレージクラス：MULTI_REGIONAL

【CO016】データ構造化サービス（Cloud Run / DataStructure）

- **種別：** サーバーレス処理基盤（Cloud Run コンテナサービス）
- **ランタイム：** Python 3.12
- **デプロイ先：** Cloud Run / 「entry-struct-data」（構造化データ向けエントリポイント）・「entry-unstruct-data」（非構造化データ向けエントリポイント）
- **役割：** OCR + LLM による非構造データ構造化処理。非同期コールバック方式で結果を Veda に返す。2 つのサービスはエントリポイントのみ異なり、内部処理ロジックは共通のデータ構造化パイプラインを共有する。
- **本番環境スペック：**
  - ■ entry-struct-data：0.33 vCPU / 512 MiB /最大 100 インスタンス/タイムアウト 540 秒

- ■ entry-unstruct-data：0.33 vCPU / 512 MiB /最大 100 インスタンス/タイムアウト 540 秒
- ■ リージョン：asia-northeast1（東京）

【CO016】スキーマ提案サービス（Cloud Run / SchemaSuggestion）

- **種別：** サーバーレス処理基盤
- **ランタイム：** Python 3.12
- **デプロイ先：** Cloud Run / 「suggest-schema-structure」（1 vCPU / 2 GiB /タイムアウト 900s）
- **役割：** PDF の OCR+LLM 解析によるスキーマ自動提案。同期レスポンス方式
- **本番環境スペック：**
  - ■ suggest-schema-structure：1 vCPU / 2 GiB /最大 100 インスタンス/タイムアウト 900 秒
  - ■ リージョン：asia-northeast1（東京）

【CO020】Cloud Identity Platform

- **種別：** 外部認証サービス
- **提供元：** Google Cloud
- **役割：** Google Cloud の認証サービス。データ構築モジュール / LINKS BI の共通ユーザー認証基盤として使用する。Firebase Authentication と共通のクライアントライブラリを使用するが、サービスとしては別のプロダクトである。
- **統合方式：** クライアントが ID トークンを取得 → Veda が firebase-admin で検証 → Cookie セッション発行 → BI は Veda に認証を委譲。

【CO012】AWS Bedrock Claude（Sonnet 4.5）

- **種別：** 外部 AI サービス
- **提供元：** Amazon Web Services
- **モデル：** 「jp.anthropic.claude-sonnet-4-5-20250929-v1:0」
- **役割：** データ構造化（FN008）・スキーマ提案（FN007）・不正検知（FN010）・BI チャット（FN109）・SQL 説明・タイトル生成等の LLM 推論。
- **リージョン：** ap-northeast-1（東京）

【CO014】Azure Document Intelligence

- **種別：** 外部 AI サービス（OCR）
- **提供元：** Microsoft Azure
- **モデル：** 「prebuilt-layout」
- **役割：** PDF・画像のテキスト・レイアウト・テーブル抽出。データ構造化（FN008）とスキーマ提案（FN007）の OCR ステップで使用。

【CO023】LINKS BI フロントエンド（React + DuckDB-Wasm）

- **種別：** ブラウザアプリケーション（SPA）
- **技術スタック：** React ^19.2 / Vite / @duckdb/duckdb-wasm ^1.30 / @tanstack/react-router / jotai
- **デプロイ先：** 静的アセットとして配信（または Cloud Run 経由）
- **役割：** ブラウザ内 SQL 分析、ダッシュボード構築・編集、AI チャット UI、地図表示

【CO024】LINKS BI バックエンド（Hono）

- **種別：** API サーバー
- **技術スタック：** Hono ^4.10 / @hono/zod-openapi / Kysely / @ai-sdk/amazon-bedrock
- **デプロイ先：** Cloud Run（links-bi）
- **役割：** LINKS BI フロントエンドへの REST API 提供。認証（Cookie 検証）・AI チャット（Bedrock 呼び出し）・ダッシュボード管理・e-Stat 連携。
- **本番環境スペック（dev 環境確認値）：**
  - CPU：4 vCPU /メモリ：16 GiB
  - 最大インスタンス数：4 /同時接続数：300
  - タイムアウト：600 秒
  - リージョン：asia-northeast1（東京）

【CO025】PostgreSQL（BI 用 Cloud SQL）

- **種別：** マネージドデータベース
- **バージョン：** PostgreSQL 16
- **デプロイ先：** Cloud SQL — CO002 と同一の物理インスタンスだが、LINKS BI のデータを管理するコンポーネントとして論理的に分離して定義する。
- **役割：** LINKS BI のチャット履歴（「chats」・「messages」 テーブル）・ダッシュボード定義（「dashboards」 テーブル）を管理する。
- **CO002 との関係：** 同一の Cloud SQL インスタンス（links-veda-postgres）上で動作するが、アクセスするアプリケーションが異なる。CO002 はデータ構築モジ

ュール（Prisma）経由、CO014 は LINKS BI バックエンド（Kysely）経由でアクセスする。

- **接続方式：** Kysely（BI バックエンド）経由

【CO026】PostgreSQL（ベクトルストア/ Cloud SQL）

- **種別：** マネージドデータベース
- **バージョン：** PostgreSQL 16 + pgvector 拡張
- **データベース名：** 「links_veda_retrieval」
- **デプロイ先：** Cloud SQL — **CO002 と同一の物理インスタンス**上の別データベースとして稼働する。
- **役割：** DocumentIndexer（【FN045】）のベクトル埋め込み・ドキュメントメタデータ・セクション情報・チャンクテキストを格納する。HNSW インデックスによるベクトル近似最近傍検索および GIN インデックスによる全文キーワード検索を提供する。
- **CO002 との関係：** 同一の Cloud SQL インスタンス（links-veda-postgres）上で動作するが、データベース名が異なる（CO002: Veda メイン DB、CO015: links_veda_retrieval）。アクセスするアプリケーションも異なり、CO015 は Cloud Run 処理サービス（CO005）から asyncpg 経由でアクセスする。
- **接続方式：** asyncpg（【SL029】）経由。コネクションプール（min=1, max=10）を使用。SSL モードは環境変数で設定可能。
- **主要テーブル：** 「retrieval_documents」（メタデータ）・「retrieval_document_sections」（セクション情報）・「retrieval_document_chunks」（ベクトル埋め込み）。詳細は【IF017】を参照。

【CO004】Cloud Load Balancing

- **種別：** ロードバランサー

- **提供元：** Google Cloud
- **役割：** データ構築モジュール・LINKS BI への HTTPS トラフィックのロードバランシング。外部クライアントからのリクエストを適切な Cloud Run サービスにルーティングする。

【CO003】Cloud Armor

- **種別：** WAF（Web アプリケーションファイアウォール）
- **提供元：** Google Cloud
- **役割：** DDoS 防御・IP 制限・WAF ルールの適用。Cloud Load Balancing（【CO004】）と連携してトラフィックをフィルタリングする。

【CO021】Cloud Logging

- **種別：** ログ収集基盤
- **提供元：** Google Cloud
- **役割：** アプリケーション・インフラのログ収集・分析基盤。Cloud Run サービス・Cloud SQL 等のログを集約する。

【CO027】Amazon Bedrock Titan Text Embeddings V2

- **種別：** 外部 AI サービス（ベクトル埋め込み）
- **提供元：** Amazon Web Services
- **役割：** ベクトル埋め込み生成（1024 次元）。ドキュメントインデキシング（【FN045】）でドキュメントチャンクをベクトル化し、【CO026】に格納する。

【CO028】G 空間情報センター

- **種別：** 外部連携システム
- **役割：** オープンデータ公開先の外部システム。【FN043】でデータを連携する。

【CO029】e-Stat API（政府統計の総合窓口）

- **種別：** 外部連携 API
- **役割：** 政府統計データ取得のための外部 API。【FN112】で使用する。

2.3. ハードウェア（HW）

2.3.1　ハードウェアアーキテクチャ

図 2.75 ハードウェアアーキテクチャ

2.3.2　ハードウェア一覧

表 2.9　ハードウェア一覧

ID	種別	ベンダー	品番/型番	用途
HW001	デスクトップ/ラップトップ PC	—	Windows 11 Enterprise 搭載 PC	国土交通省職員がデータ構築モジュール / LINKS BI にアクセスするための端末

2.3.3　ハードウェアの詳細

【HW001】利用者 PC（Windows デスクトップ/ラップトップ）

本ハードウェアの概要

- 実証を行う国土交通省職員の標準 PC のスペックである。
- Microsoft Edge または Google Chrome ブラウザでデータ構築モジュール / LINKS BI にアクセスする。

本ハードウェアを提供するベンダー

- 国土交通省標準 PC

本ハードウェアの仕様・スペック

- CPU：AMD Ryzen（TM）3 5400U
- メモリ：32 GB
- ブラウザ：Microsoft Edge、Google Chrome
- SSD：512 GB
- ディスプレイ：13.3 インチ
- 重さ：約 1000 g
- OS：Windows 11 Enterprise

画面サイズの制約

- 推奨動作環境
  - ■ ブラウザ：1440 × 900、コンテンツ領域：1440 × 802

- 最小動作環境
  - ブラウザ：1200 × 832、コンテンツ領域：1200 × 750
- 最小サイズを下回る場合には、正しくコンテンツ表示ができず、オペレーションに影響が出る可能性がある。

2.4. データインターフェース（IF）

2.4.1　データアーキテクチャ

- 本システムのデータアーキテクチャは、ユーザーがアップロードした非構造ファイルおよび構造化・加工後のデータを、データセット／データテーブル／アセットとして一元管理し、各処理機能（FN）が共通の入出力インターフェース（IF）を介して疎結合に連携する構成とする。
- データ構造化処理（FN008）を中心に、スキーマ自動提案（FN007）、不要ページ除外（FN009）、信頼度評価・不正生成検知（FN010）、バウンディングボックス抽出・構造化結果確認（FN011）、訂正線検知（FN012）等が処理結果を受け渡しし、最終的な構造化結果はデータテーブルとして蓄積され、後段の各種テーブルアクション群（FN017〜）やエクスポート（FN042）等に接続される。

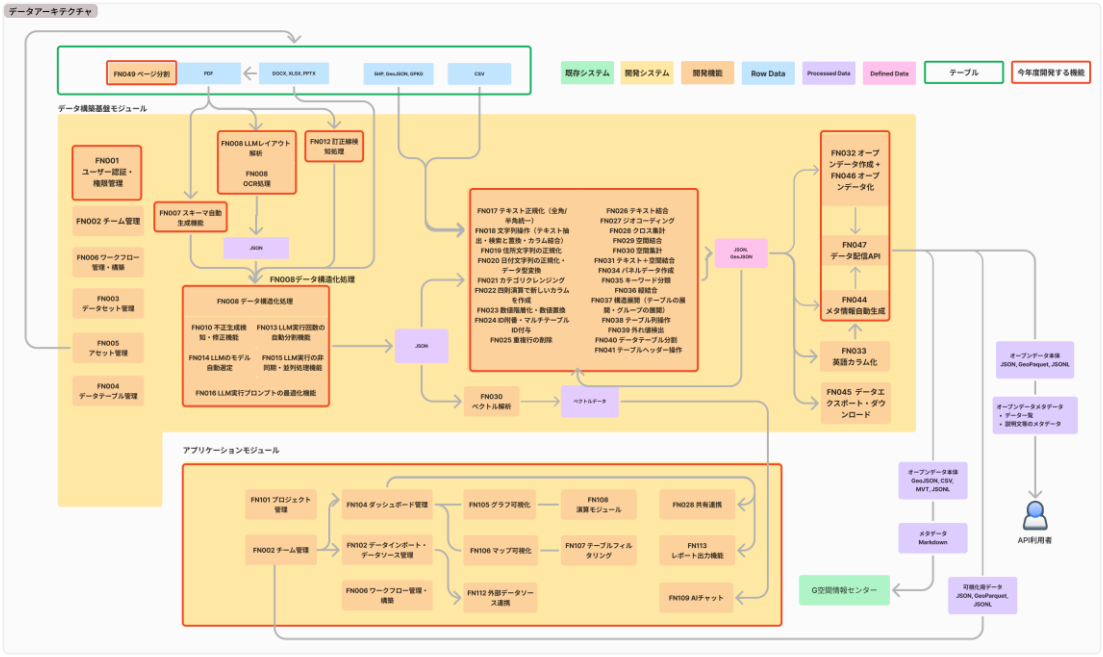


図 2.76 データアーキテクチャ

2.4.2 データインターフェース一覧

出力側・入力側の ID は 「CO（コンポーネント）」 ID で示す。「外部」は本システム外部サービスを示す。

表 2.10　データインターフェース一覧（内部 IF）

ID	名称	出力側 ID	入力側 ID
IF001	GCS ファイルストレージ	CO001 / CO005 / CO006	CO004
IF002	構造化処理 API	CO001	CO005
IF003	スキーマ提案 API	CO001	CO006
IF004	WorkflowRun コールバック API	CO005	CO001
IF005	LINKS BI AI チャット HTTP ストリーミング API	CO012	CO013
IF006	Parquet ファイル配信	CO004	CO012
IF007	PostgreSQL データアクセス	CO001 / CO013	CO002 / CO014
IF008	データセットアクション実行 API	CO001（フロントエンド）	CO001（バックエンド）
IF009	Office → PDF 変換 API	CO001	CO005
IF010	LINKS BI 汎用 REST API	CO012	CO013

IF016	GCS HMAC 認証（DuckDB 直接アクセス）	CO004	CO001
IF017	pgvector ベクトルストア	CO005	CO015
IF018	データ配信 API	CO001	外部クライアント（QGIS 等）
IF019	経路検索 API	CO013	Cloud SQL（pgRouting）

表 2.11　データインターフェース一覧（外部サービス連携 IF）

ID	名称	出力側 ID	入力側（外部サービス）
IF011	Cloud Identity Platform 認証 API	CO001 / CO012	Cloud Identity Platform（CO009）
IF012	AWS Bedrock API（Claude）	CO005 / CO006 / CO013	AWS Bedrock（CO010）
IF013	Azure Document Intelligence API	CO005 / CO006	Azure Document Intelligence（CO011）
IF014	AWS Location Service API	CO001	AWS Location Service
IF015	e-Stat API	CO013	e-Stat（CO021）
IF020	G 空間情報センター連携 API	CO001	G 空間情報センター（CO020）

2.4.3　データインターフェースの詳細

【IF001】GCS ファイルストレージ

本インターフェースの概要

- アセットファイルおよびデータテーブルをストレージに保存・取得するためのインターフェース。

本インターフェースを利用する機能

- 【FN005】アセット管理
- 【FN003】データセット管理
- 【FN008】データ構造化処理

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
PDF	.pdf	PDF バイナリ	上限：1 GB
Excel	.xlsx, .xls	Microsoft Excel	上限：1 GB
CSV	.csv	テキスト（UTF-8 / Shift_JIS）	上限：1 GB
JSON	.json	JSON テキスト	上限：1 GB
GeoJSON	.geojson	GeoJSON テキスト	上限：1 GB

Word	.docx	Microsoft Word	上限：1 GB
PowerPoint	.pptx	Microsoft PowerPoint	上限：1 GB
Parquet	.parquet	Apache Parquet（列指向圧縮）	上限：1 GB

【IF002】構造化処理 API

本インターフェースの概要

● データ構築モジュールからデータ構造化サービスへの構造化処理リクエストを送信するインターフェース。非同期コールバック方式で動作する。

本インターフェースを利用する機能

● 【FN008】データ構造化処理

本インターフェースを利用してやり取りを行うデータの詳細

API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「${CLOUD_FUNCTIONS_GATEWAY_URL}/api/structure」
レスポンスデータ形式	JSON
文字コード	UTF-8

リクエストパラメータ

名称	説明	型	必須

なし	URL パス・クエリで指定する項目はない。	-	-

リクエストボディ

名称	説明	型	必須
「input」	入力ファイルの GCS パスまたはダウンロード URL。	string	○
「page」	構造化対象ページ番号の配列	number[]	○
「schema」	抽出対象スキーマ定義配列	object[]	○
「api_endpoint」	処理完了後のコールバック URL	string	○

共通レスポンスパラメータ

名称	説明
「status」	受付結果。「ok」 または 「error」
「message」	受付メッセージ

● レスポンスデータサンプル

```
{
  "status": "ok",
  "message": "Success"
}
```

● **コールバック処理**

構造化処理の詳細結果は即時レスポンスでは返却されず、「api_endpoint」 に指定したコールバック先へ非同期で送信される。コールバックの受信仕様は【IF004】で定義する。

【IF003】スキーマ提案 API

本インターフェースの概要

● データ構築モジュールからデータ構造化サービスへのスキーマ自動提案リクエストを送信するインターフェース。同期レスポンス方式で動作する（最大 900 秒）。

本インターフェースを利用する機能

● 【FN007】スキーマ自動提案

本インターフェースを利用してやり取りを行うデータの詳細

API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「${CLOUD_FUNCTIONS_GATEWAY_URL}/api/suggest-schema-structure」
レスポンスデータ形式	JSON
文字コード	UTF-8

リクエストパラメータ

名称	説明	型	必須
なし	URL パス・クエリで指定する項目はない。	-	-

リクエストボディ

名称	説明	型	必須
「input」	解析対象 PDF の GCS パスまたはダウンロード URL。	string	○
「page」	解析対象ページ番号の配列。実装上、空配列の場合は全ページ対象として扱う。	number[]	○
「prompt」	抽出対象を絞り込むユーザー指示文	string	-
「use_ocr_llm」	OCR + LLM モードを使用するかどうか。「true」 が既定。	boolean	-

共通レスポンスパラメータ

名称	説明
「status」	処理結果。「ok」 または 「error」
「message」	補足メッセージ
「schema」	提案されたスキーマ配列

● レスポンスデータサンプル

```
{
```

```
"status": "ok",
"message": "Success",
"schema": [
  {
    "key": "請求書番号",
    "type": "string",
    "description": "請求書の識別番号"
  },
  {
    "key": "合計金額",
    "type": "number",
    "description": "税込合計金額"
  },
  {
    "key": "明細",
    "type": "objectArray",
    "description": "請求書の明細行",
    "fields": [
      {
```

```
            "key": "品名",
            "type": "string",
            "description": "明細の品名"
          },
          {
            "key": "単価",
            "type": "number",
            "description": "明細の単価"
          }
        ]
      }
  ]
}
{
  "status": "error",
  "message": "Failed to analyze document"
}
```

【IF004】WorkflowRun コールバック API

本インターフェースの概要

● データ構造化サービスからデータ構築モジュールへ構造化処理結果を返却するためのコールバックインターフェース。

本インターフェースを利用する機能

● 【FN008】データ構造化処理

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「/api/structure/new-callback/:workflowRunId」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須

「workflowRunId」	コールバック対象の WorkflowRun ID	string	○

● リクエストボディ

名称	説明	型	必須
「status」	構造化処理結果のステータス	string	○
「message」	補足メッセージ	string	-
「data」	構造化抽出結果配列	object[]	○

● **共通レスポンスパラメータ**

名称	説明
「status」	コールバック受付結果。「ok」 または 「error」

● **レスポンスデータサンプル**


```
{
  "status": "ok"
}
```


● **コールバックのリクエストボディサンプル**


```
{
  "status": "ok",
  "message": "Success",
  "data": [
    {
      "key": "title",
      "type": "string",
      "bbox": [
        [0.7757, 0.8314],
        [1.3435, 0.8314],
```

```
        [1.3435, 0.9784],
        [0.7757, 0.9784]
      ],
      "page": 1,
      "value": "Extracted title value from document",
      "confidence": 0.43
    },
    {
      "key": "company info",
      "type": "object",
      "bbox": [
        [0.6043, 3.8134],
        [7.6639, 3.8134],
        [7.6639, 5.0177],
        [0.6043, 5.0177]
      ],
      "page": 1,
      "value": {
        "name": "Extracted name value from document",
```

```
        "capital": 1.6,
        "whether listed": true
      },
      "confidence": 0.31
    },
    {
      "key": "salary table",
      "type": "objectArray",
      "bbox": [
        [0.6043, 3.8134],
        [7.6639, 3.8134],
        [7.6639, 8.8813],
        [0.6043, 8.8813]
      ],
      "page": 1,
      "value": [
        {
          "name": "Extracted name value from document",
          "salary": 13.5,
```

```
            "bonus": false
          },
          {
            "name": "Extracted name value from document",
            "salary": 71.8,
            "bonus": true
          }
        ],
        "confidence": 0.09
      }
    ]
}
```

● 構造化抽出結果配列 「data[]」 の内容

名称	説明	型	必須
「key」	抽出対象のスキーマ項目名	string	○
「type」	抽出値の型。「string」、「number」、「boolean」、「object」、「objectArray」 等	string	○
「value」	抽出された値	string | number | boolean | object | object[]	○
「bbox」	元文書上の座標情報。4 点の座標配列で返却される。	number[][]	-
「page」	抽出元ページ番号	number	○
「confidence」	不正検知スコア	number	-

【IF005】LINKS BI AI チャット HTTP ストリーミング API

本インターフェースの概要

● LINKS BI フロントエンドとバックエンド間の AI チャット通信インターフェース。
HTTP ストリーミング方式

本インターフェースを利用する機能

● 【FN109】AI チャット

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「/bi-api/teams/:teamId/chats/:chatId/stream」
レスポンスデータ形式	Server-Sent Events
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須

「teamId」	チャットを保持するチーム ID	string	○
「chatId」	ストリーミング対象のチャット ID	string	○

● **リクエストボディ**

名称	説明	型	必須
「messages」	ai-sdk UI Messages 形式のメッセージ配列。	object[]	○
「system」	システムプロンプト	string	-
「maxOutputTokens」	最大出力トークン数	number	-
「temperature」	生成温度	number	-
「tools」	LLM に公開するツール定義	object	-

● **共通レスポンスパラメータ**

名称	説明
「type」	ストリームイベント種別。「tool_call」、「tool_result」、「text」 等。
「content」	テキストイベント時の本文。
「name」	ツールイベント時のツール名。
「input」	ツール呼び出し時の入力。

● **レスポンスデータサンプル**

data: {"type":"tool_call","name":"duckdb_query","input":{"query":"SELECT * FROM table_name LIMIT 10"}}

data: {"type":"text","content":"対象データの先頭 10 件を取得しました。"}

【IF006】Parquet ファイル配信

本インターフェースの概要

● ストレージに保存された Parquet ファイルを LINKS BI フロントエンドに配信するインターフェース。

本インターフェースを利用する機能

● 【FN102】データインポート・データソース管理

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
データセット	.parquet	Apache Parquet（列指向・Snappy 圧縮）	DuckDB 互換スキーマ。 列名・型はワークフローのスキーマ設定による。
CSV キャッシュ	.csv	CSV（UTF-8）	サーバーサイドで事前生成。

【IF007】PostgreSQL データアクセス

本インターフェースの概要

- データ構築モジュールおよび LINKS BI からリレーショナルデータベースへのデータアクセスインターフェース。データ構築モジュールは Prisma ORM、LINKS BI は Kysely を使用して接続する。

本インターフェースを利用する機能

- 【FN001】ユーザー認証・権限管理
- 【FN002】チーム管理
- 【FN003】データセット管理
- 【FN004】データテーブル管理
- 【FN005】アセット管理
- 【FN006】ワークフロー管理
- 【FN008】データ構造化処理
- 【FN101】BI ユーザー認証
- 【FN103】ダッシュボード管理
- 【FN109】AI チャット

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲

データ構築モジュールメタデータ	—	PostgreSQL テーブル（users, teams, workflows, workflow_runs, files, folders, datasets, background_jobs 等）	PostgreSQL 16 互換
LINKS BI メタデータ	—	PostgreSQL テーブル（chats, messages, dashboards 等）	PostgreSQL 16 互換

【IF008】データセットアクション実行 API

本インターフェースの概要

● データ構築モジュールフロントエンドからバックエンドへのデータアクション実行リクエストインターフェース。アクション件数に応じて非同期ジョブキューまたは同期実行を行う。

本インターフェースを利用する機能

● 【FN003】データセット管理

● 【FN043】オープンデータ化

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「/api/datasets/:datasetId/execute-actions」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須
「datasetId」	対象データセット ID	string	○

● リクエストボディ

名称	説明	型	必須
「actions」	実行対象アクション配列	object[]	○
「actions[].type」	実行するアクション種別	string	○
「actions[].key」	対象カラム名	string	○
「actions[].normalizationOption」	文字正規化オプション等の追加設定。	string	-

● 共通レスポンスパラメータ

名称	説明
「success」	処理成否
「async」	非同期ジョブとして受け付けたかどうか。
「message」	実行結果メッセージ
「data.jobId」	非同期実行時のジョブ ID
「data.status」	非同期実行時のジョブ状態
「data.datasetId」	対象データセット ID

● レスポンスデータサンプル

```
{
```

```
  "success": true,
  "async": true,
  "message": "Dataset actions job queued",
  "data": {
    "jobId": "01JXXXXX",
    "status": "queued",
    "datasetId": "dataset-123"
  }
}
```

【IF009】Office → PDF 変換 API

本インターフェースの概要

● Word・Excel・PowerPoint ファイルを PDF に変換する Cloud Run 処理サービスインターフェース。

本インターフェースを利用する機能

● 【FN008】データ構造化処理

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容

プロトコル	HTTPS
メソッド	POST
エンドポイント	「${CLOUD_FUNCTIONS_GATEWAY_URL}/api/convert-office-file-to-pdf」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須
なし	URL パス・クエリで指定する項目はない。	-	-

● リクエストボディ

名称	説明	型	必須
「input」	変換対象 Office ファイルの GCS パスまたはダウンロード URL。	string	○
「api_endpoint」	変換完了後のコールバック URL	string	○

● **共通レスポンスパラメータ**

名称	説明
「status」	受付結果。正常時は 「ok」
「message」	受付メッセージ

● **レスポンスデータサンプル**


```
{
  "status": "ok",
  "message": "Conversion started"
}
```

【IF010】LINKS BI 汎用 REST API

本インターフェースの概要

- LINKS BI フロントエンド（React SPA）から Hono バックエンドへの REST API の総称。AI チャット以外のすべての操作を含む。

本インターフェースを利用する機能

- 【FN101】BI ユーザー認証
- 【FN102】データインポート・データソース管理
- 【FN103】ダッシュボード管理
- 【FN109】AI チャット
- 【FN112】e-Stat 連携

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
リクエスト/レスポンス	—	JSON（REST API 標準形式）	エンドポイントごとに異なる。
ファイルダウンロード	.parquet, .csv	バイナリストリーム/テキスト	ファイルサイズに依存。

【IF011】Cloud Identity Platform 認証 API

本インターフェースの概要

- Cloud Identity Platform を使ったユーザー認証インターフェース。データ構築モジュール・LINKS BI 共通の認証基盤。

本インターフェースを利用する機能

- 【FN001】ユーザー認証・権限管理
- 【FN101】BI ユーザー認証

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
認証リクエスト	—	Firebase Auth SDK プロトコル（メール/パスワード または OAuth。）	—
ID トークン	—	JWT（JSON Web Token）	有効期限：1 時間
セッション Cookie	—	HTTP Cookie	サーバーサイドで検証

【IF012】AWS Bedrock API（Claude）

本インターフェースの概要

- AWS Bedrock 経由で Claude モデルを呼び出すインターフェース。データ構造化・スキーマ提案・BI チャットで使用する。

本インターフェースを利用する機能

- 【FN007】スキーマ自動提案
- 【FN008】データ構造化処理
- 【FN109】AI チャット

本インターフェースを利用してやり取りを行うデータの詳細

- **API の共通仕様**

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	AWS Bedrock Runtime 「InvokeModel」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須
「modelId」	呼び出すモデル ID。実装では「MODEL_ID」 環境変数または各サービス設定値を使用する。	string	○
「contentType」	リクエストボディ形式。「application/json」 を指定する。	string	○
「accept」	レスポンス形式。「application/json」を指定する。	string	○

● リクエストボディ

名称	説明	型	必須
「anthropic_version」	Bedrock Anthropic Messages API バージョン	string	○
「max_tokens」	最大出力トークン数	number	○
「temperature」	生成温度。データ構造化・スキーマ提案では 0 を使用する。	number	○
「system」	システムプロンプト	string	○
「messages」	メッセージ配列	object[]	○

「messages[].role」	メッセージ送信者。実装では 「user」 を指定する。	string	○
「messages[].content」	入力本文。「text」 パートと必要に応じて 「image」 パートを含む。	object[]	○
「messages[].content[].type」	コンテンツ種別。「text」 または 「image」。	string	○
「messages[].content[].text」	テキスト入力本文。OCR 結果やスキーマ定義、補助プロンプトを含む。	string	-

● **共通レスポンスパラメータ**

名称	説明
「content」	モデルの出力内容
「usage.input_tokens」	入力トークン数
「usage.output_tokens」	出力トークン数
「stop_reason」	応答停止理由

● **リクエストボディサンプル**

```
{
  "anthropic_version": "bedrock-2023-05-31",
  "max_tokens": 64000,
  "temperature": 0,
  "system": "You are a document extraction assistant.",
  "messages": [
    {
      "role": "user",
      "content": [
        {
          "type": "text",
```

```
            "text": "OCR TEXT FROM DOCUMENT PAGES:¥nCompany: Example Inc.¥nTotal: 12345"
          }
        ]
      }
    ]
}
```

● **レスポンスデータサンプル**


```
{
  "content": [
    {
      "type": "text",
      "text": "{¥"title¥":¥"Example Inc.¥",¥"total¥":12345}"
    }
  ],
  "usage": {
    "input_tokens": 1520,
    "output_tokens": 210
  },
  "stop_reason": "end_turn"
}
```

【IF013】Azure Document Intelligence API

本インターフェースの概要

● Azure Document Intelligence（旧 Form Recognizer）を使った PDF・画像のテキスト・レイアウト抽出インターフェース。

本インターフェースを利用する機能

● 【FN007】スキーマ自動提案

● 【FN008】データ構造化処理

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	Azure Document Intelligence 「begin_analyze_document」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須

「model_id」	解析モデル ID。実装では 「prebuilt-layout」 を使用する。	string	○
「pages」	対象ページ番号。複数ページ時は 「1,2,4」 のようなカンマ区切り文字列で指定する。	string	-

● **リクエストボディ**

名称	説明	型	必須
「body」	OCR 対象ファイル本体。PDF または画像バイナリを送信する。	binary	○

● **共通レスポンスパラメータ**

名称	説明
「content」	文書全体の連結テキスト
「pages」	ページごとの OCR 結果配列
「pages[].lines」	行単位の抽出結果
「pages[].words」	単語単位の抽出結果
「tables」	テーブル抽出結果
「tables[].cells」	セル単位の抽出結果

● **リクエストサンプル**

```
poller = client.begin_analyze_document(
    "prebuilt-layout",
    body=file_stream,
    pages="1,2,4"
```

```
)
```

● **レスポンスデータサンプル**


```
{
  "content": "Company Example Inc. Total 12345",
  "pages": [
    {
      "page_number": 1,
      "width": 8.27,
      "height": 11.69,
      "unit": "inch",
      "lines": [
        {
          "text": "Company Example Inc."
        }
      ],
      "words": [
        {
          "text": "Company"
        }
      ]
```

```
    }
  ],
  "tables": [
    {
      "row_count": 2,
      "column_count": 2,
      "cells": [
        {
          "row_index": 0,
          "column_index": 0,
          "content": "品名"
        },
        {
          "row_index": 0,
          "column_index": 1,
          "content": "金額"
        }
      ]
    }
```

```
  ]
}
```

【IF014】AWS Location Service API

本インターフェースの概要

● 住所テキストを WKT 形式の座標に変換するジオコーディングインターフェース。

本インターフェースを利用する機能

● 【FN003】データセット管理

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「${AWS_LOCATION_SERVICE_API_ENDPOINT}${AWS_LOCATION_SERVICE_API_KEY}」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須

なし	URL パス・クエリで指定する項目はない	-	-

● リクエストボディ

名称	説明	型	必須
「Text」	ジオコーディング対象の住所文字列	string	○

● 共通レスポンスパラメータ

名称	説明
「Results」	候補地点配列
「Results[].Place.Geometry.Point」	「[経度，緯度]」 形式の座標

● **レスポンスデータサンプル**


```
{
  "Results": [
    {
      "Place": {
        "Geometry": {
          "Point": [139.7528, 35.675]
        }
      }
    }
  ]
}
```

【IF015】e-Stat API（政府統計）

本インターフェースの概要

● 政府統計の総合窓口（e-Stat）が提供する統計データ取得 API。

本インターフェースを利用する機能

● 【FN112】e-Stat 連携

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	GET
エンドポイント	「/bi-api/estat/indicators」、「/bi-api/estat/data」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ（統計指標検索）

名称	説明	型	必須
「keyword」	指標検索キーワード	string	○

● **リクエストパラメータ（統計データ取得）**

名称	説明	型	必須
「indicatorCode」	統計指標コード	string	○
「regionalRank」	地域階層コード	string	-
「cycle」	時系列周期コード	string	-
「time」	対象時点・期間	string	-

● 共通レスポンスパラメータ

名称	説明
「GET_META_INDICATOR_INF」	指標検索時に返るメタデータ全体。
「GET_STATS_DATA」	統計データ取得時に返るデータ本体。
「error」	プロキシ API でエラーが発生した場合のメッセージ。

● レスポンスデータサンプル


```
{
  "GET_META_INDICATOR_INF": {
    "METADATA_INF": {
      "CLASS_INF": {
        "CLASS_OBJ": [
          {
            "@code": "0201010000000010000",
            "@name": "人口総数"
          }
        ]
      }
```

```
    }
  }
}
```

【IF016】GCS HMAC 認証（DuckDB 直接アクセス）

本インターフェースの概要

- サーバーサイド DuckDB（【CO007】内部で動作）が 「httpfs」 拡張を使って GCS に直接アクセスするためのインターフェース。Workload Identity ではなく HMAC 認証を使用する。

本インターフェースを利用する機能

- 【FN003】データセット管理
- 【FN004】データテーブル管理

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
読み取りデータ	.parquet	Apache Parquet（列指向圧縮）	GCS バケット内の Parquet ファイル。
認証情報	—	DuckDB CREATE SECRET（GCS HMAC アクセス ID・シークレットキー）	—

【IF017】pgvector ベクトルストアインターフェース

本インターフェースの概要

- PostgreSQL + pgvector（【CO026】）によるベクトル埋め込みの格納・検索インターフェース。ドキュメントメタデータ・セクション情報・チャンクテキスト＋ベクトルを格納し、ハイブリッド検索で利用する。

本インターフェースを利用する機能

- 【FN045】ドキュメントインデキシング
- 【FN110】国会答弁検索

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
ドキュメントメタデータ	—	PostgreSQL テーブル（retrieval_documents — ID・チームID・ファイル名・タイトル・要約・タグ等）	テキスト・配列型

セクション情報	—	PostgreSQL テーブル（retrieval_document_sections — セクション見出し・ページ番号・要約）	テキスト型
ベクトル埋め込み	—	PostgreSQL テーブル（retrieval_document_chunks — テキスト・VECTOR(1024)・レベル・メタデータ）	各次元 float32、チャンクサイズ最大 1000 文字。

【IF018】データ配信 API

本インターフェースの概要

● 内部利用の API として、認証済みアクセストークンによるアクセス制御を備え、特定のチームに所属するデータテーブルを取得するためのインターフェース。QGIS 等の GIS ツールからの利用を想定する。

本インターフェースを利用する機能

● 【FN044】データ配信 API

本インターフェースを利用してやり取りを行うデータの詳細

種類	拡張子	データ形式	データ範囲
CSV	.csv	テキスト（UTF-8）	上限：1 GB
Parquet	.parquet	Apache Parquet	上限：1 GB

【IF019】経路検索 API

本インターフェースの概要

● 出発地・目的地を指定して最適経路を検索するための API インターフェース。pgRouting（pgr_bdAstar、双方向 A* アルゴリズム）を使用し、OpenStreetMap ベースの道路ネットワーク上で経路計算を行う。

本インターフェースを利用する機能

● 【FN115】GIS 機能（経路検索）

本インターフェースを利用してやり取りを行うデータの詳細

● API の共通仕様

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「${BI_ROUTE_API_BASE_URL}/api/search-route」
レスポンスデータ形式	JSON
文字コード	UTF-8

● リクエストパラメータ

名称	説明	型	必須

なし	URL パス・クエリで指定する項目はない	-	-

● **リクエストボディ**

名称	説明	型	必須
「mode」	検索対象輸送モード配列。 「truck_only」、「truck_ship」、 「truck_train」、 「truck_train_ship」、 「truck_ship_train」 を指定する。	string[]	○
「origin_name」	出発地点名	string	○
「origin_lat」	出発地点緯度	number	○
「origin_lon」	出発地点経度	number	○
「destination_name」	到着地点名	string	○
「destination_lat」	到着地点緯度	number	○
「destination_lon」	到着地点経度	number	○
「departure_hour」	出発時刻の時	number	○
「weight_tons」	貨物重量（トン）	number	○
「max_transfers」	最大乗り継ぎ回数	number	-

「show_all」	候補をすべて返すかどうか。	boolean	-
「find_station_radius_km」	鉄道駅探索半径（km）	number	-
「find_port_radius_km」	港探索半径（km）	number	-

● **共通レスポンスパラメータ**

名称	説明
「origin_name」	出発地点名
「origin_lat」	出発地点緯度
「origin_lon」	出発地点経度
「destination_name」	到着地点名
「destination_lat」	到着地点緯度
「destination_lon」	到着地点経度
「results」	経路候補配列

● **レスポンスデータサンプル**


```
{
  "origin_name": "東京港",
  "origin_lat": 35.6167,
  "origin_lon": 139.7667,
  "destination_name": "大阪港",
  "destination_lat": 34.65,
  "destination_lon": 135.4,
```

```
  "results": [
    {
      "mode": "truck_ship",
      "departure_time": "08:00",
      "arrival_time": "18:30",
      "total_time_minutes": 630,
      "total_move_time_minutes": 540,
      "total_distance_km": 527.4,
      "total_co2_emissions_grams": 182340,
      "minimum_co2_flag": true,
      "geojson": {
        "type": "FeatureCollection",
        "features": []
      }
    }
  ]
}
```

【IF020】G 空間情報センター連携 API

本インターフェースの概要

- データ構築モジュールから G 空間情報センターへデータとメタデータを連携するための外部システムインターフェース。オープンデータ公開のためのデータ登録・メタデータ送信を行う。

本インターフェースを利用する機能

- 【FN043】オープンデータ化

本インターフェースを利用してやり取りを行うデータの詳細

- **API の共通仕様**

項目	内容
プロトコル	HTTPS
メソッド	POST
エンドポイント	「${G_SPATIAL_ENDPOINT}/package_create」、 「${G_SPATIAL_ENDPOINT}/package_patch」、 「${G_SPATIAL_ENDPOINT}/resource_create」

レスポンスデータ形式	JSON
文字コード	UTF-8

● 共通リクエストヘッダ

名称	説明	型	必須
「Content-Type」	リクエストボディの形式。	string	○
「X-CKAN-API-Key」	G 空間情報センターCKAN Action API の API キー。	string	○

● **リクエストボディ（パッケージ作成・更新）**

名称	説明	型	必須
「name」	データセットのカスタム URL 名。作成時に使用する。	string	○
「id」	更新対象パッケージ ID。更新時に使用する。	string	○
「title」	データセット名	string	○
「notes」	データセット説明文	string	○
「author」	作成者名	string	○
「author_email」	作成者メールアドレス	string	○
「maintainer」	保守担当者名	string	○
「maintainer_email」	保守担当者メールアドレス	string	○
「license_id」	ライセンス ID	string	○
「tags」	タグ配列	object[]	-
「owner_org」	所有組織 ID	string	○

「private」	非公開パッケージとして作成するかどうか。	boolean	○
「version」	バージョン文字列	string	-
「type」	CKAN 上のデータセット種別	string	-
「groups」	グループ配列	object[]	-
「spatial」	空間範囲情報	string	-
「quality」	品質情報	string	-
「restriction」	利用制約	string	-
「registerd_date」	登録日	string	-
「area」	対象地域	string	-
「fee」	提供条件・料金情報	string	-
「license_agreement」	ライセンス同意条件	string	-

● **リクエストボディ（リソース作成）**

名称	説明	型	必須
「package_id」	リソースを紐付けるパッケージ ID	string	○
「name」	リソース名	string	○
「url」	リソース URL	string	-
「description」	リソース説明	string	-
「format」	ファイル形式	string	-
「metadata_type」	メタデータ種別	string	-
「data_crs」	座標参照系	string	-
「standard_price」	標準価格	string	-
「acknowledgement」	出典表記	string	-
「tos」	利用規約	string	-
「selection_type」	選定区分	string	-

● **共通レスポンスパラメータ**

名称	説明

「success」	API 呼び出し成否
「result.id」	作成または更新されたパッケージ/リソース ID。
「error」	エラー発生時の内容

● **レスポンスデータサンプル**


```
{
  "success": true,
  "result": {
    "id": "3f0c2d62-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
  }
}
```

2.5. ユーザーインターフェース（UI）

2.5.1　画面遷移図

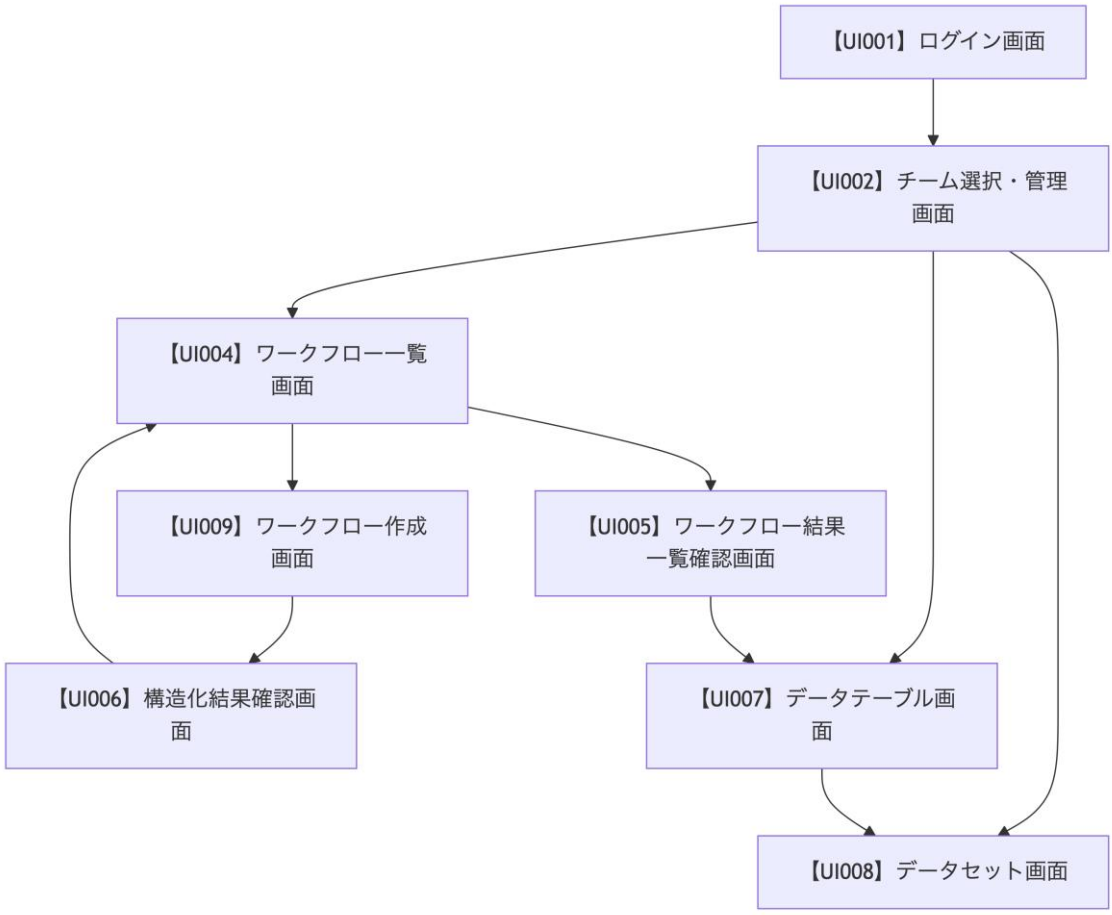


図 2.77 画面遷移図（データ構築モジュール）

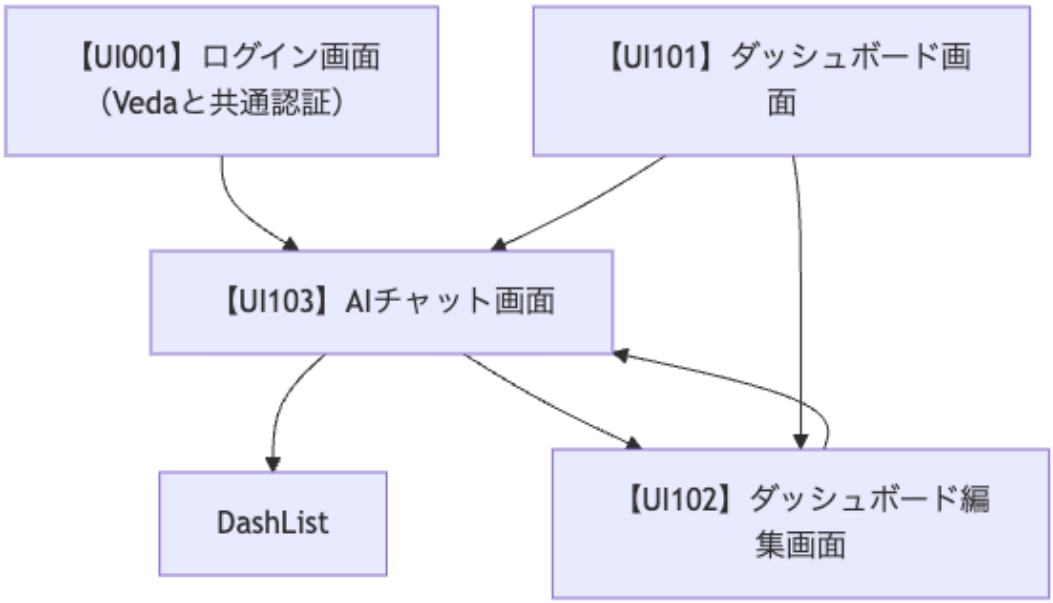


図 2.78 画面遷移図（LINKS BI）

2.5.2　ユーザーインターフェース一覧

表 2.12　画面一覧（データ構築モジュール）

ID	画面名	説明	本画面から利用する機能

UI001	ログイン画面	メール/パスワードまたは OAuth でログインする画面。	【FN001】
UI002	チーム選択・管理画面	チームの選択・作成・メンバー管理を行う画面。	【FN002】
UI004	ワークフロー一覧画面	ワークフロー一覧・ファイルアップロード・ワークフロー管理の起点画面。	【FN005】 【FN006】
UI009	ワークフロー作成画面	スキーマ自動提案・ファイルアップロード・ワークフロー新規作成を行う画面。	【FN005】 【FN006】 【FN007】
UI005	ワークフロー結果一覧確認画面	ワークフロー実行結果の確認・スキーマ設定・スキーマ自動提案を行う画面。	【FN006】 【FN007】

UI006	構造化結果確認画面	データ構造化処理の結果を確認する画面。	【FN008】 【FN009】 【FN010】 【FN011】 【FN012】 【FN013】 【FN014】 【FN015】 【FN016】

UI007	データテーブル画面	データテーブル一覧・詳細・テーブルアクション実行・エクスポートを行う画面。	【FN003】 【FN004】 【FN017】 【FN018】 【FN019】 【FN020】 【FN021】 【FN022】 【FN023】 【FN024】 【FN025】 【FN026】 【FN027】 【FN028】 【FN029】 【FN030】 【FN031】 【FN032】 【FN033】 【FN034】 【FN035】 【FN036】

			【FN037】 【FN038】 【FN039】 【FN040】 【FN042】
UI008	データセット画面	オープンデータ化・メタデータ管理・G 空間情報センター連携・データ配信設定を行う画面。	【FN003】 【FN041】 【FN042】 【FN043】 【FN044】

表 2.13　画面一覧（LINKS BI）

ID	画面名	説明	本画面から利用する機能
UI101	ダッシュボード画面	チームで作成したダッシュボードを一覧表示し、新規作成・削除を行う画面。	【FN104】
UI102	ダッシュボード編集画面	ダッシュボード編集・ウィジェット配置・データソース設定・共有設定・レポート出力を行う画面。	【FN102】 【FN104】 【FN105】 【FN106】 【FN107】 【FN113】 【FN114】

UI103	AI チャット画面	自然言語によるデータ分析・可視化・統計処理・国会答弁・GIS 分析・ファイル検索を行う AI チャット画面。	【FN105】 【FN106】 【FN107】 【FN108】 【FN109】 【FN110】 【FN111】 【FN112】 【FN115】 【FN116】

2.5.3　ユーザーインターフェースの詳細

以下に、ユーザーインターフェース（画面）の詳細を記す。

【UI001】ログイン画面

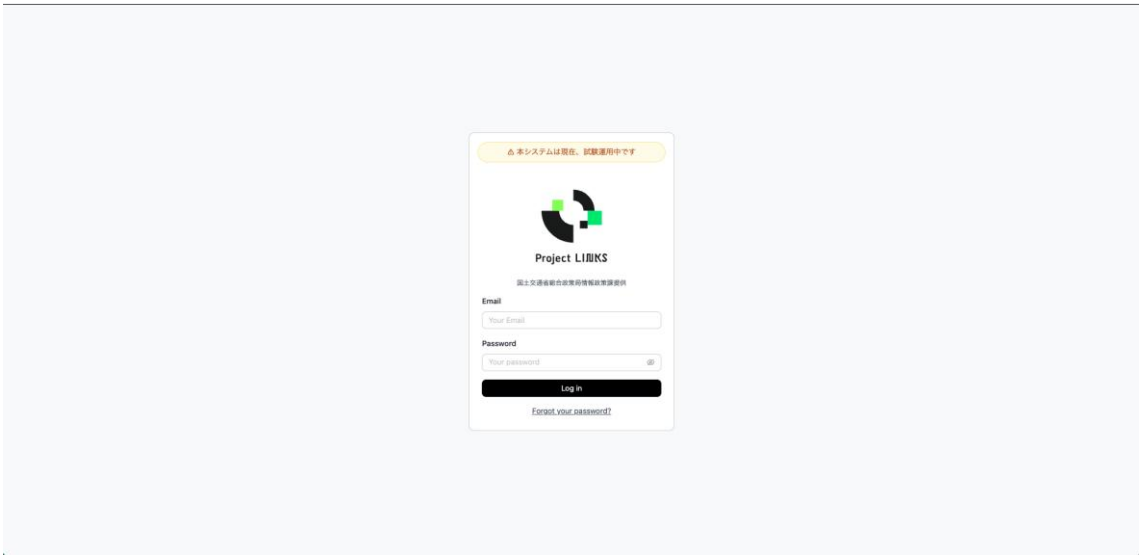


図 2.79　【UI001】ログイン画面

本画面の概要

Cloud Identity Platform によるメール/パスワードログインおよび OAuth ログインを提供する画面。初回アクセス時にリダイレクトされる。ログイン成功後はチーム選択画面（UI002）に遷移する。

本画面から利用する機能

- 【FN001】ユーザー認証・権限管理

【UI002】チーム選択・管理画面

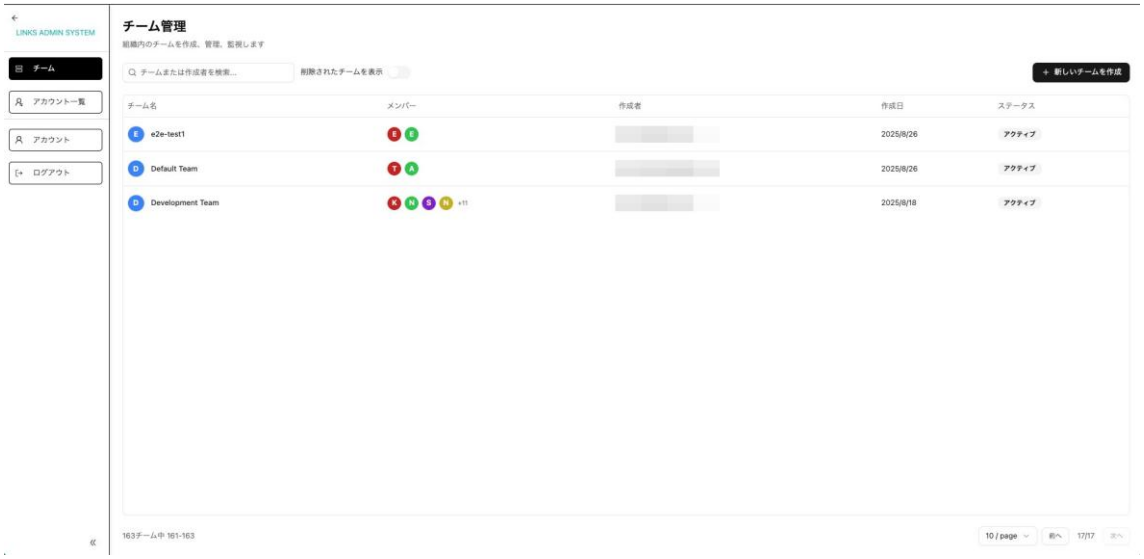


図 2.80 【UI002】チーム選択・管理画面

本画面の概要

ユーザーが所属するチームの一覧を表示し、作業対象チームを選択する画面。チームの新規作成・メンバー管理（招待・ロール変更・削除）・チーム設定変更も行える。チーム選択後はワークフロー一覧（UI004）、データテーブル画面（UI007）、データセット画面（UI008）、LINKS BI の各画面へ遷移する。

本画面から利用する機能

- 【FN002】チーム管理

【UI004】ワークフロー一覧画面

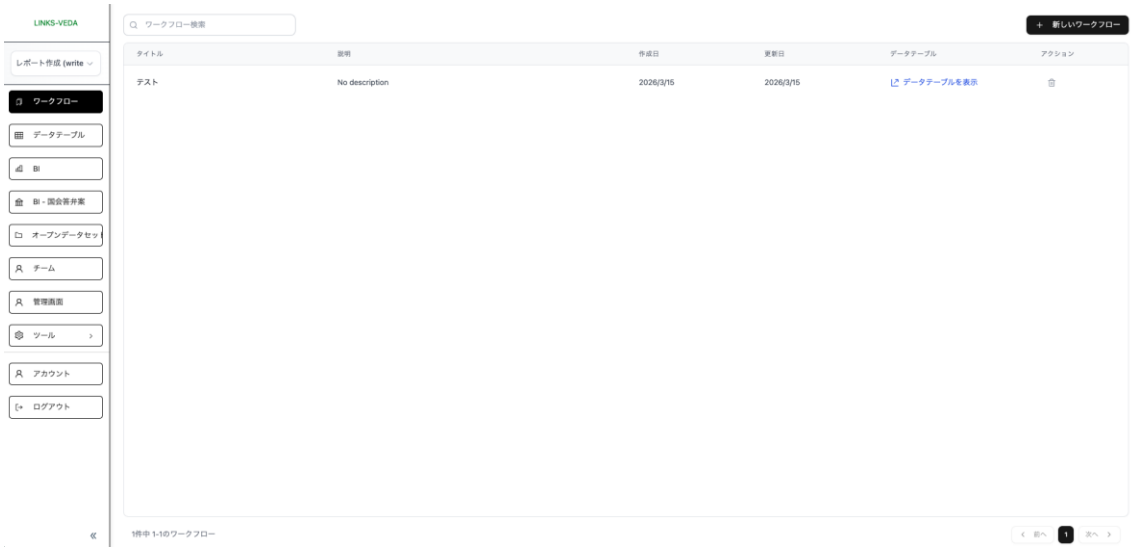


図 2.81 【UI004】ワークフロー一覧画面

本画面の概要

ワークフロー一覧を表示し、ファイルのアップロードやワークフローの管理を行う起点画面。ワークフローの新規作成・検索・複製・削除を行える。

本画面から利用する機能

- 【FN005】アセット管理
- 【FN006】ワークフロー管理・構築

【UI009】ワークフロー作成画面

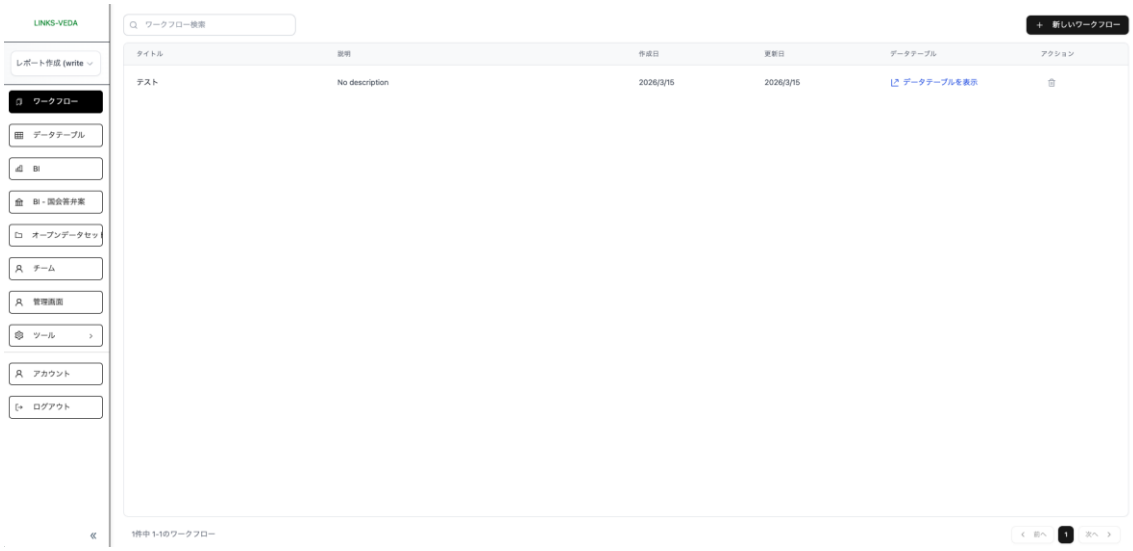


図 2.84 【UI009】ワークフロー作成画面

本画面の概要

ワークフローの新規作成を行う画面。ファイルのアップロードとスキーマ自動提案（【FN007】）の実行を行い、データ構造化のためのワークフロー設定を作成する。

本画面から利用する機能

- 【FN005】アセット管理
- 【FN006】ワークフロー管理・構築
- 【FN007】スキーマ自動提案

【UI005】ワークフロー結果一覧確認画面

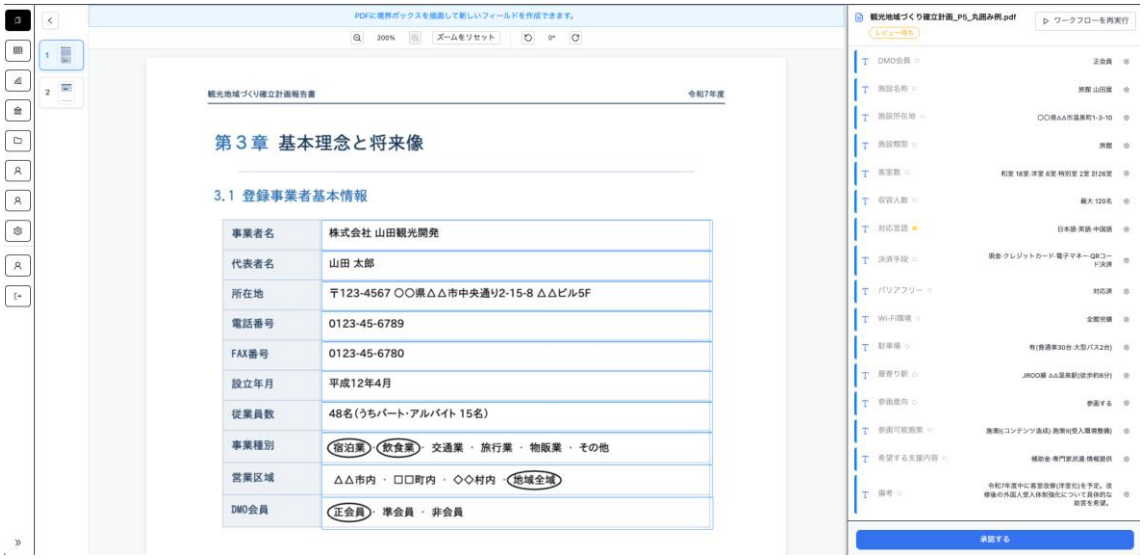


図 2.82 【UI005】ワークフロー結果一覧確認画面

本画面の概要

ワークフローの実行結果の確認やスキーマ設定を行う画面。スキーマ自動提案（【FN007】）を呼び出してフィールド候補を取得し、ユーザーが確認・編集後にワークフロー設定として保存する。

本画面から利用する機能

- 【FN006】ワークフロー管理・構築
- 【FN007】スキーマ自動提案

【UI006】構造化結果確認画面

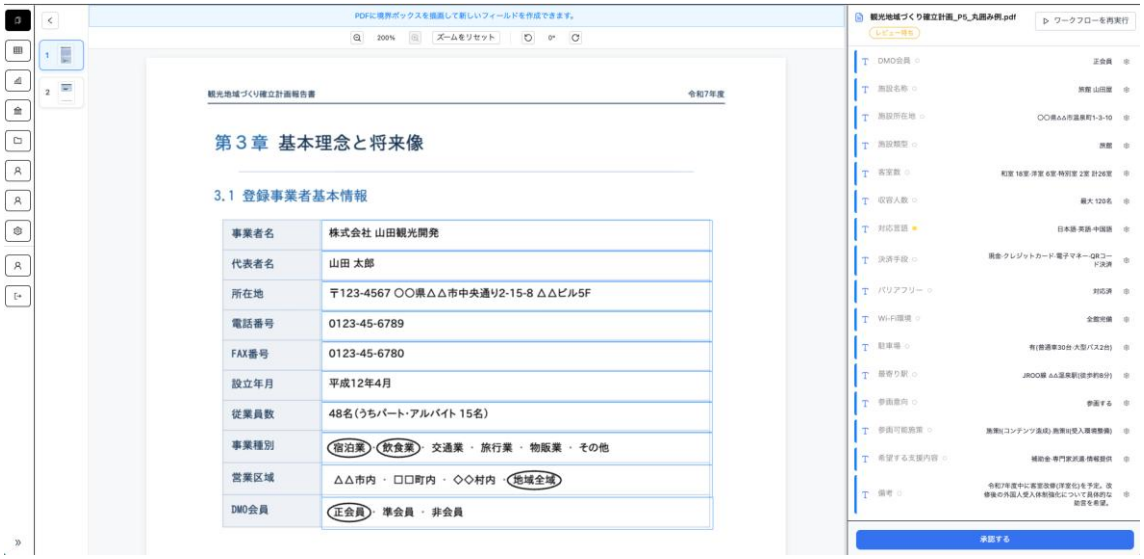


図 2.83 【UI006】構造化結果確認画面

本画面の概要

データ構造化処理の結果を確認する画面。各フィールドの抽出値・不正検知スコア・バウンディングボックス座標をハイライト表示で確認できる。

本画面から利用する機能

- 【FN008】データ構造化処理
- 【FN009】不要ページ除外
- 【FN010】不正検知
- 【FN011】バウンディングボックス抽出

- 【FN012】訂正線検知
- 【FN013】LLM 実行回数の自動分割
- 【FN014】LLM モデル自動選定
- 【FN015】LLM 実行の非同期・並列処理
- 【FN016】LLM プロンプト最適化

【UI007】データテーブル画面

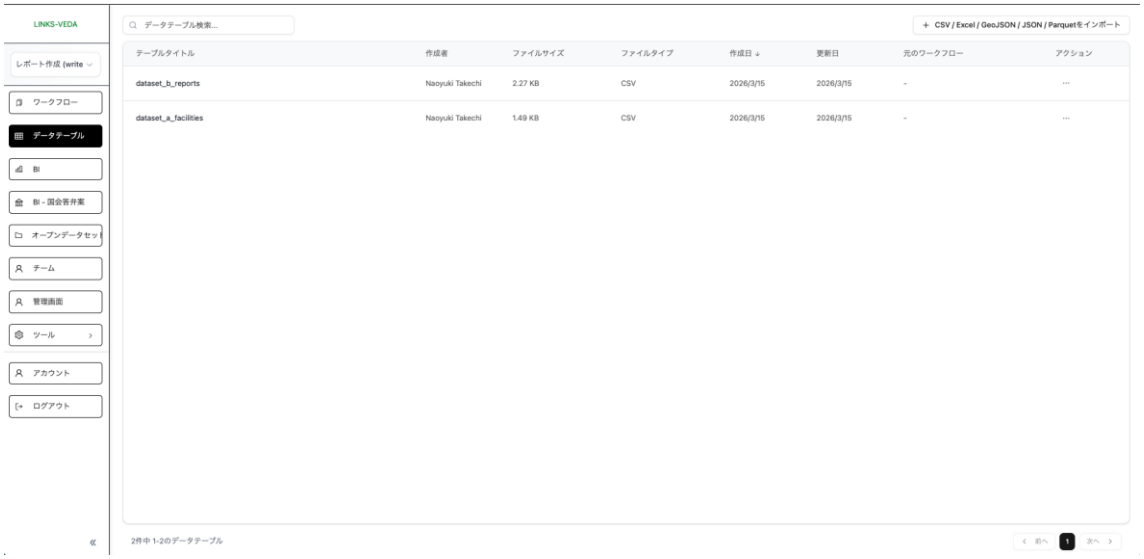


図 2.85 【UI007】データテーブル画面

本画面の概要

データテーブルの一覧・詳細確認・テーブルアクションの設定と実行・CSV エクスポートを行う画面。すべてのテーブルアクション機能が本画面から利用できる。

本画面から利用する機能

- 【FN003】データセット管理
- 【FN004】データテーブル管理
- 【FN017】テキスト正規化（全角/半角統一）

- 【FN018】文字カラム操作（テキスト抽出・検索と置換・カラム結合）
- 【FN019】住所文字列の正規化
- 【FN020】日付文字列の正規化・データ型変換
- 【FN021】カテゴリクレンジング
- 【FN022】四則演算で新しいカラムを作成
- 【FN023】数値階層化・数値置換
- 【FN024】ID 附番・マルチテーブル ID 付与
- 【FN025】重複行の削除
- 【FN026】テキスト結合
- 【FN027】ジオコーディング
- 【FN028】空間結合
- 【FN029】テキスト＋空間結合
- 【FN030】オープンデータ作成
- 【FN031】カラム名を翻訳する
- 【FN032】キーワード分類
- 【FN033】縦結合
- 【FN034】構造展開
- 【FN035】テーブル列操作
- 【FN036】外れ値フィルター
- 【FN038】テーブルヘッダー操作
- 【FN039】住所秘匿化
- 【FN040】数値偏差値化
- 【FN042】データエクスポート・ダウンロード

【UI008】データセット画面

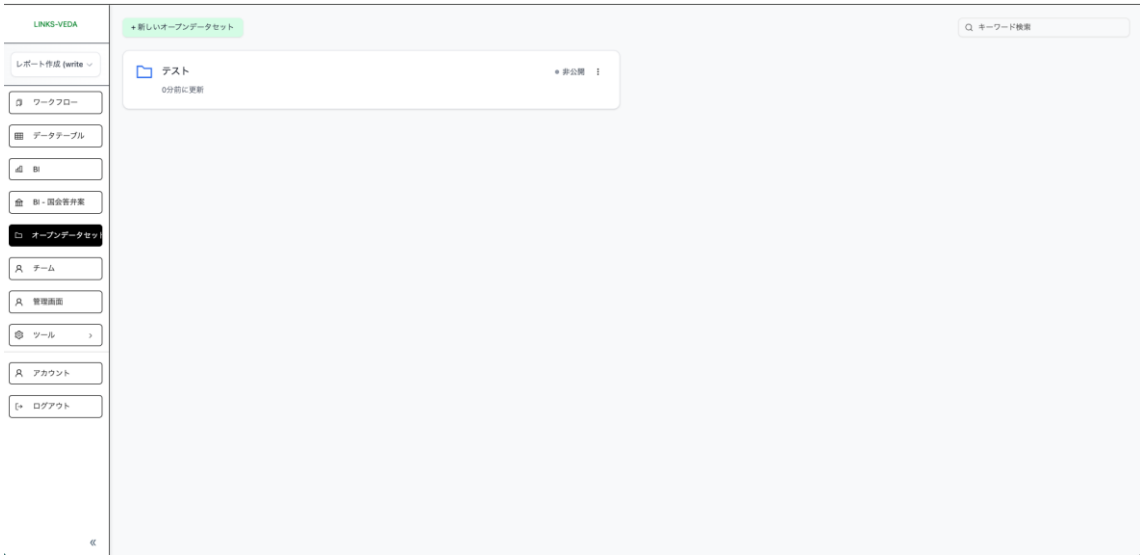


図 2.86 【UI008】データセット画面

本画面の概要

オープンデータ化の管理画面。データセットのメタデータ管理・G 空間情報センターとの連携・データ配信設定・エクスポートを行う。

本画面から利用する機能

- 【FN003】データセット管理
- 【FN041】メタ情報自動生成
- 【FN042】データエクスポート・ダウンロード

- 【FN043】オープンデータ化
- 【FN044】データ配信 API

【UI101】ダッシュボード画面

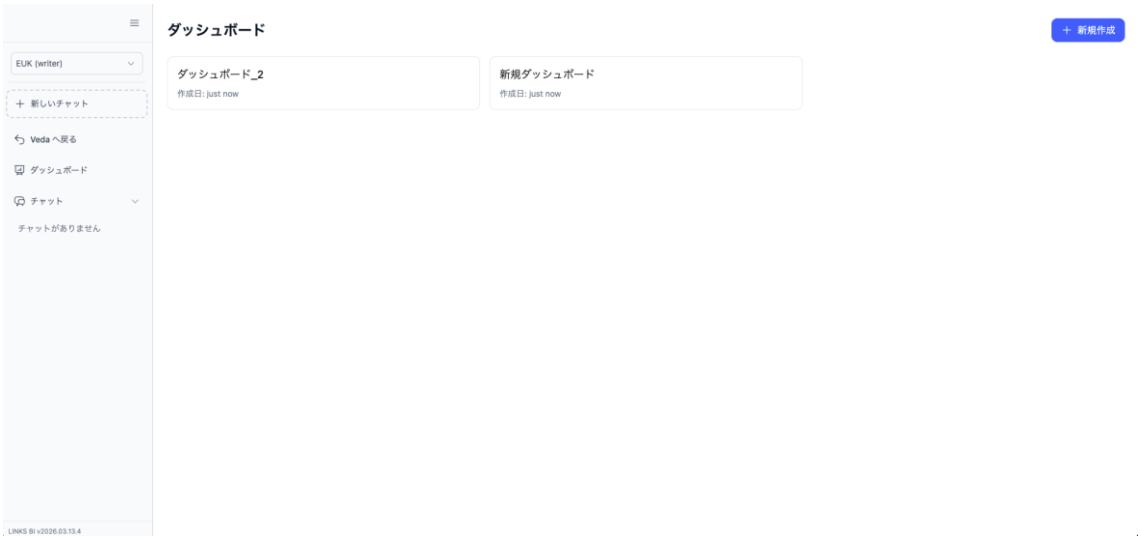


図 2.87 【UI101】ダッシュボード画面

本画面の概要

チームで作成したダッシュボードをカード形式で一覧表示する画面。カードには作成日時が表示され、クリックすることでダッシュボード編集画面に遷移する。本画面からダッシュボードの新規作成・削除を行える。

本画面から利用する機能

- 【FN104】ダッシュボード管理

【UI102】ダッシュボード編集画面

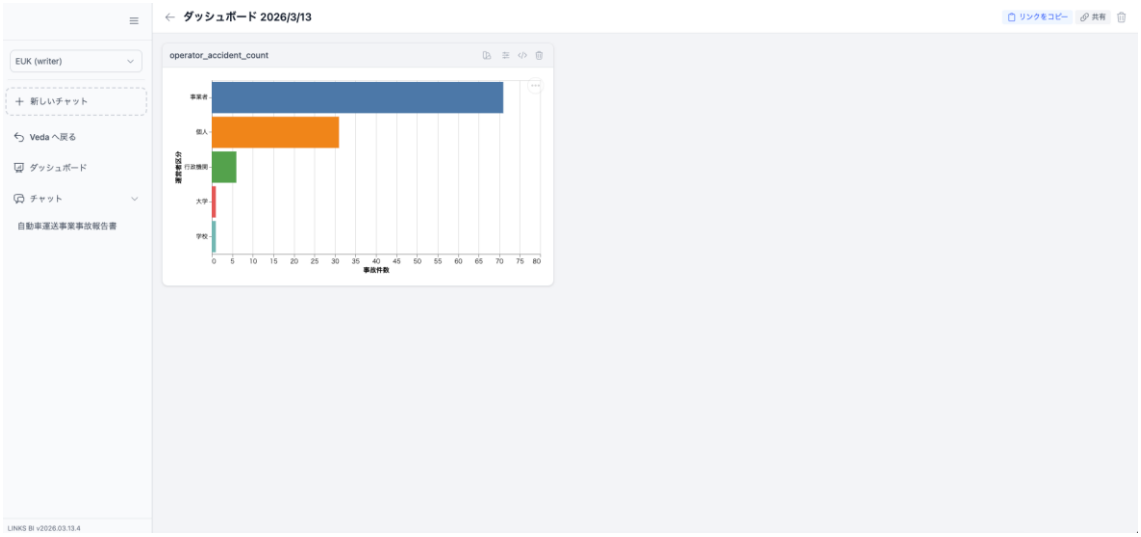


図 2.88 【UI102】ダッシュボード編集画面 1

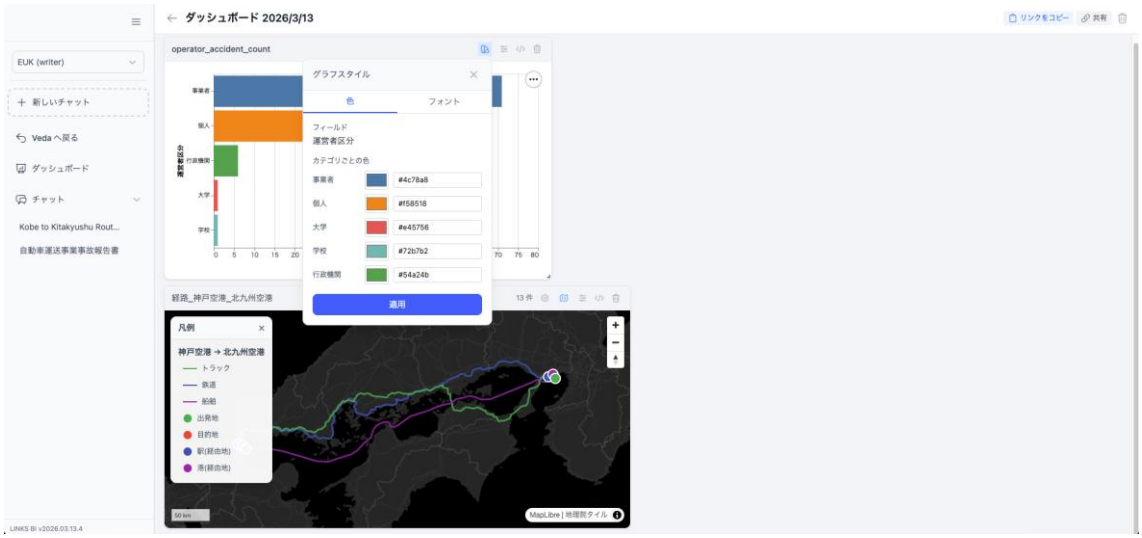


図 2.89 【UI102】ダッシュボード編集画面 2

本画面の概要

ダッシュボードを編集する画面。12 カラムのグリッドレイアウト上にグラフ・マップ・テーブルのウィジェットを自由に配置・リサイズでき、各ウィジェットのデータソース選択・グラフ設定・地図スタイル設定をノーコードで行える。ダッシュボードの共有設定・レポート出力も本画面から実施する。

本画面から利用する機能

- 【FN102】データインポート・データソース管理
- 【FN104】ダッシュボード管理
- 【FN105】グラフ可視化
- 【FN106】マップ可視化
- 【FN107】テーブルフィルタリング
- 【FN113】レポート出力
- 【FN114】ダッシュボード共有

【UI103】AI チャット画面

図 2.90 【UI103】AI チャット画面 1

図 2.91 【UI103】AI チャット画面 2

図 2.92 グラフ可視化の例

図 2.93 マップ可視化の例

図 2.94 テーブルフィルタリングの例

図 2.95 国会答弁機能の例

図 2.96 GIS 機能の例

本画面の概要

自然言語でデータの分析・可視化を行う AI チャット画面。画面はチャットパネルと可視化パネルの左右分割構成で、AI との対話を通じてテーブル・グラフ・マップをインタラクティブに生成・操作できる。グラフ可視化・マップ可視化・テーブルフィルタリング・統計分析・国会答弁・GIS 分析・ファイル検索など、AI と対話して行うすべての機能が本画面に紐づく。

本画面から利用する機能

- 【FN105】グラフ可視化
- 【FN106】マップ可視化
- 【FN107】テーブルフィルタリング
- 【FN108】演算モジュール
- 【FN109】AI チャット
- 【FN110】国会答弁
- 【FN111】統計分析処理
- 【FN112】外部データソース連携
- 【FN115】GIS 機能
- 【FN116】ファイル検索

3. 開発するシステム：非機能要件（NF）

本章に記載する非機能要件は、現在のプロトタイプ開発段階における水準として設定したものである。システムの本格運用への移行に際しては、可用性・性能・バックアップ頻度等の各要件をより高い水準に引き上げていくことを想定する。

3.1. 非機能要件一覧

本節では非機能要件一覧を示し、次節ではその詳細を示す。

表 3.1　非機能要件一覧

カテゴリ	ID	非機能項目	要件詳細
可用性	NF001	システムの連続稼働時間	平日営業日の 9:00〜18:00 において連続して稼働すること。営業時間外および休日も稼働するが、連続稼働時間の保証対象外とする。
可用性	NF002	安定動作時間	平日営業日の 9:00〜17:00 において、システムが安定的に動作することを保証する。
可用性	NF003	システム復旧時間	業務停止を伴う障害が発生した場合、24 時間以内の復旧を目指す。

可用性	NF004	データの保管期間	ユーザーがアップロードしたデータおよびデータベースに保存されたデータは、プロジェクト終了まで保持する。
性能・拡張性	NF005	データの読み込み速度	10MB 程度のファイルのアップロードを 10 秒以内に完了すること。1000 行程度のデータのインポートを 1 分以内に完了すること。
性能・拡張性	NF006	システムの処理実行速度	データ構造化処理を 3 分程度で完了すること。テーブルアクションは 1 万行のデータに対して 3 分以内に処理を完了すること。

運用・保守性	NF007	セキュリティ	国土交通省が定めるシステム開発におけるセキュリティガイドラインに準拠すること。ISMAP 対応のクラウドサービスを利用し、データの通信・保存先を日本国内のリージョンに留めること。ユーザーがアップロードした機密性の高い情報が AI モデルのトレーニングに利用されたり、外部プロバイダーに保存されないようにすること。
運用・保守性	NF008	認証	国土交通省内のユーザーのみがログインできるよう、ユーザー名・パスワードを利用したアカウントを発行し認証処理を行う。他のチームのデータにアクセスできないよう、チームを用いた認可処理を行う。
その他	NF009	クラウド環境の構築	本システムはクラウド環境で構築し、認証システムを備える。

その他	NF010	OSS 利用	本システムは可能な限り OSS（オープンソースソフトウェア）ライブラリ等を用いて構築する。
その他	NF011	OSS 提供	開発成果は可能な限り OSS として提供する。
その他	NF012	UI/UX の配慮	本システムの主たるユーザーは国土交通省職員であることに鑑み、簡素かつ明快な UI/UX を設計する。
その他	NF013	スケーラビリティと拡張性	本アプリケーションはプロトタイプ開発段階であり、100 人程度のユーザーが同時に接続・利用しても問題なくデータ構築モジュール・LINKS BI ともに利用できることを想定する。将来のデータ量の増加や新機能の追加にも対応できるように設計する。
その他	NF014	データのバックアップ	週次のバックアップを行う。

3.2. 非機能要件の詳細

以下に、非機能要件の詳細を記す。

【NF001】システムの連続稼働時間

- **本非機能要件の概要**
  - ■ 平日営業日の 9:00〜18:00 において連続して稼働すること。
  - ■ 営業時間外および休日も稼働するが、連続稼働時間の保証対象外とする。
- **設定理由**
  - ■ 国土交通省職員の業務時間帯（9:00〜18:00）にシステムが確実に利用できることを保証するための要件である。実証事業ではデータ構造化や BI 分析の操作を業務時間内に集中して行うため、この時間帯における連続稼働が業務遂行の前提条件となる。
  - ■ プロトタイプ開発段階であることから、営業時間外のメンテナンスやデプロイ作業を柔軟に実施できるよう、連続稼働時間の保証範囲を営業時間内に限定する。

【NF002】安定動作時間

- **本非機能要件の概要**
  - ■ 平日営業日の 9:00〜17:00 において、システムが安定的に動作することを保証する。
- **設定理由**

■ 国土交通省職員がデータ構造化処理の実行や BI 分析の操作を行うコア業務時間帯において、処理の中断やレスポンスの著しい低下が発生しないことを保証する必要がある。

■ データ構造化処理は 1 ファイルあたり数十秒〜数分の処理時間を要し、途中で中断すると再実行が必要となるため、安定動作の保証時間帯を明確に定めている。

【NF003】システム復旧時間

- **本非機能要件の概要**
  - ■ 業務停止を伴う障害が発生した場合、24 時間以内の復旧を目指す。
- **設定理由**
  - ■ 業務への影響を最小限に抑えるため、障害発生から翌営業日の業務開始までに復旧する体制を確保する・
  - ■ 本システムは Google Cloud Platform 上のマネージドサービス（Cloud Run、Cloud SQL 等）で構成されており、インフラ障害時にはクラウドプロバイダーの自動復旧機能を活用できる。アプリケーション障害についても、コンテナベースのデプロイにより迅速なロールバックが可能な構成としている。

【NF004】データの保管期間

- **本非機能要件の概要**
  - ユーザーがアップロードしたデータおよびデータベースに保存されたデータは、プロジェクト終了まで保持する。
- **設定理由**
  - 実証事業の期間を通じてデータを蓄積・活用するため、プロジェクト終了までのデータ保持が必要である・
  - ユーザーがアップロードした元データ、データ構造化処理の結果、BI 分析で作成したダッシュボード定義等はいずれもサーバー上のデータベースおよびストレージに保存されるため、実証期間中のデータ整合性を維持する必要がある。

【NF005】データの読み込み速度

- **本非機能要件の概要**
  - ファイルのアップロードは、10MB 程度のファイルを 10 秒以内に完了すること。
  - データのインポートは、1000 行程度のデータを 1 分以内に完了すること。
- **設定理由**
  - システム利用者がデータのアップロードおよびインポート操作において過度な待機時間を感じることなく作業を継続できるよう、ファイルサイズおよび行数に応じた処理速度の基準を設ける。
  - 行政文書のアップロードは業務フローの起点となる操作であり、この段階での遅延はその後のデータ構造化・分析作業全体のスループットに影響する。

【NF006】システムの処理実行速度

- **本非機能要件の概要**
  - ■ データ構造化処理については、3 分程度の処理時間を目指す。
  - ■ テーブルアクションの実行については、1 万行のデータに対する処理を 3 分以内に完了することを目指す。
- **設定理由**
  - ■ データ構造化処理は OCR および LLM を組み合わせた多段処理であり、1 ファイルあたりの処理時間はページ数・文書構造の複雑さに依存する。ユーザーがワークフロー画面上で処理完了を待機する利用形態を想定し、3 分程度を目標とする。
  - ■ テーブルアクションは構造化済みデータに対する加工・集計操作であり、ユーザーが対話的に操作する機能である。1 万行規模のデータに対して実用的な応答速度を確保するため、3 分以内の処理完了を目標とする。

【NF007】セキュリティ

- **本非機能要件の概要**
  - ■ 国土交通省が定めるシステム開発におけるセキュリティガイドラインに準拠すること。
  - ■ ISMAP 対応のクラウドサービスを利用し、データの通信・保存先を日本国内のリージョンに留めること。
  - ■ ユーザーがアップロードした機密性の高い情報が AI モデルのトレーニングに利用されたり、外部プロバイダーに保存されないようにすること。

■ 政府統一基準群および国土交通省セキュリティポリシーに準拠し、アプリケーション・OS・ミドルウェア・ネットワーク・物理・運用の各セキュリティを設計する。

● **設定理由**

■ 国土交通省向けシステムとして、政府情報システムのためのセキュリティ評価制度（ISMAP）に準拠したクラウドサービスの利用が求められる。本システムでは ISMAP 対応の Google Cloud Platform を利用し、データの保存・通信を日本国内リージョン（asia-northeast1 等）に限定する。

■ 本システムはデータ構造化処理および AI チャット機能において LLM（大規模言語モデル）を使用する。機密性の高い行政文書を処理するため、ユーザーがアップロードしたデータが AI モデルのトレーニングデータとして利用されない契約・設定を確保する必要がある。

■ 通信経路の TLS 暗号化、保存データの暗号化、アクセスログの記録等、多層的なセキュリティ対策を実施する。

【NF008】認証

- **本非機能要件の概要**
  - ■ 国土交通省内のユーザーのみがログインできるよう、ユーザー名・パスワードを利用したアカウントを発行し認証処理を行う。
  - ■ 他のチームのデータにアクセスできないよう、チームを用いた認可処理を行う。
- **設定理由**
  - ■ 本システムは国土交通省の実証事業専用であり、利用ユーザーを省内の関係者に限定する必要がある。アカウント発行制による認証を行い、不正アクセスを防止する。
  - ■ 実証事業では複数の部署・チームがそれぞれ異なるデータセットを扱う。チーム単位のアクセス制御により、各チームが自チームのデータのみを閲覧・操作できるようデータ分離を実現する。

【NF009】クラウド環境の構築

- **本非機能要件の概要**
  - ■ 本システムはクラウド環境で構築し、認証システムを備える。
- **設定理由**
  - ■ クラウド環境を利用することで、オンプレミス環境と比較してインフラの構築・運用にかかる負荷を軽減し、必要に応じたリソースの増減を柔軟に行うことができる。
  - ■ プロトタイプ開発段階では機能の追加・変更が頻繁に発生するため、コンテナベースのデプロイ（Cloud Run）やマネージドデータベース（Cloud SQL）等のマネージドサービスを活用し、開発・デプロイサイクルの迅速化を図る。

【NF010】OSS 利用

- **本非機能要件の概要**
  - ■ 本システムは可能な限り OSS（オープンソースソフトウェア）ライブラリ等を用いて構築する。
- **設定理由**
  - ■ 開発コストの低減と技術の透明性確保のため、OSS を積極的に活用する。特定ベンダーの製品に依存しない構成とすることで、将来的な技術変更やベンダー切り替えに対する柔軟性を確保する。
  - ■ 政府情報システムにおいては、利用技術の透明性が重要視される。OSS はソースコードが公開されており、セキュリティ監査や第三者検証が可能である。

【NF011】OSS 提供

- **本非機能要件の概要**
  - ■ 開発成果は可能な限り OSS として提供する。
- **設定理由**
  - ■ 政府情報システムの透明性確保および、開発で得られた技術知見の社会還元のために、成果物を OSS として公開する。
  - ■ 本システムの開発成果を公開することで、他省庁や自治体が類似のデータ構造化・分析基盤を構築する際に活用できるようにし、行政全体の DX 推進に貢献する。

【NF012】UI/UX の配慮

- **本非機能要件の概要**
  - ■ 本システムの主たるユーザーは国土交通省職員であることに鑑み、簡素かつ明快な UI/UX を設計する。
- **設定理由**
  - ■ 本システムの利用者は、データ構造化や BI 分析の専門知識を必ずしも有しない行政職員である。専門的な操作手順を必要としない直感的なインターフェースにより、業務への導入障壁を低減する必要がある。
  - ■ データ構造化処理の実行・結果確認、BI ダッシュボードの操作・AI チャットによる分析指示といった主要操作を、最小限の手順で完了できる設計とする。

【NF013】スケーラビリティと拡張性

- **本非機能要件の概要**
  - ■ 本アプリケーションはプロトタイプ開発段階であり、100 人程度のユーザーが同時に接続・利用しても問題なくデータ構築モジュール・LINKS BI ともに利用できることを想定する。
  - ■ 将来のデータ量の増加や新機能の追加に対応できるように設計する。
  - ■ データ構築モジュールのページ表示については、100 人の同時接続時においてp95（95 パーセンタイル、全リクエストの 95%がこの値以下に収まる応答時間）レスポンス時間 5 秒未満、エラー率 10%未満を基準とする。負荷テストにより、ワークフローページでは p95 レスポンス時間 1.26 秒、平均レスポンス時間 667ms、エラー率 0%が確認されている。
  - ■ データ構築モジュールのデータ構造化処理については、Cloud Run によるファイル単位の並列処理により、投入ファイル数に応じたスケールアウトに対応する。2,500 ファイルの同時処理で約 15 分（1 ファイルあたり約 0.36 秒）の処理性能を確保している。
  - ■ LINKS BI の AI チャット機能については、100 人の同時接続においても概ね安定して動作することを想定する。負荷テストでは 100 人同時接続で約 95%以上の成功率が確認されており、失敗はネットワーク層のレート制限に起因するものであり、アプリケーション基盤に障害は発生していない。
- **設定理由**
  - ■ 国土交通省の実証事業では複数部署が同時にシステムを利用するため、同時接続時の安定動作は必須要件である。データ構築モジュール・LINKS BI ともに

100 人の同時接続を基準として負荷テストを実施し、要件を満たすことを確認している

■ データ構造化処理では、実証事業において数千ファイル規模の一括処理が求められる。ファイル単位の並列処理設計により、数千件/時のスループットを確保する

■ プロトタイプから本格運用への移行を見据え、拡張可能なアーキテクチャとする。

【NF014】データのバックアップ

- **本非機能要件の概要**
  - ■ 週次のバックアップを行う。
- **設定理由**
  - ■ 行政データの喪失は業務に深刻な影響を与えるため、確実なバックアップ体制が必要である。
  - ■ 週次バックアップにより、障害発生時のデータ損失を最大 1 週間分に抑制する。

4. 実証調査に利用するデータ（DT）

4.1. 実証調査に利用するデータ一覧

表 4.1　実証調査に利用するデータ一覧

ID	データ名称	データ形式	出所	データを利用する ID	入手状況
DT001	行政文書（PDF 形式）	PDF	国土交通省各部局（提供）	【FN005】 【FN006】 【FN008】	入手済
DT002	行政文書（Excel / CSV 形式）	Excel（.xlsx）/ CSV	国土交通省各部局（提供）	【FN005】 【FN008】	入手済
DT003	行政文書（Word / PowerPoint 形式）	Word（.docx）/ PowerPoint（.pptx）	国土交通省各部局（提供）	【FN005】 【FN008】	入手済

DT004	地理空間データ（GeoJSON）	GeoJSON	国土地理院・各自治体等（オープンデータ）	【FN003】 【FN015】 【FN106】	入手済
DT005	政府統計データ（e-Stat）	JSON / CSV（API）	政府統計総合窓口（e-Stat）API	【FN110】	入手済（API利用）
DT006	「第2次交通政策基本計画」に基づく新たなモビリティサービスに関する基礎調査	Excel	国土交通省総合政策局モビリティサービス推進課（提供）	【FN003】 【FN004】 【FN005】 【FN006】	入手済
DT007	OpenStreetMap 道路データ	GeoJSON	OpenStreetMap	【FN115】	入手済

4.2. 実証調査に利用するデータの詳細

実証調査に利用するデータの詳細を記す。

【DT001】行政文書（PDF 形式）

- **本データの概要**
  - 国土交通省各部局が保有する PDF 形式の行政文書。様式・原票・報告書・申請書等の多様な文書種別を含む。アセット管理機能（【FN005】）でアップロードされ、ワークフロー管理（【FN006】）を通じてデータ構造化処理（【FN008】）の主要な入力データとなる。
- **サンプル・イメージ**

図 4.1 行政文書（PDF 形式）のサンプル

- **本データの形式**
  - PDF（.pdf）
- **出所**
  - 国土交通省担当部局より提供を受ける。個人情報・機密情報を含む場合は適切な匿名化・マスキング処理を施す。

【DT002】行政文書（Excel / CSV 形式）

- **本データの概要**
  - 国土交通省各部局が保有する Excel（.xlsx）または CSV 形式の行政文書。集計表・一覧表・統計データ等を含む。データ構築モジュールのアセット管理（【FN005】）でアップロードされ、DuckDB によるインポート処理でデータテーブルに変換される。
- **サンプル・イメージ**

図 4.2 行政文書（Excel / CSV 形式）のサンプル

● **本データの形式**

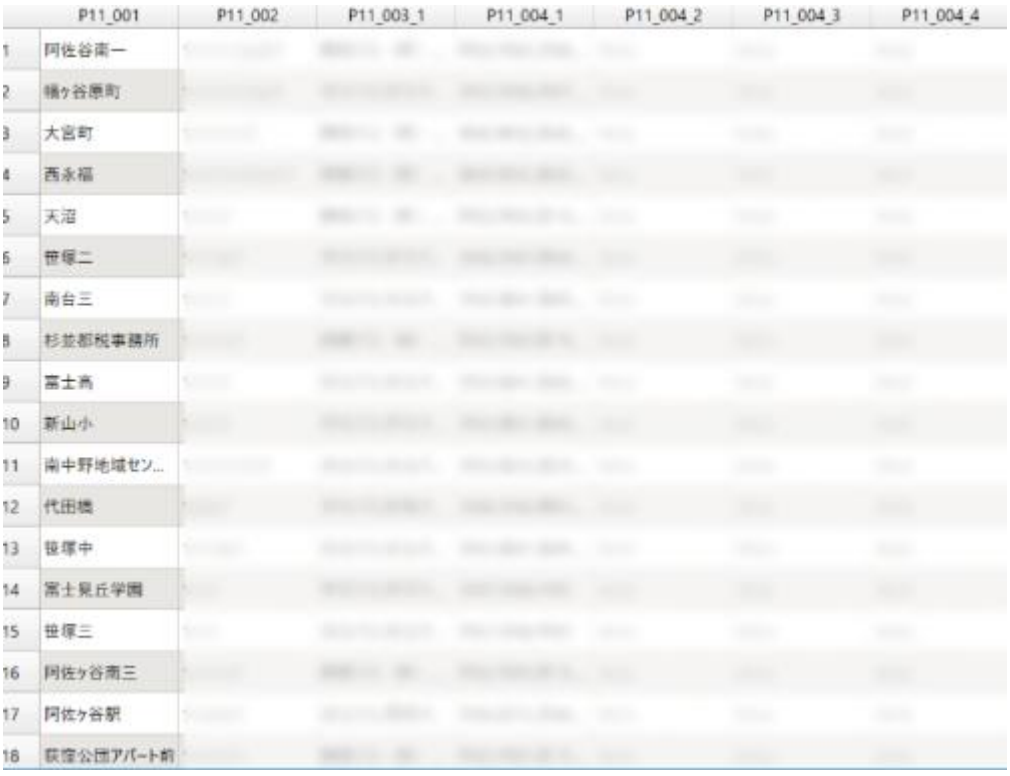
■ Excel（.xlsx）/ CSV（UTF-8 または Shift_JIS 対応）

● **出所**

■ 国土交通省担当部局より提供を受ける。

【DT003】行政文書（Word / PowerPoint 形式）

- **本データの概要**
  - ■ Word（.docx）または PowerPoint（.pptx）形式の行政文書。報告書・資料・提案書等を含む。システム内で PDF 変換（【IF009】Office→PDF 変換 API）後にデータ構造化処理（【FN008】）の対象となる。
- **サンプル・イメージ**



	P11_001	P11_002	P11_003_1	P11_004_1	P11_004_2	P11_004_3	P11_004_4
1	阿佐谷南一						
2	桃ヶ谷原町						
3	大宮町						
4	西永福						
5	天沼						
6	笹塚二						
7	南台三						
8	杉並都税事務所						
9	富士高						
10	新山小						
11	南中野地域セン…						
12	代田橋						
13	笹塚中						
14	富士見丘学園						
15	笹塚三						
16	阿佐ヶ谷南三						
17	阿佐ヶ谷駅						
18	荻窪公団アパート前						

図 4.3 行政文書（Word / PowerPoint 形式）のサンプル

- **本データの形式**
  - Word（.docx）/ PowerPoint（.pptx）。PDF 変換後に構造化処理対象となる。
- **出所**
  - 国土交通省担当部局より提供を受ける。

【DT004】地理空間データ（GeoJSON）

- **本データの概要**
  - ■ 地図上での可視化（【FN106】）・空間処理（【FN015】）に使用する地理空間データ。国土地理院や自治体が公開するオープンデータ、または利用者がアップロードするデータを含む。
- **本データの形式**
  - ■ GeoJSON
- **出所**
  - ■ 国土地理院（https://www.gsi.go.jp/）：CC BY 4.0 等のオープンライセンス
  - ■ e-Stat（https://www.e-stat.go.jp/）：政府統計の総合窓口。
  - ■ 利用者アップロード：システムが受け付けるフォーマット（GeoJSON）で提供。
- **データ定義**

表 4.2　地理空間データ（GeoJSON）のカラム定義

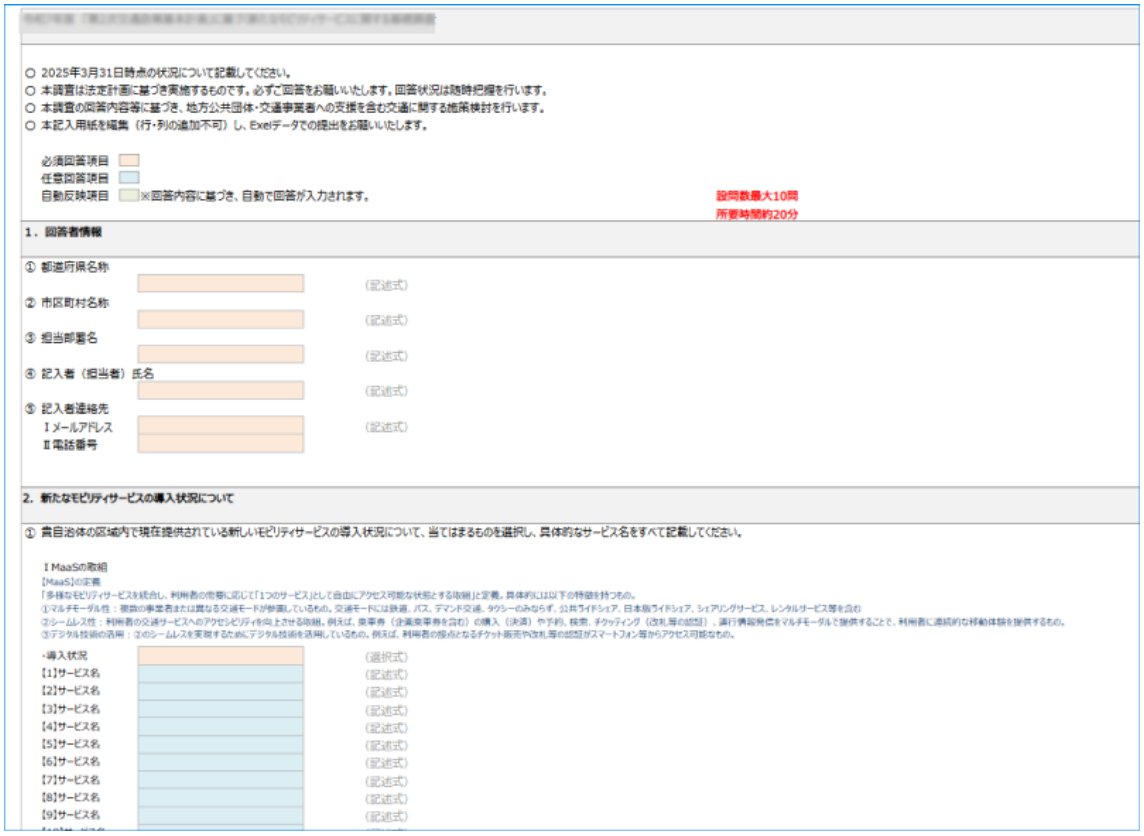
No.	日本語名称	項目長	必須	書式・選択肢など
1	type	—	○	"FeatureCollection"
2	features	—	○	Feature オブジェクトの配列
3	geometry	—	○	Point / LineString / Polygon 等の GeoJSON Geometry
4	properties	—	×	任意の属性情報（JSON オブジェクト）

【DT005】政府統計データ（e-Stat）

- **本データの概要**
  - ■ 政府統計の総合窓口（e-Stat）が提供する統計データ。LINKS BI の e-Stat 連携機能（【FN110】）でチャット経由で検索・取得してダッシュボードに取り込む。
- **本データの形式**
  - ■ JSON / CSV（e-Stat API レスポンス形式）。統計表の列定義は統計ごとに異なる。
- **出所**
  - ■ e-Stat API（https://www.e-stat.go.jp/api/）を使用。API キーの発行が必要。利用規約に基づいて利用する。

【DT006】「第 2 次交通政策基本計画」に基づく新たなモビリティサービスに関する基礎調査（Excel 形式）

- **本データの概要**
  - 全国の自治体に対して、モビリティサービスに関する導入状況を調査するためのアンケート資料。アセット管理機能（【FN005】）でアップロードされ、ワークフロー管理（【FN006】）を通じてデータ構造化処理（【FN008】）の入力データとなる。ユースケース実証に加え、省内研修会などの実証において活用された。
- **サンプル・イメージ**



○ 2025年3月31日時点の状況について記載してください。
○ 本調査は法定計画に基づき実施するものです。必ずご回答をお願いいたします。回答状況は随時把握を行います。
○ 本調査の回答内容等に基づき、地方公共団体・交通事業者への支援を含む交通に関する施策検討を行います。
○ 本記入用紙を編集（行・列の追加不可）し、Excelデータでの提出をお願いいたします。

必須回答項目
任意回答項目
自動反映項目　※回答内容に基づき、自動で回答が入力されます。

設問数最大10問
所要時間約20分

1．回答者情報

① 都道府県名称　（記述式）
② 市区町村名称　（記述式）
③ 担当部署名　（記述式）
④ 記入者（担当者）氏名　（記述式）
⑤ 記入者連絡先
Ⅰメールアドレス　（記述式）
Ⅱ電話番号

2．新たなモビリティサービスの導入状況について

① 貴自治体の区域内で現在提供されている新しいモビリティサービスの導入状況について、当てはまるものを選択し、具体的なサービス名をすべて記載してください。

Ⅰ MaaSの取組
[MaaS]の定義
「多様なモビリティサービスを統合し、利用者の需要に応じて「1つのサービス」として自由にアクセス可能な状態とする取組」と定義。具体的には以下の特徴を持つもの。
①マルチモーダル性：複数の事業者または異なる交通モードが参画しているもの。交通モードには鉄道、バス、デマンド交通、タクシーのみならず、公共ライドシェア、日本版ライドシェア、シェアリングサービス、レンタルサービス等を含む
②シームレス性：利用者の交通サービスへのアクセシビリティを向上させる取組。例えば、乗車券（企画乗車券を含む）の購入（決済）や予約、検索、チケッティング（改札等の認証）、運行情報発信をマルチモーダルで提供することで、利用者に連続的な移動体験を提供するもの。
③デジタル技術の活用：②のシームレスを実現するために、デジタル技術を活用しているもの。例えば、利用者の接点となるチケット販売や改札等の認証がスマートフォン等からアクセス可能なもの。

・導入状況　（選択式）
[1]サービス名　（記述式）
[2]サービス名　（記述式）
[3]サービス名　（記述式）
[4]サービス名　（記述式）
[5]サービス名　（記述式）
[6]サービス名　（記述式）
[7]サービス名　（記述式）
[8]サービス名　（記述式）
[9]サービス名　（記述式）

図 4.5 モビリティサービス基礎調査（Excel 形式）のサンプル

- **本データの形式**
  - Excel（.xlsx）
- **出所**
  - 国土交通省総合政策局モビリティサービス推進課より提供を受ける。

【DT007】OpenStreetMap 道路データ（GeoJSON 形式）

- **本データの概要**
  - ■ 【FN115】GIS 機能において道路ジオメトリの可視化および経路探索を目的に活用される GeoJSON 形式のデータ。オープンデータとしてダウンロード後に、経路探索に活用しやすいようノード調整を手動で施したデータを活用している。
- **サンプル・イメージ**
  - ■ （実証調査後に追記予定）
- **本データの形式**
  - ■ GeoJSON（PBF 形式から変換）
- **出所**
  - ■ Geofabrik（https://www.geofabrik.de/）よりダウンロードした OSM データを利用。利用規約に基づいて利用する。
- **データ定義**

表 4.3　OpenStreetMap 道路データのカラム定義

No.	日本語名称	項目長	必須	書式・選択肢など
1	gid	—	○	—
2	source	—	○	—
3	target	—	○	—
4	geom geometry	—	○	—

5	highway	—	○	—
6	name	—	○	—
7	oneway	—	○	—
8	reverse_cost	—	×	—
9	length_m	—	×	—
10	blocked	—	○	—
11	maxspeed_forward	—	×	—

5. 用語集

以下に本システム設計書で使用する主な用語を定義する。本用語集はアプリケーション内で出てくるドメイン用語・プロジェクト固有の概念を対象とする。技術用語・OSS ライブラリ・ライセンス等の詳細は第 2 章のソフトウェア・ライブラリ（SL）セクション等に記載する。

プロジェクト・システム名称

用語	定義・説明

LINKS Veda	国土交通省が推進する行政情報の構造化・利活用基盤整備プロジェクト。非構造行政データを機械処理可能なデータとして再構築し、オープンデータとして公開する取り組み。
データ構築モジュール	Veda の非構造データ構造化基盤。PDF・Excel・Word 等の非構造行政データを AI を活用して構造化し、管理・配信するモジュール。
データ活用モジュール（LINKS BI）	Veda のデータ可視化・分析基盤。データ構築モジュールで構造化されたデータをグラフ・マップ・AI チャットで可視化・分析するモジュール。
EBPM	Evidence-Based Policy Making（エビデンスに基づく政策立案）。統計データや実証結果に基づいて政策を立案するアプローチ。Project LINKS は EBPM を推進するためのデータ整備基盤として位置づけられる。

データ管理に関する用語

用語	定義・説明
アセット	LINKS Veda に登録・管理される各種ファイルの総称。PDF・Excel・CSV・JSON・GeoJSON・Word・PowerPoint 等が対象。
データテーブル	ワークフロー機能によって構造化されたデータ、またはCSV 等のインポートで登録されたテーブル形式のデータ。
データセット	複数のデータテーブルを束ねてオープンデータ化向けの管理単位としたもの。G 空間情報センターへの公開単位であり、LINKS BI での分析対象となる。
スキーマ	構造化データのカラム名・データ型・抽出条件等を定義したメタ情報。「構造化スキーマ」と表記する場合もある。
非構造データ	定型スキーマを持たないデータの総称。行政文書ではPDF・Word・Excel・画像等が該当する。
Parquet	Apache Parquet。列指向のバイナリファイル形式。LINKS Veda ではデータテーブルの内部保存形式およびLINKS BI へのデータ配信形式として使用する。

パネルデータ	同一観測単位を時系列で追跡したデータ構造。テーブルアクションにより指定粒度で生成し、欠損時点を自動補完する。

ワークフロー・構造化処理に関する用語

用語	定義・説明
ワークフロー	特定の文書をデータ構造化するための処理単位。様式や文書の種類ごとに 1 つのワークフローを作成し、スキーマ定義・構造化設定・実行履歴を管理する。
ワークフロー実行（WorkflowRun）	ワークフローの 1 回の実行履歴。structure（処理中）・completed（完了）・structureError（エラー）のステータスで管理される。
データ構造化処理	OCR や LLM を用いて非構造データから意味情報を抽出し、定義されたスキーマに従って構造化データを生成する処理。
スキーマ自動提案（スキーマサジェッション）	LLM がアセットの内容を解析し、データ構造化に適したスキーマ（カラム名・データ型・説明文）を自動提案する機能。
テーブルアクション	データテーブルに対してノーコードで実行できるデータ加工・変換処理の総称。テキスト正規化・ジオコーディング・空間結合・ID 附番等を含む。
信頼度（Confidence）	データ構造化処理で各フィールドの抽出精度を示す 0.0〜1.0 のスコア。

バウンディングボックス	文書内における各フィールド値の位置座標。構造化結果の根拠箇所を UI 上でハイライト表示するために使用する。
打ち消し線（訂正線）	帳票上の値を無効化・修正するために引かれる線の総称。打ち消し線が検出された場合、VLM を利用して訂正後の値を抽出する。
不要ページ除去	データ構造化の際、表紙・奥付・目次等の構造化対象外ページを自動的に無視する機能。

チーム・認証に関する用語

用語	定義・説明
チーム	複数ユーザーがデータ登録・管理を行うための作業環境。マルチテナントの単位。すべてのリソースはチームに紐付いて管理される。
ロール	チームメンバーに割り当てられる権限。編集者・閲覧者の 2 段階。管理者（情報政策課の職員）はシステム全体のアカウント管理・メンバー管理を行う。

LINKS BI に関する用語

用語	定義・説明
ダッシュボード	グリッドレイアウトでウィジェット（グラフ・マップ・テキスト・テーブル）を配置した分析画面。
ウィジェット	ダッシュボード上に配置される個々の可視化要素。グラフ・マップ・テーブル・テキスト等の種類がある。
AI チャット	LINKS BI が提供する自然言語によるデータ分析インターフェース。ユーザーが日本語で質問すると、AI がデータ分析・グラフ生成・地図表示等を自動実行する。

データソース	LINKS BI でインポートされた分析対象のデータ。LINKS Veda のデータテーブルを選択してインポートする。
DuckDB-Wasm	DuckDB のブラウザ内実行版。LINKS BI ではクライアントサイドで SQL クエリを実行し、データのプレビュー・フィルタリング・演算処理等に使用する。
ハイブリッド検索	ベクトル検索（意味的類似性による検索）とキーワード検索（テキスト一致による検索）を統合し、RRF スコアで結果をランク付けする検索手法。ドキュメント検索機能で使用する。

オープンデータ・外部連携に関する用語

用語	定義・説明
オープンデータ化	データセットを G 空間情報センター等の外部プラットフォームに公開する一連の処理。メタデータの整備・公開ステータスの管理を含む。
G 空間情報センター	地理空間情報を収集・提供する国土交通省関連のデータポータル。LINKS Veda からのオープンデータ公開先。
e-Stat	政府統計の総合窓口。LINKS BI から API 連携でデータを取得する外部データソース。
ジオコーディング	住所テキストを緯度経度座標に変換する処理。テーブルアクションの一つとして提供される。
メッシュコード	JIS X 0410 に基づく地域メッシュの識別コード。LINKS BI のマップ可視化でグリッド表示に使用する。

ユースケース・実証に関する用語

用語	定義・説明
ユースケース（UC）	Project LINKS における実証実験の単位。各省庁・部局が持つ具体的なデータ活用シナリオ。UC11〜UC34 等の番号で管理される。

記載上の凡例

表記	意味
朱文字	新規開発・既存改修の機能・項目。
【FNXXX】	システム機能 ID（FN001〜）
【SLXXX】	ソフトウェア・ライブラリ ID（SL001〜）
【ALXXX】	数理モデル・アルゴリズム ID（AL001〜）
【COXXX】	システムコンポーネント ID（CO001〜）
【HWXXX】	ハードウェア ID（HW001〜）
【IFXXX】	データインターフェース ID（IF001〜）
【UIXXX】	ユーザーインターフェース（画面）ID（UI001〜）
【DTXXX】	実証調査データ ID（DT001〜）

【NFXXX】	非機能要件 ID（NF001〜）
「^x.y」	同一メジャーバージョン内での互換更新を許容するバージョン指定。例: 「^2.15」 は 「2.x」 系の更新を許容する。
「>=x.y」	指定したバージョン以上を許容するバージョン指定。例: 「>=1.40」 は 「1.40」 以上の任意の版を含む。
「a / b」	複数ライブラリや複数パッケージを併記している表記。例: 「vega ^6.2 / vega-lite ^6.4」 はそれぞれ個別のバージョンを示す。

LINKS Veda(V2) システム設計書

2026 年 3 月 発行

委託者：国土交通省 総合政策局 情報政策課

受託者：マイクロベース・ユーカリヤ