{::options parse_block_html="true" /}。

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

GitLabデータカタログインデックス

データカタログのページでは、アナリティクスソリューション、ダッシュボード、ワークフロー、キーワードなどのインデックスを掲載しています。リンクやリソースを追加したい場合は、ご自由にご投稿ください。

Sisense ダッシュボード

Sisenseは、当社のエンタープライズスタンダードなBIツールです。

重要：2021-09-08 SAFEダッシュボードの変更点

2021-09-08に、GitLabの成熟とSAFEデータフレームワークの推進により、Sisenseダッシュボードのアクセスが変更されます。次のセクションでは、予定されているアクセスプロセスについて説明します。

Sisenseダッシュボードへのアクセスは、職務上の役割に基づいており、SAFEデータアクセスフレームワークによって管理されています。Sisenseでは、ダッシュボードは以下のスペースに分類されます。

- **SAFEダッシュボード**スペースには、GitLabのSAFE基準を満たすすべてのダッシュボードが置かれています。SAFEダッシュボードは、役割や必要性に応じてGitLabチームメンバーが利用できます。
- **GitLab**スペースは、SAFEの取り扱いを必要としないすべてのダッシュボードを収容する一般アクセスエリアです。一般アクセス用ダッシュボードは、GitLabチームメンバー全員がアクセスできます。

SAFEダッシュボードを含むダッシュボードの全リストは、GitLabDashboardIndexに掲載されています。

GitLab General Access Dashboardへのアクセス

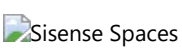
一般アクセスダッシュボードは、すべてのGitLabチームメンバーが利用でき、アクセスリクエストの発行は必要ありません。アクセス方法は、Sisenseの概要と開発ページに記載されています。

SAFEダッシュボードへのアクセス

すべてのSAFEダッシュボードは、Sisense SAFE Dashboardスペース内に保存され、アクセスはSisenseスペースレベルで許可されます。1つのSAFEダッシュボードにアクセスすると、すべてのSAFEダッシュボードにアクセスできます。SAFEダッシュボード（およびスペース）へのアクセスには、以下が必要です。

1. 直属の上司の承認
2. 部署のVP（またはそれに相当するもの）の承認
3. GitLab Dashboard Indexで定義されたSAFE Dashboard Space Ownerの承認 SAFEダッシュボードへのアクセス権の取得

1. アクセスリクエストを作成し、GitLabDashboardIndexからアクセスを必要とするSAFEダッシュボードを3つまでリストアップする。このSAFEダッシュボードのリストは、承認者があなたのニーズと意図を理解するのに役立ちます。
2. 直属の上司、所属部署の副社長（またはそれに相当する人物）、およびGitLabダッシュボードのインデックスヘッダーに定義されているSAFEスペースオーナーに承認を求める。SnowflakeでSAFEデータへのアクセスが60日以内に承認されていれば、承認は**必要ない。**このステップをスキップして、特定のARにリンクする。
3. リクエストが承認されたら、@gitlab-data/analystsをタグ付けすると、データチームがリクエストを処理します。
4. 処理が完了すると、Sisenseにログインして、ご希望のSAFEダッシュボードやSAFEダッシュボードスペース内の他のすべてのダッシュボードにアクセスできるようになります。


Sisense Spaces

SnowflakeでのSAFEデータへのアクセス方法については、「SnowflakeでのSAFEデータへのアクセス」をご覧ください。

SAFEデータのGoogle Sheetsファイルは、「SAFEダッシュボードへのアクセス」の手順でアクセスできます。Google SheetsのSAFEデータにアクセスするには、この手順に従ってください。

2021-09-08終了 SAFEダッシュボード変更のお知らせ

分野別データ

マーケティング

ダッシュボード

- TD：マーケティングデータマート
- TD：SDRパフォーマンスダッシュボード

分析プロジェクト

- 2021-10-01 SAO分析

売上高

ダッシュボード

- TD：カスタマーセグメンテー
- ション TD：セールスファネル
- 手動使用データアップロードプロセス

分析プロジェクト

- 買い方の傾向

ファイナンス

- TD：ファイナンスARR

製品

ダッシュボード

- TD：プロダクト・ジオロ
- ケーション TD：プライ
  シング・アナリシス

データモデルとプロセス

- 製品の使用状況デー
- タ SaaSサービスの
- Ping SaaS製品のイベ
  ント

分析プロジェクト

- 2020-12 プロダクトアナリティクス オフサイト

成長

ダッシュボード

- XMAU分析成長ダッ
- シュボード

分析プロジェクト

- 2021-08CustomerCentricProductInsights
- 2021-08ExperimentationWorkshop
- 2021-08Stage:Secureの採用と転換の分析 2021-08名前空
- 間の転換の分析
- FY22-Q1 Growth Team KPI Review
- 2021-08SSOログインディープダイブ
- 分析 Growth Insights
- 締めくくりの成長実験

カスタマーサクセス

- 建設中

エンジニアリング

- MRレート

人々

- People Metrics Overview
- PTO By Roots (Slack)
- People Key Metrics
- People KPI Deck
- PromotionsReport
- タレント・アクイジション・メトリクス
- People Metrics - Sisense Dashboardでのデータ発見

データチーム

- Sisenseの利用と導入 信頼でき
- るデータヘルス

指標と用語の索引

- セールス用語集
- ARR：Annual Recurring Revenue
- ATR：Available To Renewal
- CAC: Customer Aquisition Cost
- LTV: *Customer* LifeTime Value
- Namespace
- NDR: Net Dollar Retention
- PQL:Product Qualified Lead 製
- 品ステージ
- SM: Self-Managed aka Self-Hosted
- UPA:究極の親アカウント xMAU: x
- 月間アクティブユーザー数

伝説

📊は、そのソリューションが運用され、ハンドブックに組み込まれていることを示します。

🚧は、そのソリューションが`Work In Progress`であり、積極的に開発されていることを示します。この指標を使用する場合、課題はこのページからもリンクされている必要があります。

🏇は、そのソリューションが近いうちに実用化される可能性が低いことを示しています。

---

layout: handbook-page-toc title:"カスタマーセグメンテーション" 説明"顧客セグメンテーションとは、お客様が誰であるかを理解するために、共通の特徴に基づいてお客様をグループに分けるプロセスです。"

このページについて

{:.no_toc}

- TOC {:toc}

カスタマー・セグメンテーション分析

顧客セグメンテーションとは、お客様を理解し、素晴らしいカスタマー・エクスペリエンスを提供するために、共通の特徴に基づいてお客様をグループに分けるプロセスです。お客様を特定するためには、業界、製品カテゴリー、販売セグメント、配送、地域など、さまざまな特徴があります。Customer Segmentation Analysisページでは、GitLabチームメンバーが顧客データを調査し、顧客の洞察力を高めるために使用できる情報とツールを提供します。

このデータソリューションは、3つのSelf-Service Data機能を提供します。

1. ダッシュボードのユーザー業界、製品カテゴリー、販売セグメント、配送、アカウント・オーナー・チーム、テリトリーごとにARRと顧客数を視覚化するSisenseのダッシュボード
2. ダッシュボード開発者。新しいダッシュボードを構築したり、既存のダッシュボードを顧客セグメンテーション・データにリンクさせるための完全な次元モデル・コンポーネントを含む新しいSisenseデータ・モデルです。
3. SQL開発者。An Enterprise Dimensional Modelの対象領域 デー

タプラットフォームの観点から、このソリューションは提供され

ます。

1. 顧客セグメンテーション分析のためのエンタープライズ次元モデルの拡張
2. Data Pipeline Healthダッシュボードのテストとデータ検証の拡張機能
3. ERD、dbtモデル、および関連するプラットフォームコンポーネント

クイックリンク

カスタマーセグメンテーション ダッシュボード　ダッシュボードのユーザー認証 - お客様セグメンテーション

Dashboard Developer Certification - Customer Segmentation　SQL認定資格 - カスタマーセグメンテーション

Sisense Discoveryを使い始める　セルフサービス・ウォークスルー・ビデオ

データセキュリティの分類

Customer Segmentation Dashboardに含まれるデータの多くは、オレンジまたはイエローです。これには、オレンジ色のアカウントの顧客メタデータ、SalesforceやZuoraのコンタクトデータ、GitLabの非公開財務情報などが含まれますが、これらはすべて公開すべきではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する際には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。このデータはSnowflakeとSisenseに残すべきであり、特に承認されない限り、これらのアプリケーションを通じてのみ共有されることが理想的です。

ORANGE

- 説明行またはレコードレベルの顧客および個人データ。オブジェ
- クトです。
    - `dim_billing_accounts`

    - dim_crm_accounts
    - dim_crm_persons

YELLOW

- 説明します。GitLab 財務データ。集約や合計を含む。オブジェクトです。
-
    - dim_subscriptions
    - fct_charges
    - fct_invoice_items
    - fct_mrr

ソリューション・オーナーシップ

- ソースシステムオーナー：
    - Salesforce@jbrennan1
    - Zuora:アンドリュー・マーレイ
- ソースシステムのサブジェクト・マタ
    - ー・エキスパート：Salesforce
    - @jbrennan1 Zuora:アンドリュー・マーレイ（@andrew_murray
- データチームのサブジェクト・マター・エキスパートPaul_armstrong @jeanpeguero @jjstark @iweeks

主な用語

1. 製品カテゴリー、製品階層、配送
2. 販売セグメント
3. アカウントオーナーチーム
4. テリトリー
5. お客様
6. 産業

キーメトリクス、KPI、PI

1. ARR
2. お客様の数

セルフサービス・データ・ソリューション

セルフサービス・ダッシュボード開発者

Sisenseでのチャート構築を始めるには、Sisenseの10分間のDataOnboardingVideoを見るのが良いでしょう。ダッシュボードを作成した後は、そのダッシュボードを簡単に見つけられるようにしたいものです。トピックは、ダッシュボードを1つの場所に整理し、簡単に見つけることができる素晴らしい方法です。ダッシュボードの右上にあるトピックへの追加アイコンをクリックすると、トピックを追加できます。ダッシュボードは、関連性のある複数のトピックに追加することができます。トピックには、Finance（財務）、Marketing（マーケティング）、Sales（販売）、Product（製品）、Engineering（エンジニアリング）、Growth（成長）などがあります。

セルフサービスのSQL開発者

キーフィールドとビジネスロジック

- データは、ZuoraとSalesforceから取得しています。
- 親顧客は1つ以上の製品を持つことができるため、製品カテゴリと配送ディメンションで複数回カウントすることができます。総顧客数のユニークなカウントを得るには、製品と配送をARRAYに集約して顧客のCOUNT DISTINCTを行うか、製品カテゴリや配送ディメンションを含まない顧客のCOUNT DISTINCTを行う必要があります。
- 料金が定期的に発生しているとみなされるには、データ上で有効な終了月が有効な開始月よりも大きい必要があります
- 。Zuoraでは、課金のeffective_end_dateおよびeffective_end_monthは、それぞれ更新の初日または月となります。

- 月間ARR計算では、有効終了月は解約が発生する時期を示しており、有効終了月はARR計算にカウントしません。例えば、有効開始月＝2020-07-01、有効終了月＝2021-07-01の契約では、2020-07-01から2021-06-01までのARRを合計し、12ヶ月分のARRとなります。

エンティティ・リレーションシップ・ダイアグラム

			Diagram/EntityGrainPurposeKeywords
ARR and Customer Count Analytics ERD	月間、サブスクリプション、製品カテゴリー	ARRと顧客数について、様々な角度からのインサイトの提供	親顧客、製品カテゴリー、配送、業界、アカウントオーナーチーム、テリトリー、販売セグメント
Lead to Cash Overview ERD	以下のすべて	リード・トゥ・キャッシュの全プロセスの概要	親顧客、製品カテゴリー、配送、業界、アカウントオーナーチーム、テリトリー、販売セグメント、CRM、担当者、アカウント

リファレンスSQL

	スニペットライブラリ説明
カスタマーセグメンテーション SQLスクリプト	ARRと顧客数を製品カテゴリー、配送、業界、アカウントオーナーチーム、地域、販売セグメント別にスライスするクエリ
顧客セグメンテーション TY四半期とLY四半期の比較 SQLスクリプト	TY対LYのARRと顧客数を四半期ごとに取り出し、製品カテゴリー、配送、業界、アカウントオーナーチーム、テリトリー、セールスセグメントごとに分類するクエリ。

データプラットフォームソリューション

データの系統

- データはSalesforce.comとZuoraから取得し、手動で管理されたZuoraの除外アカウントリストからアカウントを除外して
- います。データの系統図はdbtmart_arr系統図に記載されています。

DBTソリューション

dbtソリューションは、RAWソースデータから次元モデルを生成します。ただし、以下のフィールドは例外で、特定のdbtモデルに実装されたビジネスロジックに基づいて計算されます。

フィールドビジネスロジック

product_	categoryZuoraproduct_rate_plan_nameに基づいて計算されています。
delivery	製品カテゴリーに基づいて計算されます。
サービスタイプ	製品レートに基づいて計算されます。
ultimate_parent_account_	segmentSFDCのmultimate_parent_sales_segmentに基づいて、UnknownとNULLをグループ化して算出したもの。 セグメントをSMBに

トラステッドデータソリューション

TrustedDataFrameworkの概要を見る

dbtガイドの例では、さらなるテストを実施するための詳細と例を示しています。

ズーラ

- 信頼できるデータダッシュボード

    - tdfタグを含むすべてのZuoraデータテストを報告します。
- データパイプラインの健全性を示すダッシュボード

    - ZuoraアカウントとSubscriptionデータのデータの問題点を明らかにするための行数の報告。

EDM エンタープライズ ディメンション モデル バリデーション

- (WIP) エンタープライズ ディメンショナル モデル バリデーション ダッシュボード
    - 最新のEnterprise Dimensionalモデルのテストと実行に関するレポート

RAWソースデータパイプラインの検証

データパイプライン・ヘルスバリデーション

---

layout: handbook-page-toc title:"WIP Email Marketing Data Mart"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

メールマーケティングデータマート

昨年、私たちのチームは、データ要求とライフサイクル・マーケティング戦術を拡張するためのソリューションの必要性に気づきました。

メールマーケティングデータマート（メールマーケティングデータベースとも呼ばれる）は、複数の異なるデータソースを統合し、よりインテリジェントでスケーラブルな方法でユーザー、顧客、見込み客にアプローチできるようにします。

企業のアプリケーションチームやマーケティングチームと協力して、GitLab社が顧客やユーザーの重要なアップデートをより迅速に伝え、見込み客とユーザーの重なりを理解し、より効率的なマーケティングを可能にするソリューションを作り上げました。

このページの目的は

- TD:EmailMarketingDataMartREADME」を使ってメールキャンペーンを実施する際の参
- 考になります。TD: Email Marketing Data Mart」の作成に使用されたデータモデルの理解を助ける。
- `Coming Soon` GitLabでのあなたの役割に最も適した認定を受けることで、あなたの理解度を評価します。
- そして、みんなが貢献できるようにしましょう。

クイックリンク

README - TD: メールマーケティングデータベース

ERD - TD: Email Marketing Database（メールマーケティングデータベース

はじめに

そのためには、お客様に理解していただく必要があります。

- メトリクスをどのように説明するかを説明するキーワード

- データベースの背後にあるデータソース

キータ ーム、キーフィールド、ビジネスロジック

▶ 主な用語

- 販売セグメント
- 販売地域 製品階
- 層 製品納入 ユー
- ザー役割
- サブスクリプションの開始日と終了
- 日 トライアルの開始日と終了日

▶ キーメトリクス、KPI、PI

- 使用方法 Ping
- 試験終了までの日数

▶ キーフィールドとビジネスロジック

データソースとデータモデルの理解

データソース

- Zuora
- Salesforce
- GitLab.com 顧
- 客DB

Entity Relationship Diagram（ERD）の略。

データの系統

▶

▶ クエリの例

追加リソース

▶ Trusted Data Solutionは近日

公開予定です。

▶ EDM Enterprise Dimensional Model Validations
▶ RAW Source Data Pipeline validations
▶ データセキュリティの分類

ORANGE

- 説明行またはレコード・レベルの個人データ。このため、このデータは`PROD.COMMON_MART_MARKETING`ではマスクされているが、`PREP.SENSITIVE`では利用可能である。顧客メタデータ（顧客の会社名など）は`PROD.COMMON_MART_MARKETING`で利用可能です。
- オブジェクトです。
  - `prep.sensitive.dim_marketing_contact`
  - `prep.sensitive.mart_marketing_contact`
  - `prod.common_mart_marketing.mart_crm_touchpoint`
  - `prod.common_mart_marketing.mart_crm_attribution_touchpoint`

▶ ソリューション・オーナーシップ

---

layout: handbook-page-toc title:"マージリクエスト（MR）レート"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

マージリクエスト（MR）レート

MR率は、生産性と効率性の指標です。分子は、一連のプロジェクトに対するマージ要求の集まりです。分母は人の集まりです。どちらも時間をかけて（通常は月単位で）追跡される。

MR Rateの全体および全体の詳細については、こちらをご覧ください。

クイックリンク

MR率ダッシュボード

はじめに

DBTドキュメント

MR率ダッシュボードは
`rpt_gitlab_employees_merge_request_metrics`では、2つの主要なフィルターを考慮しています。

1. ブレイクアウト。ダッシュボードを会社、部門、部署レベルで表示することができます。
2. `Breakout_division_department`の略。このフィルターは子フィルターで、ブレイクアウトの選択に応じて変化します。

▶ `Breakout = Company --> Breakout_division_department`では、`all_company`が1つ選択されます。

- `Breakout = Division --> Breakout_division_department`はGitLabの全ての部門をリストアップします。
- `Breakout = Department--> Breakout_division_department`では、すべての部門と部署の組み合わせが表示されます。

注：`Breakout_division_department`フィルターでは、データが事前に集計されているため、1つのオプションしか選択できません。

ダッシュボード自体は以下のように構成されています。

1. MRレートの推移
2. 少なくとも1つのMRトレンドを持つチームメンバーの%メンバーです。
3. 前月のチームメンバーによるMRレートの内訳
4. どのプロジェクトが計上されていて、どのプロジェクトが計上されていないかの内訳です。

製品の一部であるプロジェクト

MR率とMR量の計算では、製品全体の取り組みに貢献するプロジェクトからのMRを考慮しています。現在のプロジェクトのリストは、gitlab-data/analyticsプロジェクトで以下のシステム・データベースについて特定されています。

システムデータベース｜ファイル｜GitLab.com｜projects_part_of_product.csv｜ops.gitlab.net｜projects_part_of_product_ops.csv｜プロジェクトリストを更新するには、こちらの手順に従ってください。

データモデル

基本となるデータモデルは、多くのデータモデルをあらかじめ考慮していますが、これをGitLabチームメンバーレベルの1つの集約されたデータモデルにまとめ、集約レベルのレポートを作成することで、分析しやすくしています。

gitlab_employees_merge_requests_xf -- GitLabチームメンバーのすべてのマージリクエストを表示し、そのマージリクエストがMRレートの計算に含まれているかどうかを`is_part_of_product`フィールドを使って表示します。

さらに、gitlab_bamboohr_employee_baseを使って、従業員を月別、組織属性（部門、部署など）別、有効期間別に分けたベースモデルを作成しています。

---

layout: handbook-page-toc title:"Estimation Algorithm"

このページについて

{:.no_toc}

- TOC {:toc}

推定アルゴリズム

SaaS型のGitLab.com製品をホストするセルフマネージドインスタンスでは、Usage Pingsから実世界のデータを受け取ります。セルフマネージドインスタンスでは、お客様はUsage Pingを無効にするオプションを持っており、これらのインスタンスの使用量統計を受け取ることはありません。すべてのセルフマネージドカスタマーの使用量を計算するためには、Usage Ping を無効にしたインスタンスの使用量を予測するアルゴリズムを開発する必要があります。このアルゴリズムを「`Estimation Algorithm`」と呼んでいます。

このセクションでは、これらのxMAU推定に関する我々の最初の試みについて説明します。

現在の方法論

私たちが知っていることは？

- 月間アクティブサブスクリプション数
- Usage ping の特定の xMAU カウンタのリリース日（つまり月）。例えば、Create - Source Code Groupの
- GMAUである`merge_requests_users`がバージョン12.9でリリースされたことがわかります。

わからないことは？

- アクティブなフリーインスタンスの数

私たちにできることは？

実際の例を見てみましょう。Dev:CreateステージのSMAUとして使われているカウンタは、`action_monthly_active_users_project_repo`です。これはバージョン13.3でリリースされました。Estimated SMAUを算出するには、次のようなステップを踏みます。

- 記録されたGMAUを配信ごとに分割して算
- 出 1ヶ月ごとに算出
  - 10月の場合、5000のサブスクリプションのうち、Usage Ping Payloadを送信した3500のサブスクリプションがあることがわかります。
  - その中で、バージョン13.3以上のサブスクリプションと13.2以下のサブスクリプション、そしてペイロードを送信してこないサブスクリプションを分けています。
  - このSiSenseのチャートは、月ごとの分割を見たものです。
- 上のグラフは、全アクティブサブスクリプションのうち、ペイロードを送信し、13.4以上を使用しているサブスクリプションの割合を示しています。
- そして、Estimated SMAU = Recorded SMAU / % on 13.3 を導き出します。

カウンターはGitLabの異なるバージョンでリリースされているので、推定値はカウンターごとにカスタマイズされます。

この数値は、実際には有料のXMAUを推定するのに非常に正確な数値ですが、最初は推定XMAUを計算するために使用することができます。しかし、この推定値をより強固なものにするためには、いくつかの改善点が考えられます。

- 最初の改善点は、サブスクリプションの数ではなく、注文したシートの数量の合計を使用することです。また、プラ
- ンごとに見積もりを分割することもできます（現在の問題：1つのサブスクリプションに2つの異なる製品層の2つのインスタンスが含まれる可能性があります）。
- 主な改善点は、Coreの推定に関するものです。現在のところ、バージョンアップや使用量pingのオプトインに関して、有料と無料で同じパターンがあると仮定しています。使用率の推定は非常に複雑ですが、コア・インスタンスのバージョンアップと使用率のデータ傾向については、重要な改善点となるでしょう。

---

layout: handbook-page-toc title:"TD：スナップショット 年間経常収益（ARR）"このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

このページの目的は

- Snapshot ARR Dashboardsの操作方法を理解するのに役立ちます。
- Snapshot ARRダッシュボードの作成に使用されるデータモデルの理解を助ける。
- そして、全体的にみんなが貢献できるようにします。

スナップショット ARR

ZuoraデータのARRは、契約の更新や修正に伴って日々変化しています。そのため、毎日データのスナップショットを作成し、データウェアハウスにARRの過去の記録を残して、レポートや分析に利用する必要があります。ARRのスナップショットとレポート作成に使用する3つの方法を以下に示します。

方法は1つ。

dbtスナップショットは、変更可能なソーステーブルにタイプ2のSlowly Changing Dimensionsを実装しています。Slowly Changing Dimensions (SCD)は、テーブルの行が時間の経過とともにどのように変化するかを示すものです。方法1では、ライブARRメトリクスを生成するために使用されるマートであるmart_arrテーブルをスナップショットします。この方法では、毎日 `mart_arr` の正確なレプリカを作成し、日付スパイン技術を使用してモデル化します。この方法では、スナップショットされた生データを使用せず、

martテーブルをスナップショットしています。これにより、スナップショットされた mart_arr テーブルのシンプルなモデルが作成され、常に mart_arr テーブルのカラムが最新の状態に保たれるという利点があることがわかりました。

mart_arrを完全にリフレッシュする必要はありません。これは、カラムに追従するmart_arrの正確な3リングバインダーコピーが必要で、完全にリフレッシュする必要がないというユースケースにぴったりです。モデルが完全にリフレッシュされたとしても、常に同じ結果を返すことができます。

方法は2つ。

SnowflakeClone機能を使用して、mart_arrのゼロコピークローンを作成します。方法2は、毎日mart_arrの正確な3リングバインダーコピーが必要な方法1と同じユースケースを満たしています。この方法は、dbtから独立した方法を使用することで、方法1のバックアップとして機能します。これらのクローン化されたテーブルは、クエリやレポーティングに使用される主要なテーブルではありませんが、特定の日にdbtスナップショットのパフォーマンスに問題があった場合に、冗長性とフェイルセーフを提供します。

方法は3つ。

方法3では、より伝統的なボトムズアップ式のスナップショットを使用します。のスナップショットを使用して、変更可能なソーステーブルにタイプ2のSlowly Changing Dimensionsを実装し、生のソースデータをスナップショットします。その後、Date Spinningモデリング技術を使用して、スナップショットされたmart_arrテーブルをボトムアップで構築します。このモデルでは、スナップショットされたARRファクトと、ライブのCRM AccountおよびProduct Detailsディメンションを使用します。これにより、Sales CRM の Account Hierarchy と Product Tier や Product Delivery などの Product Details 属性の現在の状態で ARR がどのように見えるかという質問に答えることができます。この方法ではフルリフレッシュが可能で、ある日のARRの合計金額はフルリフレッシュ中に変更されませんが、そのARRのスライスは、ライブのSales CRM Account HierarchyとProduct Detailsディメンションが報告する内容に応じて変更され、更新されます。

Release Train Cadence:

1.近日公開

メンテナンスの予定です。

1.必要に応じて、2週間に1度、金曜日の午前9時から午前11時（米国東部時間）に定期メンテナンスを実施します。

クイックリンク

マーケット分析ハブへ　　Sisense Discoveryを使い始めるには

はじめに

そのためには、お客様に理解していただく必要があります。

- このダッシュボードでサポートされるKPI/PIは？
- メトリクスをどのように説明するかを説明するキーワー
- ド ダッシュボードを支えるデータソース
- さらに探求するために、Sisenseでビジュアルや分析を自分で作成することができます。Sisenseディスカバリーツールから始めるのが良いでしょう。Sisenseを始めたい方はこちらをご覧ください。
- さらに深く掘り下げるために、snowflakeでデータを探索することができます。Snowflakeでの探索の利点は、追加の情報（つまり他のデータソース）に結合できることです。Snowflakeでの探索に関する追加情報はこちらをご覧ください。

キータームメトリクスKPI/PIそしてキーフィールドとビジネスロジック

- 主な用語
  - 近日公開
- キーメトリクスKPI
  - PIは近日公開予定で
    す。
- キーフィールドとビジネスロジック

データソースとデータモデルの理解

ARRのダッシュボードとデータモデルは、ARRのERDにあるデータモデルを使用しています。

- データリネージの
- クエリ例

近日公開

追加リソース

- トラステッドデータソリューション

ARR モデルは、`arr`、`arr_snapshots`、`MRR`、`zuora`、`billing_account`、`crm_account` タグを Trusted Data のテストとその結果に使用します。これは、TrustedDataDashboardを使用して簡単に確認できます。

TrustedDataFrameworkの概要を見る

dbtガイド例 さらなるテストの実施に関する詳細と例 EDM Enterprise

- Dimensional Model Validations
- RAWソースデータ パイプラインバリデ
- ーション データセキュリティ分類

近日公開

ORANGE

- 説明行またはレコードレベルの顧客および個人データ。
- オブジェクトです。
  -

YELLOW

- 説明します。GitLabの財務データで、集約や合計を含みます。
- オブジェクトです。
  -

▶ ソリューション・オーナーシップ

layout: handbook-page-toc title:"License Utilization Analysis"

このページでは

{:.no_toc}

- TOC {:toc}

ライセンス使用状況の分析

お客様が注文したライセンスをどのように消費しているかを理解することは、お客様の全体像を把握する上で非常に重要なステップです。これは、製品が潜在的な導入問題を特定するのに役立ち、営業チームやTAMチームが顧客をよりよく理解し、解約やダウングレードの潜在的なリスクを特定するのに役立ちます。

このページはMVCであり、次の四半期に向けての意思表示と捉えていただく必要があります。

キーメトリクス、KPI、PI

- ライセンス稼働率は、アクティブユーザー数 / 注文したライセンス数として算出
- されます。ライセンス使用率は、使用されたライセンス数 / 注文されたライセンス数として計算されます。

将来の分析が必要

これは、SaaSとセルフマネージドの両方のライセンス・アクティビティ・レートを示す予備的な分析でした。データ・チームは、FY21 Q1ではこの分析にさらに焦点を当て、より深い分析を行いたいと考えています。焦点を当てるべき重要な分野は以下の通りです。

- ライセンス利用率にも同じグラフを表示してみる
- ライセンス使用率/アクティベーション率の低さと解約/ダウングレードの相関関係を探る 製品
- 層、案件規模、業界別に分析を深める

さらなる努力が必要

このページを当社の品質基準に合わせるためには、以下のことが必

- 要になります。EDMモデルの作成とERDの作成
- L2ソリューションへの取り組み ページ

---

layout: handbook-page-toc title:"Manual Upload of Usage Payload" こ

のページについて

{:.no_toc}

- TOC {:toc}

利用データを用いたユーザーへの取り組み

- 使用状況のPingデータを初めて手動でアップロードする場合は、`snowflake_imports` GCSバケットへのアクセスが必要です。アクセスリクエストを作成してください。アクセスの準備ができたら、アクセスリクエストに `@gitlab-data/engineers` グループをタグ付けします。
- `snowflake_imports`GCSバケットに顧客用のフォルダがまだ存在しない場合は、作成します。
- `snowflake_imports`GCSバケットの、そのお客様のフォルダ内に、使用方法のpingペイロードファイルを配置します。まず、そのフォルダをクリックします。次に、「`UPLOAD FILES`」ボタンを押し、ファイルを選択して「Upload」を押します。
- ファイルを整理する。理想的には、ファイルはjson形式で、*例えばcompanyname_usage_payload_00_00_0000.jsonのようにな*る。
- 使用状況のpingデータは、GitLabインスタンス内でどの機能やサービスが利用されているかを示す情報です。詳しくは

こちらのリンクをご覧ください。

layout: handbook-page-toc title:"グループ名空間の変換メトリクス" このページの目次

{:.no_toc}

- TOC {:toc}

グループ名空間の変換メトリクス。GitLab.com

新しいグループのネームスペースがGitLab.comに作成されると、ネームスペースの作成者とユーザーは、自分のネームスペースにメンバーを追加したり、特定のステージイベントを完了したり、最終的には無料トライアルから有料プランに変更したりして、GitLab.comとの関わりをさらに深めていくことが非常に重要になります。また、GitLab.comでネームスペースが作成された後、これらのイベントやアクションがどれだけ早く行われているかを知ることも重要です。

ダッシュボードで答えた質問

- ある期間内に、新規グループの名前空間のうち、無料トライアルから有料プランにアップグレードされ
- た割合はどのくらいですか？一定期間内に新規のグループネームスペースのうち、2名以上のメンバーで構成されている割合は？
- 新規グループのネームスペースのうち、一定期間内に「作成」ステージに参加した割合は？新規
- グループのネームスペースのうち、一定期間内に「Verify」ステージに参加した割合は？
- ステージの採用数、転換数、ネームスペースの作成数に大きな変化はありましたか？

このページの目的は

- グループネームスペースの操作方法の理解を助ける 変換指標の理解を助ける データモデルの
- 理解を助ける (WIP)
- そして、全体的にみんなが貢献できるように。

クイックリンク

新しいグループ名空間の変換メトリクス・ダッシュボード

グループ名空間コホート分析ダッシュボードビュー（WIP

データの注意点と制約事項

- 無料」または「有料」でフィルタリングすると、グループネームスペースの現在のステータスが反映され、以下のようになります。
  は、採用イベントの時点でのネームスペースのアカウント状況を把握していません。
- これらのレポートには、新規ユーザーおよび既存の有料アカウントのユーザーが作成したグループ
- 名前空間が含まれます。データには、スパムユーザーによって作成された名前空間が含まれる場合があります。
  このデータはGitLab.comの製品のみを反映しており、自己管理の利用状況は報告されていません。

データセキュリティの分類

グループ名空間変換ダッシュボードのデータの多くはオレンジ色であり、一般に公開すべきものではありません。このダッシュボードのデータを共有する際には、詳細をGitLabの組織内にとどめ、外部に共有する際には適切な承認を得るように注意してください。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。このデータはSnowflakeとSisenseに残すべきであり、特に承認されない限り、これらのアプリケーションを通じてのみ共有されることが理想的です。

ソリューション・オーナーシップ

- GitLab.comクレイグ・ゴメス（@craig-gomes
- データチームのサブジェクト・マター・エキスパート@dpeterson1 @kathleentam @mpeychet_ @iweeks @jeanpeguero @m_walker @pluthra

新しいグループ名空間 作成段階 採用率@mkarampalas
新しいグループ名空間の検証段階の採用率。@jstava 少なくとも2人のユーザーが追加された新しいグループ・ネームスペース。
@s_awezec

- 新しいグループ名空間の試みから有料コンバージョン率へ。@s_awezec

主な用語

- グループネームスペース。トップレベルのネームスペースで、サブグループやプロジェクトを保持できます。これらのKPIおよび関連ダッシュボードのデータには、個人のネームスペースは含まれていません。
- 名前空間の作成日。新しいグループネームスペースが作成された日付（週
- ステージを検証する。グループネームスペースが最初のX日間でci_pipelinesイベントを登録すれば成功
- 作成段階。グループネームスペースが最初のX日間でaction_monthly_active_users_project_repoイベントを完了した場合に成功
- 有料プラン。名前空間が現在有料か無料かを示します。

ダッシュボードのフィルター

- 有料プランです。無料プラン、有料プランでの絞り込み
- 最初の X 日間フィルター。グループネームスペースが作成されてから X 日以内の、これらの KPI のコンバージョン率と採用率を示します。現在のオプションには、最初の3日、7日、14日、30日、60日、90日、180日があります。
- 集約。ネームスペースの作成日を、選択した日付間隔（日、週、月、四半期、年）で集計します。 日付範囲。
- 過去X日以内に作成されたグループネームスペースのみを表示するようにチャートをフィルタリングします。

キーメトリクス、KPI、PI

新規グループ名空間試用から有料転換率 新規グループ名空間 作成段階での採用率 新規グループ名空間

検証段階での採用率 2人以上のユーザーが追加された新規グループ名空間

キーフィールドとビジネスロジック

- データは主にSnowflakeから取得しています。

スニペット

スニペットは、異なるチャートで再利用可能なSQLコードの文字列を作成するために使用されます。詳細については、Sisense SQL Snippetsのページをご覧ください。

リファレンスSQL

	スニペットライブラリ説明
分母_グループ_名前空間_on_trial	このスニペットは、「New Group Namespace Trial to Paid」の分母として使用されます。コンバージョン率のKPIで、作成されたグループネームスペースのうち、最初のX日以内に試用を開始したグループネームスペースの総数を算出します。
分水嶺_全グループ名空間	クエリは、上記の残りの 3 つの KPI の分母として使用されます。分母は、作成されたグループネームスペースの数を計算します。denominator_group_namespaces_on_trialのスニペットとは異なり、このスニペットはトライアルを開始していない新規グループネームスペースを除外しません。

エンティティ・リレーションシップ・ダイアグラム

			Diagram/EntityGrainPurposeKeywords
グループ名空間 変換指標 ERD	日付、場所、プランの種類	キーとなるステージのイベントやアクションの新ネームスペース採用に関する洞察を提供する	Stage Adoption, Invite Members, New Groups, Paid Conversion, New Namespaces, GitLab.com

データプラットフォームソリューション

データの系統

近日公開

DBTソリューション

近日公開

トラステッドデータソリューション

- トラステッドデータフレームワーク

近日公開

EDM エンタープライズ ディメンション モデル バリデーション

- エンタープライズ次元モデル検証ダッシュボード

近日公開

RAWソースデータパイプラインの検証

近日公開

カスタマーDB

- 信頼できるデータダッシュボード
  - tdfタグを含むすべてのCustomerDBデータテストのレポート。

近日公開

---

layout: handbook-page-toc title:"People Analytics Overview"

このページについて

{:.no_toc .hidden-md .hidden-lg}の

ように。TOC {:toc .hidden-md

.hidden-lg}。

{::options parse_block_html="true" /}。

ピープルアナリティクス

仕事をしている人の研究です。人事部は組織行動へのアプローチを反転させ、判断や意見を用いて意思決定を行う代わりに、データの力で事実に基づいた意思決定を行っています!

ピープルスペースにおけるデータのメリット

- より合理的な人材獲得プロセスを構築することで、GitLabが強固で多様性のあるチームを構築することを支援し、プロセスを経る候補者にも充実した経験を提供します。
- GitLabチームメンバーの体験を常に向上させるために、チームを推進しています。"Everyone can contribute "という言葉は、私たちが本気で言っていることです。エンゲージメント調査やKPIを用いてチームの目標を達成するために、私たちは人を最優先に考えています。
- 他のデータと結びつけようセールスデータであれ、エンジニアリングデータであれ、私たちにとってはすべてが重要です。私たちは、チームメンバーの成長をどのように支援できるか、またリーダーが組織の影響力をどのように理解するかを理解したいと考えています。

一般的なルールとガイダンス

- ピープルグループは、運用ソフトウェア（BambooHR、Greenhouseなど）の中で日々の業務を行うことができなければなりません。ウェアハウス内のピープルデータは、組織内の一般的なピープル情報を報告するためのものでなければなりません。
- リスクを把握するために、「ゆりかごから墓場まで」データにアクセスする役割とユーザーを常に把握する。

- レポート作成に必要なものだけをデータウェアハウスに取り込み、その他の機密性の高いデータは運用ツールに残しておくべきです。
- 可能な限りリスクを軽減するために、メタリックな計算や報告に使用されるセンシティブなデータを匿名化する。

人々のデータソース

BambooHR

人事管理システムです。

温室

採用・応募者管理システム

PTO By Roots

チームメンバーの休暇を記録するslackアプリケーション

---

layout: handbook-page-toc title:"People KPI Deck Report"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

People KPI Deck Report

このレポートは、People Groupの月例KPIミーティングで説明された指標を考慮しています。こ

のレポートは、このGoogleレポートに掲載されています。

基礎となるデータ

`rpt_people_kpis`データモデルのドキュメントは、こちらをご覧ください

。 これは、過去12ヶ月間のデータをキャプチャします。

---

layout: handbook-page-toc title:"プロモーションレポート"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

プロモーションレポート

このレポートは、トータルリワードチームのため

に作成されます。レポートは、このgoogleフォ

ルダにあります。

プロモーションレポートの更新

1. `Promotions Last 12 Months`」タブを更新します。A. snowflakeに向かう（Oktaでログイン） B. 以下のコードを実行する。

```
SELECT *
from "prep". "sensitive". "bamboohr_promotions_xf"
WHERE promotion_month BETWEEN DATEADD(year,-1, DATE_TRUNC(month, CURRENT_DATE()))AND
DATEADD(moon,-1, DATE_TRUNC(moon, CURRENT_DATE()))
ORDER by promotion_date
```

C.結果をダウンロードして、「過去12ヶ月間のプロモーション」タブに配置します。

2. Budget Spend」タブを更新する A. snowflakeで、以下のコードを実行する。

```
SELECT *
"prep""sensitive""bamboohr_budget_vs_actual "より
ORDER BY 会計年度, 会計四半期, 部門
```

B.結果をダウンロードして、「予算支出」タブに入れます。

3. 予算と実績のピボットを更新する 注：予算の合計を更新するには、ここをクリックします。People Budget Sheetload。このデータは全体のレポートを更新するために使用されます。

4. リフレッシュ時にプロモの予算を更新するには、以下の方法があります： sheetload.budget_people

---

layout: handbook-page-toc title:"スラックダッシュボード"

Slackダッシュボード

このダッシュボードは、Chief of Staff Team to the CEOのKPIに関連して、DMSaではないメッセージの割合を監視するために使用されます。このダッシュボードは、パブリックチャネルの利用を増やし、ハンドブックファーストにする方法を理解するのに役立ちます。

ダッシュボードはこちらからご

覧いただけます####Updating

Data

1. Head to https://gitlab.slack.com/stats
2. 日付範囲を全時間帯に変更してデータをエクスポート
3. 以下のコードを使用してデータをクリーンアップします（これはR Studioから行うことができます）
   - https://gitlab.com/gitlab- data/analytics/-blob/master/transform/general/clean_slack_data.R
4. クリーニングされたデータを入手したら、sheetload.slack_stats(https://docs.google.com/spreadsheets/d/15a2PVvSs7K_C-)に向かいます。
   EsGq2hKLUYNywxPmE74Ldx-YbXqN-w/edit#gid=673732546）を作成し、現在のタブのデータを新しいデータで置き換える。

---

layout: handbook-page-toc title:"Talent Acquisition Metrics"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

タレント・アクイジション・キー・メトリクス

GitLabはチームメンバーをサポートすることに重点を置いています。他のビジネスステークホルダーと同様に、People Group内のTalent Acquisitionチームは、チームが社内で適切なチームメンバーを採用できるように採用プロセスを常に改善し、またプロセ

スを経た候補者が良い経験をできるようにすることに重点を置いています。

タレント・アクイジションのKPIの多くは、KPIインデックスに掲載されています。

TalentAcquisitionMetricsDashboardは、他の指標に加えて、当社のKPIを考慮しています。

クイックリンク

タレント・アクイジション・メトリクス・ダッシュボード

モデル・ドキュメンテーション

ダッシュボードの使い方 People Metrics Dashboard

人材獲得指標ダッシュボードでは、先月、前四半期末のGitLab社の状況を簡単に把握することができ、選択した日付範囲（デフォルトは過去12ヶ月間）でのトレンドを確認することができます。

ダッシュボードでは、ユーザーは以下の項目でフィルターをかけることができます。

- タレント・アクイジション部門
- Talent Acquisition Department (部門レベルに移行するには、部門が選択されている必要があり
- ます) Talent Source
- タレントソース名（タレントソースが選択されている必要があります
- タレント・アクイジション・チーム（エンジニアリング・タレント・アクイジション・チーム、セールス
- ・タレント・アクイジション・チームなど） タレント・アクイジション・チームのリクルーター（タレント・アクイジション・チームのフィルターに依存する
- 候補者コーディネーター
- Is Prospect（候補者が、当社がアプローチした潜在的な見込み客であるのか、それとも他のソースから応募して採用プロセスを開始した候補者であるのかを確認するため
- ソーシング・チーム・フラッグ
- Sourcing Team Sourcer (`sourcing team flag set = sourcing team`に依存) このダ

ッシュボードを使用する際の重要なヒントがあります。

1. メトリクス名が青で表示されている場合は、データをより深く掘り下げることができる付属のダッシュボードがあります。リンクをクリックすると、ダッシュボードに移動します。

タレント・アクイジション・メトリクス・データ・モデル

このダッシュボードは `greenhouse_stage_analysis` をベースに作られています。ソースデータはGreenhouseからのものです。

このデータモデルは、採用プロセスを経た各候補者の概要と、主要な属性（ソース、リクルーター、求人部門など）に加えて、どのステージに到達したかを示しています。

このモデルを作成するための基礎データは、温室ロールにアクセスできるチームメンバーがアクセスでき

ます。Sisenseでは、基礎となるクエリが次のように表示されます。

```
WITH talent acquisition_data AS

  ( SELECT * )
  FROM LEGACY.GHGENUSIVEAN STAGE_ANalysis
  WHERE [division_modified=Talent Acquisition Division]
    AND [department_name=talent acquisition_department]
    AND [source_type=Talent_Source] 。
    AND [sourcing_team=Sourcing_Team_Flag]
    AND [sourcer_name=Sourcing_Team_sourcer]
    。
    AND [talent acquisition_team=talent acquisition_team]
```

```
        AND [candidate_recruiter=Talent Acquisition_Team_Recruiter]
        AND [source_name=Talent Acquisition_Source_Name] 。
        AND [candidate_coordinator=candidate_coordinator]
        AND [is_prospect=Is_Prospect]となります。


    [talent acquisition_metrics] --- これは、すべての主要なメトリクスを識別するsisenseスニペットを使用しています。
例えば、どのような候補者が応募者とみなされるのか、応募から審査までの割合はどのくらいか、応募から採用までの割合はどのくらいか、などを計算します。

    スニペット全体は[こちら
](https://app.periscopedata.com/app/gitlab/snippet/recruiting_metrics/55081acd27f44d7fb1706d47f44b5ae8/edit)でご覧いただけます。

    SELECT prospected
    FROM metrics
    WHERE month_stage_entered_on = DATE_TRUNC(month, CURRENT_DATE())
```

---

layout: handbook-page-toc title:"チームメンバーの分離"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

チームメンバーの分離レポート

このレポートは、分離されたすべてのチームメンバーを特定します。注：このデータは、roles.ymlで bamboohr_sensitive_separationロールを持つユーザーがsnowflakeでのみアクセスできます。このデータは Sisenseではアクセスできません。

```
ランニングでデータにアクセスできます。

``
SELECT
*.
from "prep". "sensitive". "bamboohr_separations"
```

アクセス方法

このレポートにアクセスするためには、アクセスリクエストを作成する必要があります。

1. スノーフレーク
2. bamboohr_sensitive_separationロールへのアクセス。

このアクセスリクエストは、ピープルチーム（トータルリワードマネージャーまたはピープルサクセス担当シニアディレクター）が確認する必要があります。

---

layout: handbook-page-toc title:"PTO by Roots"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

PTO by Roots（

Slack） ゴール

GitLabチームメンバーはslackを使って休暇を記録しています。このデータを他のデータソースと結びつけることで、
理解を深めるのにとても役立ちます。

- 休みを取っていないメンバーの割合は？ 家族と友人
- の日の計画を手伝う
- 今後のキャパシティを把握し、マイルストーンやOKRを計画する パ
- フォーマンス指標への影響を測定する

具体的な例としては、開発部の狭義のMR率（チームメンバーの生産性を示す指標）の変動を把握するためにPTOデータを使用しています。当社では、メンバーに充電のための休暇を取ることを奨励しており、休暇を考慮することで、時系列で見たMR率の低下や上昇を説明することができます。例えば、前月に比べて休みが増えたことで、今月の狭義のMR率が下がったことを説明できるかもしれません。

このデータを使って、個々のチームメンバーの欠席時間や理由を評価しないでください。個人の出席状況に懸念がある場合は、担当のピープルビジネスパートナーに連絡してください。

データプロセス

私たちが取得するデータ

- データレベルでは、理由ではなく、休んだ日が気になります。したがって、我々のデータでは、extract層にある休んだ理由やsnowflakeで利用可能なものはすべて削除しています。
- このデータは、様々なデータソリューションに使用するために集計されます。例えば、MR率については、月、部門、部署のレベルでKPIを表示します。休み時間やチームメンバーの情報は、Sisenseではアクセスできません。

このタイムオフデータへのアクセスは、データチーム（Parul Luthra, Data Engineer-未定）に限定されています。アクセスを希望する場合は、アクセスリクエストフォームを提出し、SlackのPTO管理者、Senior Director, People Successの承認を得る必要があります。

---

layout: handbook-page-toc title:"People Metrics"

People Key Metrics

ギットラボはチームメンバーのサポートに力を入れています。他のビジネスステークホルダーと同様に、ピープルグループは、チームメンバーがギットラボでの貢献に集中できる環境を作るために、データを重視しています。

People KPIの多くは、KPIIndexに掲載されています。

PeopleMetricsDashboardは、当社のKPIに加え、ピープルデータの意思決定に役立つその他の指標も考慮しています。

クイックリンク

ピープルメトリクス ダッシュボード　ピープルメトリクスの概要　モデルドキュメント

人物データ発見ダッシュボード　データモデルのデモ　データモデルのビデオ・デモとダッシュボード

ダッシュボードの使い方 People Metrics Dashboard

PeopleMetricsのダッシュボードでは、先月、前四半期末のGitLabの状況を簡単に把握することができ、選択した日付範囲（デフォルトでは過去12ヶ月）での会社の傾向を確認することができます。

ダッシュボードは、3つのレベルにフィルタリングすることができます。

ダッシュボ　ードビュー

フィルター **1**フィルター**2**

ダッシュボードビュー	フィルター	1フィルター2
GitLab	OverallBreakout =	CompanyBreakout_Division_Department = All Company
GitLab	OverallBreakout =	DivisionBreakout_Division_Department = 該当する部門のフィルタリング
GitLab	OverallBreakout = 部門	Breakout_Division_Department = 該当する部門でのフィルタリング

注：この画面では、一度に1つの部門または1つの部署しか選択できません。このダッシュボードを使用する際の重要なヒントがあります。

1. メトリクス名が青色で表示されている場合は、データをさらに詳しく調べることができるダッシュボードが表示されます。リンクをクリックすると、ダッシュボードに移動します。
2. すべてをレポートにまとめて見る必要がある場合は、「PeopleMetricsOverview-CurrentMonth」をご覧ください。これはダッシュボードからアクセスできます。
   - これは、ドリルダウンレポートと同じように機能します。たとえば、People_Metrics_View = 'Division` でフィルタリングすると、すべての kpis の内訳がレポートに表示され、上部には部門が、最後には GitLab Overall が表示され、部門レベルでの比較が簡単にできます。
     - これは、どの部門の離職率が高いのか、どの部門の採用が遅れているのかといった質問に最適です。
   - People_Metrics_View = 'Department'でフィルタリングすることにより、ビューアーは上記の質問をより深く掘り下げ、部門レベルで問題があるかどうかを判断することができます。

People Metricsデータモデル

ダッシュボードの大部分は、bamboohr_rpt_headcount_aggregationをベースに構築されています。このデータモデルは、4つのレベルですべての関連する人数メトリクスを月ごとに集計します。

1. kpi_breakout -- 会社レベルまで集計されたすべての指標を表示する。
2. ディビジョン_ブレイクアウト
3. デパートメント・ブレークアウト
4. eeoc_breakout -- ダイバーシティ指標を理解するために集計された指標を示します（性別、民族、地域など）。

このモデルを作成するための基礎データは、チームメンバーがbamboohr_analyticsロールにアクセスし、よりセンシティブなデータ（例：eocデータや報酬）についてはbamboohrロールにアクセスします。

独自のレポートを作成します。

自分でレポートを作成したい方は、Sisenseのディスカバリー機能の使い方を説明したデモビデオをご覧ください。これにより、ユーザーはフィールドをドラッグ＆ドロップすることができます。

この機能のプレビューは、PeopleDataDiscoveryFeatureDashboardでもご覧いただけます。

---

layout: handbook-page-toc title:"価格分析"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

価格分析

価格設定とは、特定の価格帯でお客様がGitLabから受け取る価値と、その価格帯の収益性を分析するプロセスです。また、これらの価格がビジネス全体にどのような影響を与えるかを理解し、最適な価格を決定することも含まれます。

の価格設定は、顧客とGitLabの収益性を左右するものです。価格分析ページでは、GitLabチームメンバーが現在の価格戦略を検討し、最適化のための洞察を深めるために使用できる情報とツールを提供します。

このページの目的は

- 価格設定-顧客割引ダッシュボードの操作方法の理解を助けます。
- Pricing - Customer Discounts Dashboard」の作成に使用されたデータモデルの理解を助けます
- 。GitLabでのあなたの役割に最も適した認定を受けることで、あなたの理解度を確認します。
  - ダッシュボードの使い方を詳しく知りたい方は、ダッシュボードのユーザー認定を受けてください。
  - Sisenseダッシュボードの開発について詳しく知りたい方は、DashboardDeveloper認定資格を取得し
- てください。 そして、全体的にみんなが貢献できるようにしてください。

クイックリンク

価格 - 顧客割引ダッシュボード　　PnPテスト＆リサーチ ダッシュボード

価格 - お客様の概要ダッシュボード　　Sisense Discoveryの使用開始について

はじめに

始めるにあたり、以下のことを確認したいと思います。

を理解しています。

- このダッシュボードでサポートされるKPI/PIは？
- メトリクスをどのように説明するかを説明するキーワー
- ド ダッシュボードを支えるデータソース
- さらに探求するために、Sisenseでビジュアルや分析を自分で作成することができます。Sisenseディスカバリー・ツールを使用するのが良い方法です。Sisenseを始めたい方はこちらをご覧ください。
- さらに深く掘り下げるために、snowflakeでデータを探索することができます。Snowflakeでの探索の利点は、追加の情報（つまり他のデータソース）に結合できることです。Snowflakeでの探索に関する追加情報はこちらをご覧ください。

目で見るのがお好きな方は価格分析ソリューションのトレーニングをご覧ください。

キーターム、メトリクス、KPI/PI、そしてキーフィールドとビジネスロジック

- 主な用語
- キーメトリクス、KPI、PI
- キーフィールドとビジネスロジック

データソースとデータモデルの理解

このダッシュボードは、Sisenseの`base_pricing_discount`スニペットから作成されています。Sisenseのスニペットやビューを使うと、シンプルなSQL文字列を好きなだけチャートで参照することができます。これにより、エンドユーザーがARRやシート数などの計算の裏にあるロジックを理解したり、どのような結合が可能かを推測で理解したりするのに役立ちます。

このモデルから独自のダッシュボードを作成するには、Sisenseのクエリに次のように入力するだけです。

> [ベース_プライシング_ディスカウント]

`Base_pricing_discount` は `mart_discount_arr` を基に構築されており、様々な顧客ディメンション（親顧客、製品カテゴリー、配送、アカウントオーナーチーム、再販業者/そうでないもの、サブスクリプション、アカウントなど）による割引や再販取引についての洞察を提供します。

このマートは、ERD（Entity Relationship Diagram）に見られるようなデータモデルを考慮しています。

▶ データリネージの
▶ 例 クエリ

[ベース_プライシング_ディスカウント]

)

SELECT (SUM(arr_list_price) - SUM(arr))SELECT (SUM(arr_list_price) - SUM(arr)) / SUM(arr_list_price) AS discount FROM base_pricing_discount WHERE [effective_start_month=daterange_no_tz]とします。

```
</details> </details
<br>



<style> #headerformat {
background-color:6666c4; color: black; padding:5px; text-align: center;
}
</style>
<h1 id="headerformat">追加のリソース</h1>。

<詳細
<summary markdown='span'> トラ
  ステッドデータソリューション
</summary>
概要は[Trusted Data Framework](https://about.gitlab.com/handbook/business-technology/data-team/platform/#tdf)をご覧ください。

[dbtガイドの例](https://about.gitlab.com/handbook/business-technology/data-team/platform/dbt-guide/#trusted-data-framework)を参照してください。
さらなるテストの実施に関する詳細と例
</details> </details

<詳細
<summary markdown='span'>
  EDM エンタープライズ ディメンション モデル バリデーション
</summary>
The [(WIP) Enterprise Dimensional Model Validation Dashboard] (エンタープライズ ディメンショナル モデル バリデーション ダッシュボード)
```

(https://app.periscopedata.com/app/gitlab/760445/WIP:-Enterprise-Dimensional-Model-Validation-Dashboard）では、最新のEnterprise Dimensionalモデルのテストと実行について報告しています。
</details> </details>

<詳細
<summary markdown='span'>
  RAWソースデータパイプラインの検証
</summary>
[Data Pipeline Health Validations](https://app.periscopedata.com/app/gitlab/715938/Data-Pipeline-Health-Dashboard)
</details> </details>

<詳細
<summary markdown='span'>
  データセキュリティの分類
</summary>
価格分析のデータの多くは、[オレンジ]（/handbook/engineering/security/data-classification-standard.html#orange）または[イエロー]（/handbook/engineering/security/data-classification-standard.html#yellow）となっています。これには、ORANGEのアカウントの顧客メタデータや、GitLabのNon publicの財務情報が含まれており、これらはすべて一般に公開されるべきではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する場合には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。これらのデータは、[Snowflake](/handbook/business-technology/data-team/platform/#data-warehouse)と[Sisense](/handbook/business-technology/data-team/platform/periscope/)に残すべきであり、特別な承認がない限り、これらのアプリケーションを通してのみ共有されるのが理想的です。

ORANGE。

- 説明行またはレコードレベルの顧客および個人データ。
- オブジェクトです。
  - dim_billing_accounts`。
  - dim_crm_accounts`。

YELLOW**（イエロー

- 説明します。GitLabの財務データで、集約や合計を含みます。
- オブジェクトです。
  - dim_subscriptions
  - `fct_charges`
  - `fct_invoice_items`
  - `fct_mrr`
</details> </details>

<詳細
<summary markdown='span'>
  ソリューションの所有権
</summary>
* ソース・システム・オーナー
  * セールスフォース: `@jbrennan1`.
  * Zuora: `@andrew_murray` (英語)
* ソースシステムのサブジェクト・マター・エキスパート。
  * セールスフォース: `@jbrennan1`.
  * Zuora: `@andrew_murray` (英語)
* データチーム・サブジェクト・マター・エキスパート：`@paul_armstrong` `@jeanpeguero` `@jjstark` `@iweeks`
</details> </details>

レイアウト：ハンドブック-ページ-トック
のタイトルを付けました。"Product Geolocation Analysis"

```
## On this page
```

```
{:.no_toc}

- TOC
{:toc}

---

## Product Geolocation Analysis : Self-Managed

自社の製品が世界中のどこで使われているかを理解することは、顧客や製品のグローバル展開、関連するロケーションインサイトをより深く理解するための重要なステップです。

現在、GitLabのお客様の大半は、「GitLabセルフマネージドインスタンスをダウンロードし、インストールし、ホストする」(/handbook/marketing/strategic-marketing/dot-com-vs-self-managed/#why-you-probably-want-gitlabcom)ことを選択しており、そのために私たちは「素晴らしいセルフマネージドカスタマーエクスペリエンスを提供することに重点を置いている」(/direction/#strategic-challenges)。

セルフマネージ製品のライフサイクルや投資すべき機能について、データに基づいた適切な判断を下すために、[セルフマネージのお客様](/is-it-any-good/)は、[セルフマネージのインスタンスでusage pingを有効にする](https://docs.gitlab.com/ee/user/admin_area/settings/usage_statistics.html#instance-level-statistics)ことで、またはカスタマーサクセスチームと値を共有することで、インスタンスレベルで毎週GitLabに[usage ping](/handbook/customer-success/tam/usage-ping-faq/)を送信しています。

このインスタンスレベルのデータにより、GitLabは国レベルの統計や、インスタンスの採用率、バージョン採用率、インスタンスのライフサイクルの傾向を把握することができます。

**このページの目的：**。
  * Product Geolocation dashboard] の操作方法の理解を助ける(https://app.periscopedata.com/app/gitlab/731086/Product-Geolocation-:-Self-Managed-(doesn't-include-SaaS))
  * Product Geolocation Dashboardの作成に使用されたデータモデルの理解を深めることができます。
  * GitLabでのあなたの役割に最も適した認定を受けることで、あなたの理解度を評価してください。
    * Product Geolocationダッシュボードの使用方法については、[Dashboard user certification](https://docs.google.com/forms/d/e/1FAIpQLScflqXOnU-W6kz24b5qD715uw9_1s6tfKF34qf1tqvFgguIVw/viewform)をご利用ください。
    * Sisense ダッシュボードの開発について詳しく知りたい方は、【ダッシュボード開発者認定】（https://docs.google.com/forms/d/e/1FAIpQLSeqicaMfWVUfFsex9_o6GTkWJKobYBT8qucz9YNmyDm5ZKqiA/viewform）を受けてみてください。?usp=sf_link)となっています。
    * SQLを使ってデータを深く理解するためには、[SQL Certification]（https://docs.google.com/forms/d/e/1FAIpQLScH9CkiACQ1worzldjUi6cUWUL03tXrLEEaZALABabZPV7GuQ/viewform）を受講してください。?usp=sf_link)となっています。
  * そして、全体的にみんなが貢献できるように。

### Quick Links
<div class="flex-row" markdown="0" style="height:80px">。
  <a href="https://app.periscopedata.com/app/gitlab/731086/Product-Geolocation-:-Self-Managed-(does not-include-SaaS)" class="btn btn-purple" style="width:33%;height:100%;margin:5px;float:left;display:flex;justify-content:center;align-items:center;">製品のジオロケーション（セルフマネジメント・ダッシュボード）</a>。
  <a href="https://www.youtube.com/watch?v=F4FwRcKb95w&feature=youtu.be" class="btn btn-purple" style="width:33%;height:100%;margin:5px;float:left;display:flex;justify-content:center;align-items:center;">Sisenseをはじめよう</a
</div> </div>
<br> <br>



### Getting Started
そのためには、お客様に理解していただく必要があります。
1.このダッシュボードを使ってどのようなKPI/PIがサポートされるのか
```

1. メトリクスをどのように説明するかを説明するキーワード

1.ダッシュボードのデータソース
1.さらに探求するために、Sisenseでビジュアルや分析を自分で作成することができます。Sisenseディスカバリーツールを使用して始めるのが良い方法です。Sisenseを始めてみたい方はこちらをご覧ください。](https://about.gitlab.com/handbook/business-technology/data-team/direction/self-service/#self-service-dashboard-developer)
1.さらに深く掘り下げるために、snowflakeでデータを探索することができます。Snowflakeでの探索の利点は、追加の情報（つまり他のデータソース）に結合できることです。Snowflakeでの探索に関する追加情報は[こちら](https://about.gitlab.com/handbook/business- technology/data-team/direction/self-service/#self-service-sql-developer)をご覧ください。

Product Geolocation Dashboardは、以下のKPIやPIの把握に役立ちます。

- [アクティブホスト](/handbook/product/performance-indicators/#active-hosts)
- [ロスト・インスタンス](/handbook/product/performance-indicators/#lost-instances)
- [有料ユーザー](/handbook/product/performance-indicators/#paid-user)
- [有料UMAU](/handbook/product/performance-indicators/#paid-umau)
- [月間ユニーク・アクティブ・ユーザー数 - UMAU](/handbook/product/performance-indicators/#unique-monthly-active-users-umau)

ダッシュボードをご覧になる際には、以下の点をご理解ください。

- [アカウント](/handbook/sales/#additional-customer-definitions-for-internal-reporting)
- アカウントインスタンス - 1つのアカウントにマッピング可能なレポートされたインスタンスの合計数
- [ホスト](https://docs.gitlab.com/ee/development/product_analytics/event_dictionary.html)
- [インスタンス](https://docs.gitlab.com/ee/development/product_analytics/event_dictionary.html)
- Instance User Count - インスタンス上のユーザーの合計数
- [有料ユーザー](/handbook/product/performance-indicators/#paid-user)
- [製品階層](/handbook/marketing/strategic-marketing/tiers/#overview)
- [使用方法 Ping](https://docs.gitlab.com/ee/development/product_analytics/event_dictionary.html)
- [バージョン](/handbook/sales/process/version-check/#what-is-the-functionality of the-gitlab-version-check)

注：製品ジオロケーション分析をサポートする追加データは、[オレンジ](/handbook/engineering/security/data-classification-standard.html#orange)または[イエロー](/handbook/engineering/security/data-classification-standard.html#yellow)に分類されます。これには、アカウントからのORANGEの顧客メタデータ、SalesforceやZuoraからのコンタクトデータ、GitLabのNon公開財務情報などが含まれ、これらはすべて公開されるべきものではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する際には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。これらのデータは、[Snowflake](/handbook/business-technology/data-team/platform/#data-warehouse)と[Sisense](/handbook/business-technology/data-team/platform/periscope/)に残すべきであり、特別な承認がない限り、これらのアプリケーションを通じてのみ共有されることが理想的です。
ORANGE。

- 説明行またはレコードレベルの顧客および個人データ。
- オブジェクトです。
  - dim_billing_accounts`。
  - dim_crm_accounts`。
  - `usage_ping_mart`

ソリューション・オーナーシップ

- ソース・システム・オーナー
    - バージョンアップしました。`@jeromezng`
    - セールスフォース: `@jbrennan1`.
    - Zuora: `@andrew_murray` (英語)

    - 場所（IPアドレス）：`@m_walker`さん
- ソースシステムのサブジェクト・マター・エキスパート。
    - バージョンアップしました。`@jeromezng`

    - セールスフォース: `@jbrennan1`.
    - Zuora: `@andrew_murray` (英語)
    - 場所（IPアドレス）：`@m_walker`さん
- データチーム・サブジェクト・マター・エキスパート: `@mpeychet_` `@m_walker`

データソース
Product Geolocation Dashboardは、`usage_ping_mart`データモデルを使って構築されています。
  - データは主に使用率のpingデータから得ており、顧客セグメントのフィールドはZuoraとSalesforceから得ています。
  - 分析は、自己管理インスタンスから暦月に受信した最後の使用pingを考慮して標準化されており、`is_last_ping_in_month`として利用できます。これにより、使用状況のメトリクスがインスタンス間で重複しないようになります。
  - Geolocationフィールドは、インスタンスの*ホスト*のIPアドレスに由来するもので、必ずしもインスタンスの物理的な位置を示すものではありません。
  - usage_ping_martを作成する際にどのようなテーブルが考慮されたかを理解するには、Usage Ping Mart Entity Relationship Diagramをご覧ください。
    - ERDを理解することで、異なるデータソースがどのように結合するかを理解し、必要に応じてモデルを修正することができます。

  ### エンティティ・リレーションシップ・ダイアグラム

| ダイアグラム/エンティティー ---------- 穀物| ---------- 目的| キーワード |
|
||| |
| [Usage Ping Mart](https://app.lucidchart.com/documents/view/be5f5dc8-8ad5-4586-af53-93ff5e00f720/0_0)| usage_ping_id｜使用状況のpingと関連する顧客セグメンテーションメトリクスを探索するためのマート｜#### Sisenseのデータディスカバリー機能

Sisenseでさらに探究する
Sisenseは私たちが視覚化のために使っているツールです。ダッシュボードデザイナーはSQLに精通している必要があります。しかし、Sisenseのデータディスカバリー機能を使えば、ドラッグ＆ドロップでチャートを作成することができるので、制限を感じることはありません（SQLクエリは必要ありません）。

<figure class="video_container">。
  <iframe src="https://www.youtube.com/embed/h_b9A8F7Ic8" frameborder="0" allowfullscreen="true">。
</iframe>。
</figure>

詳細はこちら【SisenseのData Discoveryページ】（https://dtdocs.sisense.com/article/data-discovery）。

Sisense Snippets

スニペットは、異なるチャートで再利用可能なSQLコードの文字列を作成するために使用されます。詳しくは、[Sisense SQL Snippetsページ](https://dtdocs.sisense.com/article/snippets)をご覧ください。

usage_pings_mart スニペット

このスニペットは現在、Sisenseに[usage_pings_mart]という名前で存在しています（https://app.periscopedata.com/app/gitlab/snippet/usage_pings_mart/553a4fc6bf004b749eb60a144d722ccc/ 編集）。

```sql
WITH pings AS (

  SELECT *
```

```
FROM analytics.usage_ping_mart
WHERE ping_source = 'Self-
Managed'
  AND is_last_ping_in_month =
  TRUE AND date_id >= 20191101
```

```
    AND [ping_product_tier=product_tier]とする。
    AND [ping_country_name=Usage_Ping_Country].

)
```

リファレンスSQL

1ヶ月間に使用量のpingを送信したアカウントの合計数

```
[use_pings_mart]

SELECT
  ping_monthです。
  COUNT(DISTINCT account_id) AS
total_accounts FROM pings
GROUP BY 1
```

国ごとの使用量pingを送信するインスタンスの月間数

```
[use_pings_mart]

SELECT
  ping_monthです。
  ping_country_        nameAS country_name,
  COUNT(DISTINCT uuid) AS instances_reporting
FROM pings
GROUP BY 1,2
```

データプラットフォームソリューション

データの系統

- dbtモデルの系統図
- ジオロケーションへのIPアドレスのマッピングは、無料のgeolite2Maxmindデータベースから取得しています。
- 位置情報もMaxmindのデータベースから取得していますが、iso3の国コードフィールドを除いては、リポジトリのZuoraCountryCSVから取得しています。

DBTソリューション

- テーブル間の大きな結合や、IPアドレスとジオロケーションのマッピングがless-than/greater-than結合節で構成されることを避けるために、今回のマージリクエストで最初に実装されたように、IPアドレスは他のモデルとは別にジオロケーションに段階的にマッピングされています。
- この方法では、SisenseでIPアドレスを不明瞭にすることができますが、異なるデータベーステーブル間でIPアドレスを照合する機能は維持されます。

トラステッドデータソリューション

トラステッドデータフレームワーク

EDM エンタープライズ ディメンション モデル バリデーション

- エンタープライズ次元モデル検証ダッシュボード

RAWソースデータパイプラインの検証

バージョン

- 信頼できるデータダッシュボード
  - tdfタグを含むすべてのバージョンデータテストを報告します。

ライセンス

- 信頼できるデータダッシュボード
  - tdfタグを含むすべてのライセンスデータテストを報告します。

カスタマーDB

- 信頼できるデータダッシュボード
  - tdfタグを含むすべてのCustomerDBデータテストのレポート。

手動でのデータ検証

- 手動使用のPing検証ダッシュボード

---

layout: handbook-page-toc title:"製品利用データ" 説明"製品利用データは、お客様が製品としてのGitLabをどのように、いつ、どこで利用しているかを定量的に測定したものです。"

このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

製品使用データ

製品使用データは、お客様がいつ、どこで、どのようにしてGitLabを製品として使用しているかを定量的に測定したもので、GitLabチームがより良い製品を作り、お客様の導入を促進し、お客様の維持を改善するために使用します。製品利用データのページでは、GitLabチームのメンバーが製品利用データを調査し、顧客のインサイトを開発するために使用できる情報とツールを提供します。

顧客の製品使用データのビジネスユースケース。

- 顧客のGitLabインスタンスにおけるユーザー導入/オンボーディングの洞察
- 特定のDevOpsステージにおける機能の利用率の増減傾向の監視 アクティブユーザーの減少の監視
- 最新バージョンが遅れているアカウントの特定 デプロ
- イに使用されたインストールの種類の特定 GitLab
- Runnersの数の増減の可視化
- GitLabの実装アーキテクチャの可視化、インフラのスケールアップ/ダウン
- 社内のチャンピオンと利用や採用パターンについて話し合い、新規ユーザーの導入やアカウントの全体的な価値実現のために、どこにエネルギーを投入すべきか戦略を練る

将来的には、以下のようなユースケースへの対応を予定しています。

- 顧客と最終的なチームによるステージの採用をROIと成熟度で追跡する。これは、カスタマーサクセスプランの目標に対する進捗状況の把握にもなります。
- 活用度、顧客ライフサイクルのフェーズ、エンゲージメント（＝Time-to-Value）に基づいて、デジタルエンゲージメント（＝アクションやコンテンツ）を促進する。

製品使用データのソース

ソースキーユースケースソースからEDWへのデータフロー

ソースキー	ユースケース	ソースからEDWへのデータフロー
使用状況Ping	Gainsight 製品使用率、xMAU、推定MAU	セルフマネージドインスタンスから送信されたJSONペイロード -> version.gitlab.com -> VersionPostgresDatabase <- pgp -> snowflake.raw.version_db
シートリンク	Gainsight製品使用状況	顧客ポータル→顧客Postgresデータベース <- pgp -> snowflake.raw.tap_postgres.customers_db_license_seat_links
バージョンチェック	Noneversion -> VersionPostgresDatabase <- pgp -> snowflake.raw.version_db.version_checks	
	ProductAdoptionDashboardです。	
GitLab.com	Gainsight製品の使い方（近日公開	gitlab.com -> replicas/clones <- pgp -> snowflake.raw.tap_postgres

セルフサービスの機能

このデータソリューションは、3つのSelf-Service Data機能を提供します。

1. **Gainsightユーザーの皆様**へセルフマネージド製品の使用状況データがGainsightで利用できるようになり、Gainsightユーザーは特定のワークフローの作成、トレンドの可視化、顧客の健全性スコアカードの作成、ユースケースの採用戦略の検討などが可能になりました。Using Product Usage Data in Gainsight a full guide（製品使用状況データをGainsightで使用する方法）をご覧ください。
2. **ダッシュボード開発者**。新しいダッシュボードを構築し、既存のダッシュボードを顧客製品採用データにリンクさせるための完全な次元モデルコンポーネントを含む新しいSisenseデータモデルです。
3. **SQL開発者**。A EnterpriseDimensionalModel」をご覧ください。`R2A Objects`」タブを参照してください。

データプラットフォームコンポーネント

データプラットフォーム技術の観点からも、このソリューションは有効です。

1. Gainsight Data Pump - EDWからGainsight、GainsightからEDWへ
2. 製品使用状況データのためのEnterprise Dimensional Modelの拡張
3. Data Pipeline Healthダッシュボードのテストとデータ検証の拡張機能
4. ERD、dbtモデル、および関連するプラットフォームコンポーネント

クイックリンク

Gainsightの製品使用状況データ　　カスタマーサクセスでのGainsightの活用

WIPお客様の製品採用ダッシュボード　　WIP製品使用データ - 知識評価

Sisense Discoveryを使い始める　　セルフサービス・ウォークスルー・ビデオ

データセキュリティの分類

Product Usage Dataに含まれるデータやそれを支えるデータの多くは、オレンジまたはイエローです。これには、オレンジ色のアカウントの顧客メタデータ、SalesforceやZuoraのコンタクトデータ、GitLabの非公開財務情報などが含まれますが、これらはすべて公開すべきものではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する際には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。このデータはSnowflakeとSisenseに残すべきであり、特に承認されない限り、これらのアプリケーションを通じてのみ共有されることが理想的です。

ORANGE

- 説明行またはレコードレベルの顧客および個人データ。オブジェ
- クトです。
  - `dim_crm_account`
  - `dim_billing_account`
  - `dim_ip_to_geo`
  - `dim_location`

YELLOW

- 説明します。GitLab 財務データ。集約や合計を含む。オブジェクトです。
-
  - `dim_subscriptions`
  - `prep_recurring_charge`

ソリューション・オーナーシップ

- ソース システム オーナー:
  - 使用法 Ping`@jfarris`
  - セールスフォース`@jbrennan1`
- ソースシステムのサブジェクト・マタ
  - ー・エキスパート：使い方
  - Ping`@jfarris`
  - Gainsight:`jbeaumont` セールス
    フォース`JBRENAN1`
- データチームのサブジェクト・マター・エキスパート`rparker @snalamaru`

主な用語

1. お客様
2. 使用方法 Ping
3. GitLabセルフマネジメント・サブスクリプション
4. GitLab SaaSサブスクリプション
5. シートリンク
6. 製品カテゴリー、製品階層、配送
7. バージョンチェック
8. ビラブルメンバー。API, 定義, EDM フィールド名: `billable_user_count`
9. アクティブなユーザーCustomerDocs, MetricDictionary, EDM フィールド名: `active_user_count`

キーメトリクス、KPI、PI

1. データマート - メトリクスの定義
2. イベントベースのメトリクス
3. ユーザーベースのメトリクス
4. ステージおよびグループのパフォーマンス指標
5. ARR

メトリックフォーマット

1. **ユーザーベースのメトリクス**アクション/イベントを実行した
   - ユーザーの数 過去28日間
   - 過去7日間
     - 例"# of users who completed a merge request in last 28 days"
2. **イベントベースのメトリクス**。実行されたアクション]の数
3. イベント/属性/ユーザー]の**カウント**数を**合計します。**
   1. 例"# of runners" または "# of auto_devops_enabled projects"
4. **指標となるメトリクス**。ある属性が真か偽か
   1. 例"シェアードランナーが有効かどうか"
5. **パワーユーザーのメトリクス**。このアクション、このアクション、そしてこのアクションを行ったユーザーの数

セルフサービス・データ・ソリューション

セルフサービス・ダッシュボード開発者

Sisenseでのチャート構築を始めるには、Sisenseの10分間のDataOnboardingVideoを見るのが良いでしょう。ダッシュボードを作成した後は、そのダッシュボードを簡単に見つけられるようにしたいものです。トピックは、ダッシュボードを1つの場所に整理し、簡単に見つけることができる素晴らしい方法です。ダッシュボードの右上にあるトピックへの追加アイコンをクリックすると、トピックを追加できます。ダッシュボードは、関連性のある複数のトピックに追加することができます。トピックには、Finance（財務）、Marketing（マーケティング）、Sales（販売）、Product（製品）、Engineering（エンジニアリング）、Growth（成長）などがあります。

セルフサービスのSQL開発者

キーフィールドとビジネスロジック

- GitLab SaaSとGitLab Self-Managedの顧客デプロイメントから得られた製品使用状況データは、定期的にエンタープライズデータウェアハウスに投入され、GainsightとSalesforceで利用されます。
- 私たちはUsage Pingを利用して、自己管理された顧客の利用データを導き出します。自己管理された顧客の製品利用データは、自己完結型のUsage Pingペイロードに大部分が含まれています。
- SaaS顧客製品利用データは、ソースデータベースのテーブルを使用して再構築されます。
- シートリンクデータは、自営かSaaSかに関わらず、すべてのお客様のライセンス使用率データを含みます。集約
- されたメトリクスは 7 日間と 28 日間の期間で収集され、Usage Ping ペイロードのトップレベルキーである counts_weekly と counts_monthly のサブキーに aggregated_metrics が追加されます。
- すべてのタイムフレームの集約されたメトリクスは、名前に aggregate_ のプレフィックスが追加された count トップレベル
- キーに存在します。Gainsightの消費量の基礎となるテーブルは、Self-Managedの料金プランに関連付けられたすべてのZuoraサブスクリプションのセットに基づいて構築されています。このテーブルに含まれる事実のセットを構築するために、Customers DBからのSeat Linkデータが優先度の高いUsage Pingメトリクスと組み合わされています。各月の最新の受信および最新のUsage Ping（指定されたサブスクリプションIDのcreated_at dateによる）およびSeat Link（dim_subscription_idによる）ペイロードが報告されます。

エンティティ・リレーションシップ・ダイアグラム

			Diagram/EntityGrainPurposeKeywords
製品使用データERD	以下のすべて	カラム名、カラムのデータタイプ、カラムの制約、主キー、外部キー、テーブル間の関係など、すべてのテーブル構造を表示します。	顧客、使用状況Ping、サブスクリプション、シートリンク、セルフマネージド、SaaS、プロダクト、デリバリー、アカウント

リファレンスSQL

スニペットライブラリ説明

データプラットフォームソリューション

Gainsightデータポンプ

データは、GitLab SaaSおよびGitLab Self-Managedの顧客デプロイメントから得られたものです。

EDWからGainsight Data Pumpへ。

データチームはGainsightのネイティブ機能を利用してSnowflake Enterprise Data Warehouseからデータを読み込みました。データチームは、Gainsightがアクセスできる読み取り専用のマートレベルのテーブルを構築し、そこには現在利用可能なすべてのデータが含まれています。時間の経過とともにデータチームがメトリクスや顧客セグメントを追加すると、このテーブルは自動的に追加データで更新されます。この「インターフェース」を`Gainsight Data Pump`と呼びます。

GainsightからSnowflake Data Pipelineへ。

データチームは、GainsightからSnowflakeへの新しいソースデータパイプラインを開発し、新しいカスタムオブジェクトやGainsightで作成されたデータを含めることで、使用pingの一致率を高めるなどの改善を行いました。

ProductUsagedatadevelopmentementalStreams」の図は、Self-ManagedおよびSaaS Product UsageをGainsightに提供するための開発アプローチを示しています。

データの系統

dbtソリューションは、Source ModelsからGainsightまでの各テーブルを介したデータの流れを表す次元モデルを生成します。

- dbtモデルの完全なデータ系統は、DataFlowDiagramに掲載されています。

トラステッドデータソリューション

TrustedDataFrameworkの概要を見る

詳細なテストの実施方法については、dbtガイドの例を参照してください。

製品使用状況 信頼性のあるデータ ダッシュボード

dbtテスト、ソースモデルの新しさ、レコード数、最終実行日、ゴールデンレコードの検証などを表示する詳細なダッシュボードです。最新のEnterprise Dimensional Modelのテストと実行に関するレポートです。

(WIP) 製品使用状況 信頼性の高いデータダッシュボード

RAWソースデータパイプラインの検証

データパイプライン・ヘルスバリデーション

---

layout: handbook-page-toc title:"SaaS製品イベントデータ" 説明"製品利用データは、有料・無料ユーザーがGitLabを製品としてどのように、いつ、どこで利用しているかを定量的に測定するものです。"

{::options parse_block_html="true" /}。

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

SaaS製品のイベントデータ

SaaS製品イベント・データ・パイプラインは、ビジネス・ユーザー（プロダクト・アナリスト、成長担当PM、コア・プロダクトPMなど）に、当社のSaaS製品（Gitlab.com）のユーザーと名前空間の活動を記録したファクト・テーブルとマート・テーブルのセットを提供します。

このテーブルの汎用性により、ビジネス関係者は以下の例のように様々な分析を行うことができます。

実行可能な分析の種類

1. ユーザージャーニー分析。同じネームスペースで、異なる製品ステージがどれだけ頻繁に使用されているか。どのステージが組み合わせて使用されているかを確認します。
2. ネームスペース・ステージの採用。新しいネームスペースが使用開始後数日で「Create」や「Verify」などのステージをどの程度採用しているかを評価する。
3. 組織ごとのステージ。ネームスペースが最初の数日間でどのようにステージを採用し、それが有料コンバージョンや長期的なエンゲージメントとどのように相関しているかを確認します。

ERDとデータフロー、テーブルの記述

データチームは、これらのファクトテーブルを管理しており、これを利用して、:

- fct_event_400: イベントレベルの粒度でのファクトテーブル。つまり、このテーブルではイベントごとに1つの行が見つかります。イベントとは、例えば、issueの作成やci_pipelineの開始などです。fct_daily_event_400: 近日中に実装予定の
- テーブルで、進捗状況はこのイシューで確認できます。
- fct_daily_xmau_400: 近日中に実装される予定のテーブル、進捗状況はこの課題で確認できます。

これらのモデルは、prep_eventという名前のdbtモデルを介して作成されます。

prep_eventモデルでは、月ごとに分割されたテーブルを作成します。Snowflakeを見ると、月ごとに1つのスキーマがあります。

つまり、1ヶ月に1つのprep_eventテーブルがあるということです。dbtでは、スキーマの定義が
dbt_project.yml.

クエリの中には、データを月ごとのサブセットに限定するWHERE文もあります。以下は、

prep_eventテーブルのERDです。

これらのテーブルで現在追跡されているイベントのリストは、ここで入手できます。いくつかの新しいイベントは非常に簡単に追加できます。

このテーブルに新しいイベントを追加するには？

テーブル全体の複雑さはprep_eventテーブルに含まれる（dbtドキュメントはこちら

このモデルは少し長く、多くの異なるテーブルからのデータを集約/統合しています。とはいえ、新しいイベントを追加するのは、割と簡単な操作です。以下にその手順を示します。

Gitlabデータチームのワークフロー

すべては課題から始まるので、まず最初のステップとして、Data Projectに課題とMRを開設します。MRで作成したブランチに入ったら、以下の手順で作業を進めてください。

データを格納するCTEの追加

テーブルに含めたいデータが準備中のテーブルに（フィルタリングの必要なく）完全に取り込まれている場合、これはかなり簡単に実行できるアクションです。ここで示したシンプルなCTEマクロで、新しいタプルを追加するだけでよいのです。

```
{{ simple_cte([
    ('prep_ci_pipeline', 'prep_ci_pipeline'),
    ('prep_action', 'prep_action'),
    ('prep_ci_build', 'prep_ci_build'),
    ('prep_deployment', 'prep_deployment'),
```

```
    ('prep_issue', 'prep_issue'),
    ('prep_merge_request', 'prep_merge_request'),
    ('prep_note', 'prep_note'),
    ('dim_project', 'dim_project'),
    ('dim_namespace', 'dim_namespace'),
    ('prep_user', 'prep_user')
])}}
```

取り込みたいデータが既存のテーブルのサブセットである場合は、操作が少し複雑になります。

1. まず、上記のようにsimple_cteマクロに、キャプチャしたいイベントに対応するテーブルを追加します。
2. 次に、最初のアクションで呼び出したテーブルの行をフィルタリングするための別のCTEを以下のように作成します。

例えば、成功したCIパイプラインだけを選択したいとします。このイベントはprep_ci_buildで取得されています。
テーブルで、failure_reasonカラムでフィルタリングする必要があります。

まず、prep_ci_build テーブルを simple_cte マクロに追加します（まだ追加されていない場合）。次に、

failure_reason列でフィルタリングする新しいCTEを追加する必要があります。

```
successful_ci_pipelines AS (

    SELECT * )
    FROM prep_ci_pipeline
    WHERE failure_reason IS NULL

)
```

JSONにエントリを追加する event_ctes

JSONはこのようになります。

```
  {
    「event_name": イベントの名前。issue_creationなどのアクション名をつけることをお勧めします。
    "source_cte_name": 必要なデータを含むCTE名で、先に定義したものです。 "user_column_name": ユーザーID
    を含むカラム。
この情報を得るためには、dbtを参照する必要があります。
    「project_column_name": プロジェクトのIDを含むカラム、この情報を得るためにはdbtを参照する必要があり
ます。
    "primary_key": CTEのプライマリキー
  }
```

MRのテスト

変更をテストする最も簡単な方法は、CIパイプラインの機能を使って行うことです。最初のステップは、本番のデータベースをクローンすることです（以下の例）。これには10分から15分かかります。

そして、以下の小さなビデオに示すように、dbtジョブを実行す

る必要があります。 

dbt job specify_l_modelを⚙をクリックして実行し、以下の変数を追加する。

- キーを使用しています。DBT_MODELS
- 値：プリップ_イベント

ジョブが無事に実行されたら、R&Dチームのメンバーにレビューを依頼します。

MRに関する質問は、Slackの#dataチャンネルやMerge Requestを通じて、データチームのメンバーに遠慮なく質問してください。

データの埋め戻し

データが統合されたら、prep_eventデータをバックフィルする必要があります。そのためには、データエンジニアにt_prep_dotcom_usage_events_backfillというAirflow Dagを起動してもらい、過去3年分のデータのバックフィルを実行してください。

---

layout: handbook-page-toc title:"SaaS Service Ping Automation" このページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

サービスPingの概要

以前は`Usage Ping`と呼ばれていたサービスPingは、GitLabインスタンスで毎週実行されるバックグラウンドプロセスで、分析目的に有用な一連のメトリクスを収集、集約、パッケージ化する役割を担っています。メトリクスは簡単に拡張することができ、新しいメトリクスはメトリクス・ライフサイクルに従って定期的に変更されます。メトリクスの完全なセットは、Metric Dictionaryで定義されています。インスタンスからメトリクスが収集されると、それらはJSONペイロード（「ping」）にまとめられ、GitLab Versions Appに投稿され、Snowflakeなどのダウンストリームプロセスに同期されます。以下は、ServicePingペイロードの例です。サービスPingには、セルフマネージドサービスPingとSaaSサービスPingという2つの主要なバリエーションがあります。セルフマネージドサービスPingはシングルテナント型のGitLabに対して実行され、SaaSサービスPingはマルチテナント型のGitLab.comに対して実行されます。

サービスPingの使用例

サービスPingのデータは、GitLabがどのように使われているかをプロダクト、サポート、セールスの各チームが理解するためのインサイトを提供します。例えば、このデータは以下のことに役立ちます。

1. サポート GitLabxMAUKPIs KPI
2. ステージの月間アクティブユーザー数（SMAU）を算出することで、ステージや機能の成功を測ることができます。
3. どの機能が使われているのか、あるいは使われていないのかを理解し、顧客がGitLabの豊富な機能を活用できるようにガイダンスを提供する。
4. 月ごと（または週ごと）にカウントを比較し、インスタンスが異なる製品機能をどのように使用しているかを大まかに把握することができます。
5. GitLabインストールの分類と理解に役立つその他の事実の収集

セルフマネージドサービスPing

セルフマネージドのお客様は、サービスPingをセットアップして実行し、GitLabの独自のデプロイメント（インスタンス）の分析を行います。お客様はオプションでサービスPingを無効にすることができ、その場合はメトリクスがGitLabに送信されません。お客様は、コンソールアプリケーションを通じてサービスPingのデータにアクセスできます。

SaaSサービスのPing

GitLab.com（またはGitLab SaaS）は、基本的にGitLabがホストするマルチテナント版のGitLabです。手動で生成されたサービスPing（Manual SaaS Service Ping）がSaaS用に実装されており、Self-Managedインスタンスで実現しているものと同等の分析カバーをSaaSでも提供しています。

しかし、現在のプロセスには、大きく分けて2つの問題があります。

パフォーマンスの問題

- プロセスにエラーが発生しやすく、お客様のライブ活動とリソースの競合が発生
-

する。 プロセスに時間がかかり、個々の指標の問い合わせが定期的に失敗する。

- プロセスはオフピーク時に手動で実行されるように実装されてい
- る プロセスはエンドツーエンドで管理するためのスタッフを必要とする

データカバレッジの不足

また、手動のSaaSサービスPingは、インスタンスレベル（サイト全体）のデータしか生成できないため、セールスやカスタマーサクセスなど、GitLab.comのお客様の個々の採用状況を測定するためにネームスペースレベルのより詳細なデータを必要とする人たちのニーズをすべて満たすものではありません。

この2つの主要な問題を解決するために、データチームは「Automated SaaS Service Ping」を開発しています。Automated SaaS Service Pingは、ビッグデータ、自動化、スケールのために設計されたシステムであるEnterprise Data Platformでネイティブに実行されるプログラムのセットです。Automated SaaS Service Pingが完全に運用されると、Manual SaaS Service Pingは廃止されます。Automated SaaS Service Pingを構成する主なサブプログラムは2つあります。

- **SaaS Instance Service Ping** - GitLab.comインスタンスのサービスPingを毎週自動生成。
- **SaaS Namespace Service Ping** - すべてのGitLab.comインスタンス->名前空間のサービスPingを毎週自動生成。

4種類のサービス pingプロセスの実行環境 3種類の環境

サービスPingには、運用中のものと開発中のものがあり、全部で4種類あります。

	1. セルフマネージドサービスPing	**2. マニュアルSaaSサービスPing**	**3. Automated SaaS Instance Service Ping**	**4. 自動化されたSaaS名前空間サービスのPing**
Where Run	環境1：お客様のセルフマネージドインスタンスの場合	環境2：GitLab.comのインフラ内	環境3：データプラットフォーム基盤	環境3：データプラットフォーム基盤
Run リジェンス	ByGitLab (Automatically)	プロダクト・インテ (手動)	エアフロー(自動)	エアフロー(自動)
				頻度週刊
誌週刊誌週刊誌コード		オーナープロダクト	インテリジェンスプロダクトインテ	
リジェンスデータチームデータチーム				
	ソースコードRuby,	SQLRuby,	SQLPython, SQL,	dbtPython, SQL, dbt
データ Granularity グラニュラ スタンス・ネームスペース				InstanceInstanceNamespace（ リティ・インスタンス・イン
開発 ステータス	実写化実写化		実写化実写化	開発 中

(自動）SaaSサービスのPing実施

プロセス概要

(自動化された）SaaSサービスPingは、Airflowでオーケストレーションされた**Python**プログラムとdbtプロセスのコレクションで、Enterprise Data Platform内で毎週実行されるようにスケジュールされています。自動化されたSaaSサービスPingプロジェクトは、すべてのソースコードと構成ファイルを保存します。このプログラムは、2つの主要なデータソースに依存しています。Redisベースのカウンターと**SQL**ベースのpostgresテーブルです。この2つのデータソースは、Snowflakeの自動データパイプラインとして実装されており、SaaS Service Pingの実装プロセスとは独立して実行されるようになっています。

- SaaSからのSQLベースのpostgresデータは、pgp経由で同期され、`RAW.SAAS_USAGE_PING`スキーマで利用できるよう
- になります。 Redisデータはプログラムの実行時にアクセスされ、`RAW.SAAS_USAGE_PING`スキーマにも保存されます

。

自動化されたSaaSサービスPingは、2つの主要なデータ処理フェーズで構成されています。

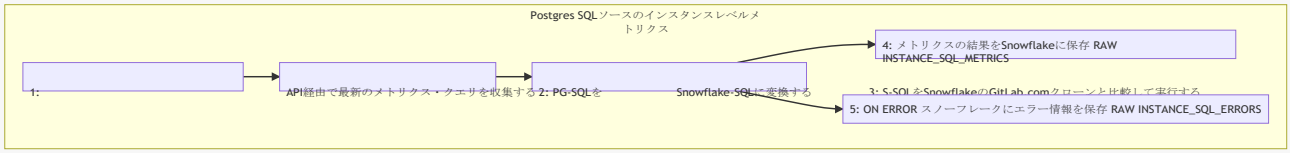
1. **フェーズ1**では、MetricDictionaryで定義されているメトリクスの収集とジェンター化を行います。
2. **フェーズ2**では、メトリクスをTrusted Data Model（FCTとDIMテーブル）形式に変換します。

フェーズ1：測定基準の収集と生成

SaaS Instance Service Ping

SaaS Instance Service Pingは、「プロセスの概要」で説明したとおりに実行されます。

インスタンスのSQLベースのデータフロー



データフローの詳細については、ServicepingのREADME.mdファイルをご覧ください。

Redisベースのデータフロー

SaaS 名前空間サービス Ping

SaaS Namespace Service Ping は、SaaS Instance Service Ping よりも細かいレベルのメトリクスを生成します。このプロセスでは、GitLab.com のすべてのネームスペースのリストにアクセスし、各ネームスペースをループして最後親のネームスペースレベルの使用率メトリクスを生成します。名前空間テーブルはプログラムへの入力となり、効率化のために、従来の1対1の名前空間ループではなく、SQL SET OPERATIONとSQL `GROUP BY` namespace-idが使用されています。**最終的なメトリクスの出力は、最終的な親のネームスペース・レベルに格納されます**。名前空間サービスPingの欠点は、現在のところSQLソースのメトリクスしか利用できず、`analytics_unique_visits.g_analytics_contribution`のようなRedisソースのメトリクスは現在利用できないことです。

メトリクス収集・生成プロセス 擬似コード

1. `GitLab.com`の`Postgres`ソースデータのパイプラインが稼働しており、Snowflakeでは新鮮な最新のデータがそれぞれ、`RAW.SAAS_USAGE_PING`スキーマと`PREP.SAAS_USAGE_PING`スキーマです。
2. サービスPingのパイソンプログラムの開始
    1. **SQLベースの**メトリクス生成の開始
        1. PostgresのSQLソースのメトリクスを開始

    1. 最新のPostgres SQL-sourced (PG-SQL) メトリクスクエリのセットをMetrics Dictionary API Query Endpointから取得します。
    2. pythonトランスフォーマーを使用して、インスタンスレベルのPG-SQLをSnowflake SQL（S-SQL）に変換します。
    3. Snowflakeデータウェアハウスで利用可能なSaaS GitLab.comクローンデータに対してS-SQLを実行し、その結果を`RAW.SAAS_USAGE_PING.INSTANCE_SQL_METRICS`に格納します。エラーが発生した場合、データは`RAW.SAAS_USAGE_PING.INSTANCE_SQL_ERRORS`テーブルに格納されます。
  2. **Redisベースの**メトリクスの開始
    1. Redis APIの呼び出し
    2. データはJSON形式でピックアップして保存されます。おおよそのサイズは2k行程度で、通常は1回のロードで1つのファイルになります（現時点では週1回のロードです）。データは`RAW.SAAS_USAGE_PING.INSTANCE_REDIS_METRICS`に保存されます。
2. **名前空間レベルの**メトリクス生成開始
  1. NamespaceQueriesJSONから最新のメトリクスクエリを取得
  2. Snowflakeデータウェアハウスで利用可能なSaaS GitLab.comクローンデータに対して名前空間クエリを実行し、結果を`RAW.SAAS_USAGE_PING.GITLAB_DOTCOM_NAMESPACE`に格納する。

フェーズ2：メトリクスの信頼性の高いデータモデルへの変換

すべてのソースメトリクスがSnowflake `RAW.SAAS_USAGE_PING`スキーマで利用可能になると、データをTrusted Data Modelフォーマットに変換するためのdbt処理を開始します。

- **SQL**ベースのメトリクスのdbt処理
- **Redis**ベースのメトリクスのdbt処理
- **名前空間**レベルのdbt処理

既知の制限事項／改善事項

- 名前空間レベルのRedisソース・メトリクスはまだ利用できません。
- Snowflakeには冗長な "レガシー "サービスpingプロセスがあり、これを廃止する必要があります。

サービスPingメトリクスの種類

Service Pingでは、主に2種類のメトリクスをサポートしていま

- す。SQLメトリクス：`Postgres`テーブルから取得されたメ

  トリクス
- Redis metrics: `Redis`ベースのカウンターから取得したメトリクス

SQL Metricsの実装

SQLベースのメトリクスのワークフローは、最も複雑なフローです。SQLベースのメトリクスは、実際には、インスタンスのPostgres SQLデータベースに対して実行されるSQLクエリによって作成されます。大規模なテーブルの場合、これらのクエリの実行には非常に時間がかかります。例えば、`Counts.ci_builds` メトリックは、最大級の ci_builds に対して `COUNT(*)` を実行しています (10 億行以上の dbt テーブルを参照)。このモジュールの目的は、SaaSインスタンスのpostgres SQLデータベースの代わりに、Snowflakeデータベースに対して実行することです。

プロダクト・インテリジェンス・チームは、メトリクスを計算するために実行するすべてのSQLクエリを取得できるAPIエンドポイントを作成しました。以下はそのファイル例です。

APIエンドポイントに関するテクニカルドキュメントはこちらか

らどうぞ。 それでは、JSONレスポンスで受け取ったいくつかの

```
 "counts "です。{
    "assignee_lists"。"SELECT COUNT(〃¯˘-˽¯˷) FROM 〃¯˷ WHERE 〃¯˘-˽¯˷"
3",
    "boards""SELECT COUNT(〃¯˘-˽¯˷) FROM 〃¯˷"
    "ci_builds"。"SELECT COUNT(øøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøø)FROM"
= 'Ci::Build'"となります。
    "ci_internal_pipelines"。"SELECT COUNT(ci_pipelines\.id\) FROM ` ci_pipelines\ WHERE
(ci_pipelines\.source\ IN (1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13) OR"
「i.s.................................NULL)
    "ci_external_pipelines"。"SELECT COUNT(CI_PIPELINES\.ID\) FROM ʻCI_PIPELINES\ WHERE"
「source\」=「6」です。
    "ci_pipeline_config_auto_devops "です。"SELECT COUNT(ci_pipelines\.id\) FROM  ci_pipelines\ WHERE
ci_pipelines\.config_source\ = 2"。
    "ci_pipeline_config_repository "です。"SELECT COUNT(ci_pipelines\.id\) FROM ` ci_pipelines\ WHERE `
ci_pipelines\.config_source\ = 1 "です。
    "ci_runners":"SELECT
    COUNT(øøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøøø
    øøøøøøøøøøøøøøøøøø)", "ci_triggers""SELECT COUNT(\"ci_triggers\".\"id\") FROM
    \"ci_triggers\"",
```

そのためには、Snowflakeのテーブル *(GitLab Saasから同期されたもの)* に対して実行できるようにすることが目標です。その

ためには、PostgresのSQLテーブルと同じカラム名、同じ粒度のテーブルを用意する必要があります。

現在、SnowflakeでPostgresのデータを変換している様子は以下の通りです。

 ALT

重複を避け、妥協のない正確なデータを提供するためには、デデューブ・レイヤーが有効です。

次に、SQLベースのメトリクス・クエリを実行できるテーブルを特定しました。次に、これらのテーブルに対してクエリを実行するために、SQLステートメントを変換する必要があります。

このSQL文を変換するスクリプトを実行しています。

```
"SELECT COUNT('ci_builds\.'id\) FROM
   'ci_builds\"
  WHERE "ci_builds"."type" = 'Ci::Build'"となります。
```

をこのSQL文に追加します。

```
"SELECT 'counts.ci_builds      AS counter_name,
        COUNT(ci_builds.id)   AS counter_value,
        TO_DATE(CURRENT_DATE) AS run_day
   FROM prep.gitlab_dotcom.gitlab_dotcom_ci_builds_edupe_source AS ci_builds WHERE
  ci_builds.type = 'Ci::Build'"
```

そして、これらのクエリをすべて実行し、その結果をjsonに格納して、それをテーブルの`raw.saas_usage_ping.instance_sql_metrics`.このテーブルには以下の列があります。

- `query_map`: すべてのクエリを格納する
- run`_results`: 結果を格納する `ping_date`:
- クエリが実行された日付 `run_id`: 処理の一
- 意な識別子

エラーが発生した場合、データは`RAW.SAAS_USAGE_PING.INSTANCE_SQL_ERRORS`テーブルに保存されます。このテーブルは、データを`json`データタイプで保持します。このテーブルの上には、エラーを`Trusted Data Model`に取り込み、`SQL`処理の不具合を簡単に可視化する仕組みが構築されています。

Redisメトリクスの実装

Redis カウンターは、GitLab で起きている任意の状況のうち、データベースに永久的な記録が残らない高頻度の発生を記録するために使われます。このような場合、バックエンドエンジニアは、例えば navigation_sidebar_opened のように状況を表す名前を定義し、それが発生する瞬間を（既存の実行経路に専用のコードを追加することで）任意に決定します。

プロダクト・インテリジェンス・チームは、データ・チームがいつでも好きな時にすべてのRedisメトリクス値を取得できるAPIエンドポイントを作成しました。JSON レスポンスの例は UsageDataNonSqlMetricsAPI のページにあります。なお、`-3` はメトリクスが Redis でないことを意味しており、API はその値を取得しません。JSONレスポンスを受信すると、Snowflakeの`RAW.SAAS_USAGE_PING.INSTANCE_REDIS_METRICS`テーブルに保存されます。APIエンドポイントに関する追加の技術文書はこちらをご覧ください。このテーブルには、以下の列があります。

- `jsontext`: 実行されたすべてのクエリを
- 格納するもの `ping_date`: クエリが実行
- された日付 `run_id`: 処理の一意な識別子

エアフローの設定

毎週土曜日に実行されるAirflow dag `saas-instance-usage-ping`を作成し、以下のすべての操作を実行しました。API Endpoint

- からのクエリの取得

- クエリを変換してSnowflakeのデデューープ層で実行できるようにする クエリの
- 実行
- 結果をSnowflakeに保存

RAWからPRODデータベースとSisenseへ

現在は、RAWで保存されたデータの変換を限定的に行っています。将来的には、このようなデータフローになる予定です。

つまり、作成されたデータセットは、モデル`prep_usage_data_flattened`で現在のデータパイプラインとUNIONされます。

---

layout: handbook-page-toc title:"Sales Funnel" こ

のページについて

- TOC {:toc}

{::options parse_block_html="true" /}。

セールスファネル

セールスファネルは、ギットラボのLead to Cashプロセスの中核をなすもので、潜在顧客からの問い合わせを、獲得した機会のクローズまで追跡します。このプロセスでは、顧客はリードから、マーケティング適格リード、営業許可されたオポチュニティ、そして最終的にはクローズしたオポチュニティ、または失われたオポチュニティへと移行します。このハンドブックページでは、GitLabチームメンバーがセールスファネル全体を分析し、インサイトを生み出すための分析ツールやリソースを提供しています。

このページの目的は

- TD: Sales Funnel - Targets vs. Actualダッシュボードとその派生ダッシュボードのナビゲーション方法の理解を助ける。TD:
- Sales Funnel - Targets vs. Actuals Dashboardの作成に使用されたデータモデルの理解に役立ちます。
- GitLabでのあなたの役割に最も適した認定を受けることで、あなたの理解度を評価してください。
  - ダッシュボードの使い方を詳しく知りたい方は、ダッシュボードのユーザー認定を受けてください。
  - Sisenseダッシュボードの開発について詳しく知りたい方は、DashboardDeveloper認定を受けてください。
- そして、全体的にみんなが貢献できるように。

Release Train Cadence:

1. 要件収集期限：Sales Key Meetingの1週間後。
2. UAT用にリリースされたのは、Sales Key Meetingの1週間前。

メンテナンスの予定です。

1.必要に応じて、2週間に1度、金曜日の午前9時から午前11時（米国東部時間）に定期メンテナンスを実施します。

クイックリンク

TD: Sales Funnel - Targets vs. Actuals ダッシュボード

セールスファネルの次元モデルとダッシュボードのトレーニング

TD：Sales Funnel Management View - Preloaded Cuts

TD：スタンダードブッキングのカット

Sisense Discoveryの使用開始について

はじめに

そのためには、お客様に理解していただく必要があります。

- このダッシュボードでサポートされるKPI/PIは？
- メトリクスをどのように説明するかを説明するキーワー
- ド ダッシュボードを支えるデータソース
- さらに探求するために、Sisenseでビジュアルや分析を自分で作成することができます。Sisenseディスカバリー・ツールを使用するのが良い方法です。Sisenseを始めたい方はこちらをご覧ください。
- さらに深く掘り下げるために、snowflakeでデータを探索することができます。Snowflakeでの探索の利点は、追加の情報（つまり他のデータソース）に結合できることです。Snowflakeでの探索に関する追加情報はこちらをご覧ください。

キーターム、メトリクス、KPI/PI、そしてキーフィールドとビジネスロジック

▶ 主な用語

- 注文タイプ
- **販売資格のあるソース。**機会がどのように作られたか**購入チャ**
- **ネル。**アカウントが購入した方法。GTM戦略
- **フォーカスアカウント。**アカウントベースドマーケティングチームが、セールスやフィールドマーケティングと密接に連携して、特にターゲットとするアカウントを表します。GTM戦略の価値である「アカウント・セントリック」、「アカウント・ベース・ネット・ニュー」、「アカウント・ベース・エクスパンション」のうち、「フォーカス・アカウント」に相当します。
- マーケティングチャネル（イニシャルソース
- **セールスヒエラルキー・ライブ**販売エリア > 販売地域 > 所在地地域 > 販売セグメント
- **Sales Hierarchy Stamped:**セールスエリア > セールスリージョン > ロケーションリー

▶ ジョン > セールスセグメント Key Metrics, KPIs, and PIs

- **Marketing Qualified Lead（MQL）の略。**販売機会を得ることができる適格なリードの数。現在、MQLは一度だけですが、将来的には、コンタクトを育成する中で、MQLは複数回認定される可能性があります。MQLとは、人口統計学的、企業統計学的、および行動学的情報に基づいて、一定の基準に達したリードのことで、当社では累積100ポイントとしています。MQLスコア」は、様々な行動やプロフィールデータで構成されており、プラスまたはマイナスのポイント値で重み付けされています。パーソンスコアが更新されるたびに、LeanDataはそのレコードをフローで処理する必要があるかどうかのチェックを実行します。
- **Sales Accepted Opportunity（SAO）の略。**購入権限を持つ見込み客の担当者とのミーティングが設定されているオポチュニティ、またはGitLabがソリューションとなりうるイニシアチブ、またはユースケースや潜在的なシート数、今後60日以内の次のステップを含む適合性を持つオポチュニティです。この案件は、SDRが作成した営業担当者によって検証され
- 、受理されています。**SAO/MQLです。**1ヶ月間にSAOに変換されたすべてのMQLのスナップショット変換率。1ヶ月間のSAO数を1ヶ月間のMQL数で割ったもの。
- **新しいロゴ。**獲得したファーストオーダーアカウントの数。
- **Closed Won IACV (Net_ARR):**クローズドな獲得案件の予約収入。
- **クローズドウォンディール。**収益に結びついた機会の数。**Win Rate（勝率）。**解決
- に至った案件のうち、期間中に何件の案件を獲得したかを示す。**ASP on New First**
- **Order Deals:**初回受注案件の平均案件サイズ。
- **IACV (Net_ARR) Created:**各日に作成されたパイプラインの値です。パイプラインは時間の経過とともに変化します。これは、作成日に基づく現在のパイプラインです。
- **SAOからClosed Won Dealまでのサイクルタイム：**Closed Won案件の結論を得るまでの平均的な時間。
- **SAOからClosed Wonへのコンバージョン率。**月間ベースでSAOがClosed Wonの案件に転換したコンバージョン率。
- **SAOから獲得済みクローズド案件への転換率：SAOから獲得済みクローズド案件への転換率。SAOの**数：SAOがクローズド案件の結論に達するまでの時間を基準にしたSAOの数。

▶ キーフィールドとビジネスロジック

データソースとデータモデルの理解

TD: Sales Funnel - Target vs. Actual Dashboard**は、Sisenseの3つの主要なマートビュー、**[mart_crm_opportunity]**、**[mart_crm_person]**、**[mart_sales_funnel_target]**から生成されたレポートビューから作成されます。Sisenseのビューでは、シンプルなSQLの文字列を、必要なだけのチャートで参照することができます。これにより、エンドユーザーは新規ロゴ数やSAO(Sales Accepted Opportunity)数などの計算の裏にあるロジックを理解することができ、また、どの結合が可能かを理解するための推測作業を省くことができます。

独自のダッシュボードを作成するには、Sisenseで利用できる以下のビューを参照します。

- [**mart_crm_opportunity**] - 該当するファクトとディメンションを結合して、セールスファネルのオポチュニティセクションのビューを取得するマートテーブル。
- [**mart_crm_person**] - 該当するファクトとディメンションを結合し、セールスファネルのマーケティング適格リードセクションのビューを得るためのマートテーブル。
- [**mart_sales_funnel_target**] - 該当するファクトとディメンションを結合して、セールスファネルターゲットのビューを取得するマートテーブル。
- [**rpt_crm_opportunity_closed_period] - [mart_crm_opportunity]**ビューをクローズド期間にピボットし、有用な日付のアグリゲーションフィールドを生成するレポートです。
- [**rpt_crm_opportunity_accepted_period] - [mart_crm_opportunity]**ビューを受理された期間にピボットし、有用な日付のアグリゲーションフィールドを生成するレポートです。
- [**rpt_crm_opportunity_created_period] - [mart_crm_opportunity]**ビューを作成期間でピボットし、有用な日付のアグリゲーションフィールドを生成するレポートです。
- [**rpt_sales_funnel_target] - [mart_sales_funnel_target]** ビューをターゲット期間にピボットし、有用な日付のアグリゲーションフィールドを生成するレポートです。
- **[rpt_crm_person_mql**] - MQLのカウントに使用される**[mart_crm_person]** WHERE is_mql = 1をフィルタリングするレポート。

このビューでは、ERD（Entity Relationship Diagram）に見られるようなデータモデルを考慮しています。

▶ データリネージの
▶ クエリ例

-- MQLsのカウント SELECT DATE_TRUNC('month',mql_date_first) AS mql_month, COUNT(mql_date_first_id) AS actual_mqls FROM "PROD". "COMMON_MART_MARKETING". "MART_CRM_PERSON" WHERE is_mql = 1 GROUP BY 1 ORDER BY 1 DESC

--SAOのカウント SELECT DATE_TRUNC('month',sales_accepted_date) AS sao_month, COUNT(*) AS actual_saos
FROM "PROD". "COMMON_MART_SALES". "MART_CRM_OPPORTUNITY"WHERE is_sao = TRUE GROUP BY 1 ORDER BY 1
DESC

```
</details> </details
<br>

<style> #headerformat {
background-color:5px; text-align: center;} #6666c4; color: black; padding:5px; text-align: center;}。
</style>
<h1 id="headerformat">セールスファネル標準レポート</h1>
<br>

<詳細
<summary markdown='span'>
  Sales Funnel Management View - Preloaded Cuts
</summary>

TD: Sales Funnel Management View - Preloaded Cuts]
(https://app.periscopedata.com/app/gitlab/828239/TD:-Sales-Funnel-Management-View---Preloaded-Cuts) ダッシュボードは、セールスファネルから主要な9つのKPIを追跡します。

1.ネットARR
1.新しいロゴマーク
1.パイプラインの作成
1.セールス・アクセプタンス（SAO
1.マーケティング・クオリファイド・リード（MQL）について
1.トライアル
1.平均販売価格（ASP
1.MQLからSAOへ
1.勝率

KPIは、四半期ごとのペースでそれぞれの目標値と比較され、異なる次元で切り分けられます。これらの切り口により、経営陣はビジネスが異なる次元でどのように実行されているか、高レベルの概要を素早く把握することができます。これは、ビジネスのどの分野が軌道に乗っていて、どの分野にもっと注意を払う必要があるかという疑問に答えるものです。

現在、販売管理のカットは以下のスニペットで動いています（使用方法はスニペットの説明欄に記載されています）。

1.[main_qtd_view_sales_funnel]
(https://app.periscopedata.com/app/gitlab/snippet/main_qtd_view_sales_funnel/55d49d17d8cf4cc6bf976c6
1da2d0017/edit)
1.[ratio_qtd_view_sales_funnel]
(https://app.periscopedata.com/app/gitlab/snippet/ratio_qtd_view_sales_funnel/55d5211e130f45a29a3a15
62307f95d9/edit)

販売管理pythonモジュール】(https://gitlab.com/gitlab-)
data/periscope/-tree/periscope/master/custom_modules/sales_management）のスニペットと一緒に、BI層でレポートを視覚化するために使用されます。

</details> </details

<詳細
<summary markdown=span>
  Standard Bookings Cuts
</summary>

TD: Standard Bookings Cuts](https://app.periscopedata.com/app/gitlab/831911/TD:-Standard- Bookings-
Cuts) ダッシュボードでは、予約済みネットARRの前年比（Year over Year, YY）や、四半期および年間の財務目標に対するパフォーマンスをさまざまな角度から追跡します。

各カットは、「四半期ビュー」と「年度ビュー」で構成されています。

ダッシュボードを使用するには、フィルターで会計年度を選択し、また
ドリルダウン "フィルタです。後者は、「Sales Segment Drilldown」フィールドの粒度を選択します。
```

全体	大規模-パブセックの内訳	大規模-地域の内訳	中間市場の内訳	大規模-地域の内訳
		USイースト	USイースト
		USウェスト	USウェスト
		EMEA	EMEA
		APAC	APAC
	大きい PubSec	大きい PubSec	大きい PubSec
大規模＋PubSec		大規模＋PubSec	大規模＋PubSec
			ファーストオーダーチーム
			テリトリー（500未満
ミッド-マーケット	ミッド-マーケット	ミッド-マーケット	ミッド-マーケット
SMB	SMB	SMB	SMB
WW	WW	WW	WW

キーフィールドとビジネスロジック

* [ATR (Available To Renew)](https://about.gitlab.com/handbook/sales/sales-term-glossary/#available-to-renew-atr)
* ATR：特定のカテゴリーの正味ARR／小計ATR。例：セグメントがSMB、成長タイプがContractionの場合、%ATRを計算するには、`SMBのNet ARR | Contraction` / `SMBのATR`となります。
* TRX：チャンスの数
* %MIX(ARR)。特定のカテゴリーの純ARR/小計ARR。例）セグメントがSMBで、Deal Sizeが5-25kの場合の%MIX(ARR)を計算する場合。SMBのネットARR | 5-25k` / SMBのARR`。
* 予約数の割合特定のカテゴリーの純ARR / 特定の四半期の総ARR
* 総予約数に対する割合特定のカテゴリーの純ARR / 特定のカテゴリーの総ARR 小計
* ProServ #: proserv_amountが0と異なるオポチュニティの数
* [A R (Attach Rate)](https://about.gitlab.com/handbook/sales/performance-indicators/#proserve-deal-and-dollar-attach-Rate)
* 売上セグメントのドリルダウンフィールドに表示される「米国東」の行は、「東」と「東」の両方で構成されています。LATAM」地域
* 販売セグメントのドリルダウン = APACは、「East」、「West」、「LATAM」、「EMEA」以外のすべての地域で構成されます。
* セールス・セグメント・ドリルダウン＝SMBは、セグメントSMBに加えて、すべての大規模」と「中規模」について
* Channel Type = Fulfilled」は、「Channel Type Fulfilled」に「NULL/Missing Channel Type」を加えたものです。

</details> </details>

<style> #headerformat {
background-color:6666c4; color: black; padding:5px; text-align: center;
}
</style>
<h1 id="headerformat">追加のリソース</h1>。

<詳細
<summary markdown='span'>
トラステッドデータソリューション
</summary>

セールスファネルモデルでは、Trusted Dataのテストとその結果に`sales_funnel`タグを使用しています。これは[Trusted Data Dashboard](https://app.periscopedata.com/app/gitlab/756199/Trusted-Data-Dashboard)で最も簡単に見ることができます。

概要は[Trusted Data Framework](https://about.gitlab.com/handbook/business-technology/data- team/platform/#tdf)をご覧ください。

[dbtガイドの例](https://about.gitlab.com/handbook/business-technology/data-team/platform/dbt-guide/#trusted-data-framework)を参照してください。

さらなるテストの実施に関する詳細と例
</details> </details

<詳細
<summary markdown='span'>
EDM エンタープライズ ディメンション モデル バリデーション

```
</summary>
(WIP) Enterprise Dimensional Model Validation Dashboard]
(https://app.periscopedata.com/app/gitlab/760445/WIP:-Enterprise-Dimensional-Model-Validation-
Dashboard)は、最新のEnterprise Dimensional Modelのテストと実行をレポートします。
</details> </details>

<詳細
<summary markdown='span'>
  RAWソースデータパイプラインの検証
</summary>
[Data Pipeline Health Validations](https://app.periscopedata.com/app/gitlab/715938/Data-Pipeline-
Health-Dashboard)
</details> </details>

<詳細
<summary markdown='span'>
  データセキュリティの分類
</summary>
セールスファネル分析のデータの多くは、[オレンジ] (/handbook/engineering/security/data-classification-
standard.html#orange)または[イエロー] (/handbook/engineering/security/data-classification-
standard.html#yellow)です。これには、ORANGEのアカウントの顧客メタデータや、GitLabのNon publicの財務情報
が含まれており、これらはすべて一般に公開されるべきではありません。このダッシュボードからデータを共有する際に
は、その詳細が GitLab の組織内に留まるようにし、外部に共有する場合には適切な承認を得られるように注意しなけ
ればなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを
保存しないよう、常に注意を払う必要があります。これらのデータは、[Snowflake](/handbook/business-
technology/data-team/platform/#data-warehouse)と[Sisense](/handbook/business-technology/data-
team/platform/periscope/)に残すべきであり、特に承認されない限り、これらのアプリケーションを通じてのみ共有さ
れることが理想的です。

**ORANGE**。

- 説明行またはレコードレベルの顧客および個人データ。
- オブジェクトです。
  - dim_crm_person`
  - dim_crm_account`。

YELLOW**（イエロー

- 説明します。GitLabの財務データで、集約や合計を含みます。
- オブジェクトです。
  - `fct_crm_person`
  - `fct_crm_opportunity` の略。
</details> </details>

<詳細
<summary markdown='span'>
  ソリューションの所有権
</summary>
* ソース・システム・オーナー
  * セールスフォース: `@jbrennan1`.
* ソースシステムのサブジェクト・マター・エキスパート。
  * セールスフォース: `@jbrennan1`.
* データチーム・サブジェクト・マター・エキスパート：`@paul_armstrong` `@jeanpeguero` `@jjstark` `@iweeks`
</details> </details>

---
レイアウト：ハンドブック-ページ-トック
のタイトルである。"Trusted Data Solutions Dashboard"
---

## On this page
{:.no_toc}
```

```
- TOC
```

```
{:toc}

---
Coming Soon!
---
layout: handbook-page-toc
title:"ウェブトラフィック分析"
---

## On this page
{:.no_toc}

- TOC
{:toc}

---

# # Google Analyticsによるウェブトラフィック解析

ウェブトラフィック分析は、訪問者やウェブサイト上での行動についての洞察を得て、より良い結果をもたらすのに役立ちます。BigQueryとGoogle Analyticsは、ウェブサイトを改善するために必要な情報を提供し、ウェブサイトを最高のものにしてくれます。

**以下のサブドメインが含まれます:**。
- about.gitlab.com _(マーケティング分析のためのプライマリ)_。
- docs.gitlab.com
- forum.gitlab.com
- customers.gitlab.com
- learn.gitlab.com

### Googleアナリティクスデータの解析になぜBigQueryデータが使われるのか？
- サンプリングデータがなくなる
- ディメンション数は無制限
- 異なるスコープを1つのレポートにまとめる_(例：セッションとヒット数)_。
- カスタムチャンネルグルーピングの構築とデータエラーの修正
- Google Analyticsのデータをサードパーティのデータソースと組み合わせる_(例：Salesforceのトライアルキャンペーンメンバーとトライアルページビュー)_(例：Salesforceのトライアルキャンペーンメンバーとトライアルページビュー)

### Quick Links
- Googleアナリティクスの基礎を学ぶ。[アナリティクス アカデミー オンライン講座】（https://analytics.google.com/analytics/academy/）
- BigQueryデータのフォーマットとスキーマを理解する。[BigQuery Export Schema - Columns] (https://support.google.com/analytics/answer/3437719?hl=en)を参照してください。
- GitLab公開サイトへのGoogleアナリティクス導入情報 [Marketing Web Analytics - Inbound Mktg](https://about.gitlab.com/handbook/marketing/growth-marketing/inbound-marketing/analytics/#dimensions-vs-metrics)
- BigQueryデータを利用したSaaS Trial Sisenseのダッシュボード。[SaaS Trial Dashboard] (https://app.periscopedata.com/app/gitlab/697554/SaaS-Trial-Dashboard)

### Data Caveats and Constraints
- ユーザー数（Nbr of users）は、BigQueryとGoogle Analytics 360 UIの間で常に約0.1〜0.2%の相違があります。これは、Google Analyticsではユーザー数が概算で表示され、BigQueryでは正確に表示されるためです。
- ヒットレベルのデータ（例：ページビュー）を報告する場合は、次のようにリンクする必要があります。Google_analytics_360.session_hit` テーブルを `analytics.ga360_session_xf` に変更し、session_date でフィルタリングできるようにしました。これは現在、ヒットレベルのデータでは利用できないためです。

### Data Security Classification
```

Google Analyticsでは、クライアントIDや訪問者IDなどの固有の識別子が保存されますが、これには **変更** の分類が必要です。

ソリューション・オーナーシップ

- ソースシステムオーナー：`@shanerice`さん
- マーケティング戦略とパフォーマンスのサブジェクト・マター・エキスパート: `@vjuhasz`.
- Data Team Subject Matter Expert: `@paul_armstrong`

`@jeanpeguero` ### Key Terms

- **セッション**（別名：訪問）：一定の時間内に行われるユーザーのウェブサイトとのやり取りのこと。例えば、1つのセッションには、複数のページビュー、イベント、ソーシャルインタラクションを含めることができます。セッションは午前0時に失効します。また、GitLabのサブドメインでは、一定時間の活動がないと、**60分**で失効します。
- **ページビュー**：Google Analyticsのトラッキングコードが挿入されたページをユーザーが閲覧した回数です。これはすべてのページビューを対象としています。したがって、ユーザーがページを更新したり、ページからナビゲートして戻ってきたりした場合、これらはすべて追加のページビューとしてカウントされます。
- **ユニーク・ページビュー**。ユニークページビュー数は、同一ユーザーが同一セッション中に複数回閲覧したページを1回だけカウントすることで算出されます。
- **新規訪問者**。新規訪問者/ユーザーとは、指定された期間以前にサイトを訪れたことのないユーザーのことです。
- **リターニング・ビジター**:リターニングビジター/ユーザーは、以前にサイトの少なくとも1つのページに少なくとも1回の訪問をしています。これは、Google Analyticsが以前の訪問を示すクッキーを検出できるかどうかによって決まります。Googleがクッキーを検出できない場合、ユーザーが個人のブラウザ設定でクッキーを無効にしていない限り、今後の記録のためにクッキーが設定されます。
- **直帰率**：ユーザーがどのようにサイトにアクセスしたか、またどのくらいの時間そのページに滞在したかに関わらず、1ページだけでサイトを離れた訪問数をパーセントで表したものです。
- **ヒット**：ページビュー、イベント、トランザクションを含む、訪問中のインタラクション。
- イベント**：イベントは、トラッキングコードでは記録されないアクティビティを測定するのに役立ちます。ユーザーが動画を再生したとき、CTAをクリックしたときなどがこれに該当します。
- **カスタムディメンション**：ブログ記事のタグ、ブログ記事のタイプ（ブログ記事の作成者）など、追加の非標準的なデータをGoogle Analyticsに記録することができます。

Key Metrics, KPIs, and PIs

- **セッション**：SisenseのBigQueryウェブトラフィックデータをGoogle Analytics UIで適切にクエリしてマッチさせるためには、ミッドナイトスプリットをオーバーライドするセッションIDを作成する必要があります。
`CONCAT(visitor_id, CAST(visit_start_time AS STRING)`)
- **Users**：ユーザーの計算は、visitor_id `COUNT(DISTINCT visitor_id)`に基づいて行う必要があります。Google Analyticsでは、ユーザーを計算する際にすべての行を考慮します（セッションとは異なります）。
- **新規ユーザー**：ユーザーの計算は、訪問者のIDに基づいて行う必要があります `COUNT(DISTINCT visitor_id)` where total_new_visits is not null`.
- **ページビュー**：この指標はヒットレベルのデータに基づいており、すべての行が `COUNT(1)` where `HIT_TYPE = 'PAGE'` とカウントされる必要があります。
- **ユニークなページビュー**：この指標もヒットレベルのデータに基づいており、セッション内のユニークなページビューをカウントするためには、セッションIDを連結する必要があります（詳細は _Sessions KPI_) with `PAGE_PATH` where `HIT_TYPE = 'PAGE'`.全ページの結果は、Google AnalyticsのUIと比較して、約-0.2%の差がある場合があります。
- **バウンスレート**。近日公開予定

キーフィールドとビジネスロジック
- session_date`: 日付を含むすべてのレポートは、`session_date`に基づいていなければなりません。
ビジットスタートタイム
- Visitor_id`: セッションIDの作成とユーザー数の算出に使用されます。
- total_visits`: セッションの数です。この値は、インタラクションイベントがあるセッションでは `1` です。セッション内にインタラクションイベントがない場合は、`null`となります。Google Analyticsではインタラクティブなセッションのみが表示されるため、nbr of sessionsを検索する際にはこの点を考慮する必要があります。
- `hit_type`: ヒットの種類で、`PAGE`または`EVENT`のいずれかになります。
- `page_path`:ページのURLパス、_e.g:/フリートライアル/_
- ホスト名`: これは、異なるサブドメイン_(about.gitlab.com、docs.gitlab.comなど)でフィルタリングするために使用されます。

スニペット

近日公開予定

リファレンスSQL

以下のSQLクエリは、Google Analytics UIの結果と一致する各主要指標の計算を定義しています（All Data - Raw No Filtersビュー）。

2020年9月の日別セッション数、ユーザー数、新規ユーザー数**。

```sql
SELECT
  DATE_TRUNC('day',session_date)::date
AS visit_date,
  COUNT(DISTINCT IFF(total_visits=1, CONCAT(visitor_id, CAST(visit_start_time AS STRING)), NULL)AS
interactive_Sessions,
  COUNT(DISTINCT visitor_id)
AS users,
  COUNT(DISTINCT IFF(total_new_visits IS NOT NULL, visitor_id,
NULL))AS new_users
from legacy.ga360_session_xf
WHERE DATE_TRUNC('day',session_date)::date >= '09/01/2020'
  AND DATE_TRUNC('day',session_date)::date <= '09/30/2020'
GROUP BY 1
ORDER BY 1
```

2020年2月以降の日別総ページビュー数およびトライアルページビュー数

```
このサブクエリーは、一意のセッションIDごとにsession_dateを取得し、それをリンクさせるために必要です。
ヒットレベル表へ
WITH ga_unique_sessions AS (

SELECT
 DISTINCT(CONCAT(visitor_id, CAST(visit_start_time AS STRING)))AS session_id,
 SESSION_DATE
from legacy.ga360_session_xf

総ページビュー数とトライアルページビュー数
), ga_pageview_hits AS (

SELECT
 CONCAT(visitor_id, CAST(visit_start_time AS                    AS
 STRING)) page_path,                                            session_id,
 CASE
  WHEN page_path = '/free-
    trial/' THEN COUNT(1)
  else 0
 end                                                            AS trial_pageviews,
 count(1)                                                       AS total_pageviews
from legacy.ga360_session_hit
where hit_type = 'page'
GROUP BY 1,2

)、ファイナルAS(

SELECT
 SESSION_                                                       DATEAS visit_date,
 SUM(trial_pageviews)                                           AS trial_pageviews,
 SUM(total_pageviews)                                           AS total_pageviews,
 COUNT(DISTINCT ga_pageview_hits.session_id,page_path )          AS unique_pageviews
FROM ga_pageview_hits
LEFT JOIN ga_unique_sessions
 ON ga_pageview_hits.session_id=ga_unique_sessions.session_id
WHERE DATE_TRUNC('day',SESSION_DATE)::date >= '2/1/2020'
GROUP BY 1
```

```
)

SELECT *
FROM final
```

エンティティ・リレーションシップ・ダイアグラム

近日公開

データプラットフォームソリューション

データの系統

Google Analytics 360/BigQueryのデータは、ロサンゼルスのタイムゾーン(GTM-08:00)を使用している「All Data - Raw No Filters」ビューに基づいています。Google Analytics 360/BigQueryのデータは、Sisenseでは2020年2月19日から利用可能です。これらは、Google Analytics 360のUIでクエリ結果を比較する際に知っておく必要があります。

Google Analytics BigQueryの各行は、1つのセッションを表しています。Google AnalyticsのUIには多くのディメンションやメトリクスがあるため、このスキーマはデータベースレベルでは少し圧倒されるかもしれません。下の図は2つのセッションを表しており、BigQueryのデータ構造をよりよく理解するのに役立ちます。

image{: .shadow}。

BigQueryのエクスポートスキーマについては、以下のインタラクティブツリーマップを参照して

ください。BigQuery

ソース

Webトラフィック分析を行う上で重要となる**BigQueryのテーブルは大きく分けて3つ**あります。

1. analytics.ga360_session_xf: セッション数、ユーザー数、ユニークビジター数の算出に使用されるセッションレベルのデータ。
2. google_analytics_360.session_hit: ページビュー、イベントなどを計算するためのヒットレベルデータ。
3. ga360_session_custom_dimension_xf：GitLab固有のディメンションのレポート用（これは、各カスタムディメンションに定義されたスコープに依存します。

DBTソリューション

近日公開

トラステッドデータソリューション

近日公開

EDM エンタープライズ ディメンション モデル バリデーション

近日公開

RAWソースデータパイプラインの検証

近日公開

---

layout: handbook-page-toc title:"xMAU分析" 説明"xMAUは、月間アクティブ使用量（MAU）を把握するための様々なレベルを把握するための単一の用語です。"

このページについて

{:.no_toc}

- TOC {:toc}

xMAU分析

xMAUは、月間アクティブユーザー数（MAU）を把握するための様々なレベルを表す単一の用語であり、アクション（AMAU）、グループ（GMAU）、ステージ（SMAU）、トータルCMAUを含みます。製品グループに有用な単一の指標を提供し、製品全体の重要業績評価指標にうまくマッピングするために、xMAUの各指標は上述の順序で上に向かって連鎖していきます。

xMAUメトリクスは、Usage Ping（セルフマネージドインスタンスレベルの粒度）とGitLab.com（SaaSネームスペースレベルの粒度）から得られます。このアナリティクス・ワークフローは、様々な顧客セグメントにおける各レベルのxMAUメトリクスの分析を可能にし、報告、推定、予測メトリクスの報告の基礎となります。

このページの目的は

- Product Adoption Dashboardの操作方法の理解 Product Adoption Dashboard
- に使用されるデータモデルの理解
- GitLabでのあなたの役割に最も適した認定を受けて、理解度を確認してください。ダッシュボ
  - ード・ソリューションへの理解度を測るには、ダッシュボード・ユーザー認定を受けて
  - ください。ダッシュボード開発者認定資格は、今後取得する予定です。
  - さらに、推定xMAUを記録するための方法を理解することをお勧めします。
-

クイックリンク

製品採用ダッシュボード　SiSense xMAU入門 ダッシュボード例 推定アルゴリズムのページ

はじめに

1. 重要な用語、測定基準、KPI/PIの理解
2. データモデルの理解

主な用語

- **アカウント**
- **ホスト イン**
- **スタンス**
- **Instance User Count** - インスタンス上のユーザーの合計数
- **有料ユーザ**
- **ー 製品階層**
- **使用量 Ping**
- **バージョン**

キーメトリクス、KPI、PI

以下の指標の説明は、Product Team Performance Indicatorのページにあります。

- **Action Monthly Active Users (AMAU)**
- **Stage Monthly Active Users (SMAU)**
- **セクションの月間アクティブユーザー数（セクショ**
- **ンMAU） セクションの総月間アクティブユーザー数（**
- **セクションCMAU） 複合月間アクティブユーザー数（**
  CMAU

各指標には、3つの異なるバージョン（記録、推定、予測）があり、「製品チームのパ

- フォーマンス指標」のページで説明されています。

- the Sisense Style Guide 現在、使用量を特定した記録されたpingメトリクスは、Centralized Dashboardにチャート化されていますが、今回のEstimated valuesの最初のバージョンに取り組んでいます。

xMAUの算出方法は？

xMAUは主にサービスPingデータソースによって計算されます。プロジェクトの開始時、ステージとグループのプロダクトマネージャーは、xMAUチャートを作成する特定のサービスPingメトリックを選択しました。

- 使用されたGMAUメトリクスのリスト
- 使用されたSMAUメトリクスのリスト

現在、xMAUメトリクスのSSOTはこのスプレッドシートで、Sheetload経由でデータハウスにインポートされています。つまり、特定のメトリクスのGMAU、SMAU列を更新すると、その変更はハンドブックで更新されたxMAUチャートに伝搬するということになります。

このSSOTをこのスプレッドシートから辞書のYAMLファイルに移行する計画がありますが、その作業は本号で行います。

メトリクスの定義についてさらにご質問がある場合は、プロダクト・インテリジェンス・チームにお尋ねください。彼らは現在、ここで利用可能なメトリクス辞書を管理しています。また、メトリクスがデータベースの計算である場合は、メトリクスの値を生成するために実行したSQLクエリを提供することができます。

日付の範囲

すべてのインスタンス（自己管理型とSaas/GitLab.com）について、その月に生成された最後のpingを使用してxMAUを計算しています。インスタンスには、サービスのpingを生成する曜日がランダムに割り当てられますが、その割り当ては時間の経過とともに変化します。例えば、あるインスタンスが火曜日にPingを生成するように割り当てられた場合、そのインスタンスは常に火曜日にPingを生成します。pingが生成される曜日はインスタンスごとに異なるため、28日カウンタに記録される正確な日付範囲も異なります。

現在、「今月の最後のping」は、週の初め（月曜日）の日付で決定されています。(SQLでは`DATE_TRUNC('month', ping_created_at_week))`を使用しています。そのため、「その月の最後のping」が技術的には測定月に起こらないこともあります。ここでは、2021年8月の例を紹介します。

Ping 作成日	Ping作成週間	2021年8月のxMAUに使用されています。
2021-08-05	2021-08-02	FALSE
2021-08-12	2021-08-09	FALSE
2021-08-19	2021-08-16	FALSE
2021-08-26	2021-08-23	FALSE
2021-09-02	2021-08-30	TRUE

xMAU」と「有料xMAU」の違い

有料xMAUの定義

上記の各指標は、xMAUと有料xMAUについて計算されます。有料xMAUは現在、「28日間のローリング期間中に、Usage Pingデータを介してセルフマネージドの有料インスタンスにロールアップしたユーザー、*または*GitLab.com Postgres Database Importsを介してSaaSの有料ネームスペースにロールアップしたユーザー」を月間アクティブユーザーと定義しています。(例として「PaidStageMonthlyActiveUsers-PaidSMAU」を参照してください)。

現在、GitLab ビジネス機能は、どの名前空間やインスタンスが OSS、EDU、内部プロジェクト、または有料プランタイプを持つが ARR を貢献しないその他のサブスクリプションに属するかを識別する標準的な方法を持っていないため、xMAU の現在の実装では、これらのサブスクリプションに関連するユーザーを「有料」として含めています。

xMAUおよびPaid xMAUの算出について

xMAUとpaid xMAUを計算するために、バージョンアプリとGitlab.comのPostgresデータベースの2つの主なデータソースがあります。下の表は、これらの計算に使用されるデータソースをまとめたものです。

デリバリー	xMAU	有料xMAU
SaaS	バージョンアップのお知らせ	Gitlab.com Postgres Table**。
セルフマネジメント	バージョンアプリ	バージョンアップしました。

メモ

*:SaaSのxMAUでは、gitlab.com用に生成されたペイロードを使用しています。これらのペイロードは、uuid = `ea8bf810-1d6f-4a6a-b4fd-93e8cbd8b57f` のインスタンスにリンクされているので、簡単に識別できます。

**:SaaSの有料XMAUでは、Gitlab.comのpostgresテーブルを使用する必要があります。Gitlab.com インスタンスから生成された Usage Ping ペイロードは、インスタンスレベルでのハイレベルな統計情報を提供します。これは集約された数値であり、例えば製品の階層、プランの種類、ネームスペースなど、さらに細かく分類することはできません。

- SaaSの有料xMAUを生成するためには、データウェアハウスに格納されているGitlab.comデータベースのレプリカにxMAUカウンターを複製する必要があります。
  - バッチカウンタは、シンプルなSQLで生成されたカウンタです。カウンタの生成に使用されたSQLクエリはアクセス可能で、簡単に再現できます。
  - 残念ながら、これはすべてのカウンターに対してできることではありません。Redisのカウンタは、SQLで生成されたカウンタではありません。Redisカウンタは、ページビューやフロントエンドのインタラクションなど、Postgres DBにはないアクションも追跡します。
- そのため、Gitlab.com の Postgres レプリカを使って再現できるのは一部のメトリクスだけです。つまり、現時点では Monitor StageのようなsaaS PaidのxMAUメトリクスの一部を計算することができません。

***:セルフマネージの有償 xMAU を計算するために、使用量Pingペイロードのエディション・フィールドを使用し、`EEP`、`EES`、および `EEU`エディションの使用量 Ping のみを選択します。エディションの値はライセンスが生成された時点で、licenseDotデータベースのライセンステーブルのプラン列から得られ。つまり、現在はEDU/OSSのサブスクリプションを有料のxMAUの計算から除外していません。

データソース

上記の説明に基づいて、私たちが使用している2つの主要なデータソース

- があります。サービスPingデータソース
- Gitlab.comのデータソース

データマート

私たちは、ユーザーがさまざまな製品データソースを探索できる一連のデータマートを構築しました。

マートサービス Ping使用量 Pingデータ

`mart_service_ping_product_usage_data`は、すべてのService Pingデータのための最も包括的なデータモデルです。このデータ・モデルは、Service Pingデータを、サブスクリプション、CRM Account...などの財務およびGTMデータ・ソースと結合します。

これにより、ここで定義されているすべての月別およびすべての時間別指標の使用データを取得することができます。

マート推定xMAU

mart `mart_estimated_xmau`は、推定XMAU PIを簡単に生成するために構築されています。このモデルには、このページで説明されているすべての推定ロジックが含まれています。

エンドユーザーは、非常にシンプルなチャートを使って、xMAUの推定値を作成することができます。

```
SELECT
Report_month,
Product_tier,
SUM(推定月次指標値_sum)
FROM common_mart_product.mart_estimated_xmau
WHERE xmau_level = 'SMAU' AND stage_name =
'create' GROUP BY 1,2```。
```

mart_paid_estimated_xmauデータマートが作成され、特別に支払われたxMAUのチャートが作成されました。

サービスPingデータパイプライン

プロダクトマネージャーのためのSisense Snippets

プロダクトマネージャーのために作成されたスニペットをすべて網羅した別のページを作成しました。このトピックについては、このページをお読みになることをお勧めします。

また、これらのスニペットをどのように使用するかの例を満載したダッシュボード📊も作成しました。

これらのスニペットを使って、特定のxMAUチャートのターゲットを更新するには？

ダイナミック・ターゲット・バリューの導入

Usage Ping Metrics の SiSense チャートに Monthly Target Value を実装することができるようになりました。これを実現するには、パフォーマンス指標(xMAU、またはUsage Pingから得られる他のPI)のあるymlファイルを更新し、以下に説明する例に従う必要があります。

```
----
- name: Configure:Configure - Adoption PI - GitLab Managed Terraform Stateを持つプロジェクトの数
  base_path:"/handbook/product/performance-indicators/ops-section-performance-indicators/"
  definition:のGitLab Managed Terraform Stateを使用するプロジェクト数のローリングカウントです。
過去28日間
  目標： FY22.1Q末までに6000人
  ターゲット名：Projects_with_terraform_states 月
  間記録されたターゲット。
    "2021-04-20":6000
    "2020-11-01":3000
    "2020-07-01":700
  毎月の推定目標値"2021-05-
    01":6000
    "2020-11-01":3000
    "2020-07-01":700
  org:オペレーションセクション
```

ターゲットを正しく埋めるには

ymlセクションに追加される日付は、各日付間隔の終了日です。つまり、例えば以下の例では

```
毎月の推定目標値
"2021-05-01":6000
"2020-11-01":3000
"2020-07-01":700
```

そして、次のような目標を掲げます。

- start_dateから2020-07-01までは700 2020-08-01
- から2020-11-01までは3000
- 2020-12-01から2020-05-01まで、目標は6000

そのため、yml定義の中に2つの異なるセクションがあり、それを使用することができます。

▶ 月間記録目標

PIが推定PIの場合（xMAUのチャートなど）。ダイナミックターゲットを使ってチャートを可視化するには、このスニペットを使用する必要があります。このスニペットの詳細については、こちらをご覧ください。
できることの例

▶ 月間推定目標値

ターゲットを設定したいPIが記録された値に基づいている場合（推定値を含まない）。ダイナミックターゲットを設定したチャートを表示するには、次のようなコードが必要です。

このスニペットの詳細はこちらをご覧ください 何ができるかの例

スニペットを使って静的なターゲットを取得する

当社の標準化されたスニペットとビジュアライゼーションを使用したすべての埋め込みPI/xMAUチャートでは、ターゲットの更新は非常に簡単な4ステップのプロセスです。

1. 興味のあるPIにアクセスし、sisenseのリンク  をクリックします。
2. チャート  」をクリックします。
3. 最後の値を変更する(これが目標値です。10進数でやれば一定の割合で増えます。完全な数値を入力すると、その数値が目標ラインになります。) 
4. 保存ボタンをクリックすると、 

エンティティ・リレーションシップ・ダイアグラム

私たちの目標のひとつは、分析に必要なすべてのデータを簡単に提供できる1モデルを作ることです。しかし、ソリューションの開発を続けていると、このモデルでは得られない情報が出てくることがあります。ここで、エンティティ・リレーションシップ・ダイアグラムの理解が役立ちます。このモデルは、アクセスしているレイヤーを作成するためにどのテーブルが結合されているかを示しています。このモデルは、より深く掘り下げて、さらなる洞察を得ようとするときに役立ちます。

		Diagram/EntityGrainPurpose
ファクト 月間使用量Pingデータ ERDとデータフロー	host_id, instance_id, reporting_month, metrics_path	さまざまなインスタンスおよびサブスクリプションのディメンション別に、機能の使用状況を把握することができます。

▶ **データの分類**

xMAU分析をサポートするデータの中には、[オレンジ](/handbook/engineering/security/data-classification-standard.html#orange)または[イエロー](/handbook/engineering/security/data-classification-standard.html#yellow)に分類されるものがあります。これには、アカウントからのORANGEの顧客メタデータ、SalesforceやZuoraからのコンタクトデータ、GitLabのNon公開の財務情報などが含まれ、これらはすべて公開されるべきではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する際には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。これらのデータは、[Snowflake](/handbook/business-ops/data-team/platform/#data-warehouse)と[Sisense](/handbook/business-ops/data-team/platform/periscope/)に保存され、特に承認されない限り、これらのアプリケーションを通じてのみ共有されることが理想的です。

ORANGE

- **説明行またはレコードレベルの顧客および個人データ。オブジェク**
- **トです。**
  - `dim_billing_accounts`
  -

```
dim_crm_accounts
```

- ユーズドピングマート

▶ ### ソリューション・オーナーシップ

セルフサービス・ダッシュボード・ソリューション

	ダッシュボード目的
エグゼクティブ・オーバービュー - TBD	このダッシュボードは、すべてのメトリクスの現在のステータスをエグゼクティブ・オーバービューとして表示します。
Dev Section Analysis - TBD	このダッシュボードでは、関連するメトリクスの現状をセクションごとに紹介しています。
DRAFT: ハンドブック更新のための一元化されたダッシュボード	このダッシュボードのチャートは、ハンドブックのエンベッドに使用されます。

トラステッドデータソリューション

トラステッドデータフレームワーク

手動でのデータ検証

- 手動での使用方法**PingValidationDashboard**
- **Version Trusted Data Dashboard**

{:toc}

---

xMAU分析

xMAUは、月間アクティブユーザー数（**MAU**）を把握するための様々なレベルを表す単一の用語であり、アクション（**AMAU**）、グループ（**GMAU**）、ステージ（**SMAU**）、トータル**CMAU**を含みます。製品グループに有用な単一の指標を提供し、製品全体の重要業績評価指標にうまくマッピングするために、**xMAU**の各指標は上述の順序で上に向かって連鎖していきます。

xMAUメトリクスは、**Usage Ping**（セルフマネージドインスタンスレベルの粒度）と**GitLab.com**（**SaaS**ネームスペースレベルの粒度）から得られます。このアナリティクス・ワークフローは、様々な顧客セグメントにおける各レベルの**xMAU**メトリクスの分析を可能にし、報告、推定、予測メトリクスの報告の基礎となります。

このページの目的は

- **Product Adoption Dashboard**の操作方法の理解 **Product Adoption Dashboard**に
- 使用されるデータモデルの理解
- **GitLab**でのあなたの役割に最も適した認定を受けて、理解度を確認してください。ダッシュボード・
  - ソリューションへの理解度を測るには、ダッシュボード・ユーザー認定を受けてください。ダッシ
  - ュボード開発者認定資格は、今後取得する予定です。
  - さらに、推定**xMAU**を記録するための方法を理解することをお勧めします。
-

クイックリンク

製品採用ダッシュボード	**SiSense**を始めよう	**xMAU**のダッシュボード例	推定アルゴリズムのページ

はじめに

1. 重要な用語、測定基準、KPI/PIの理解
2. データモデルの理解

主な用語

- アカウント
- ホスト イン
- スタンス
- **Instance User Count** - インスタンス上のユーザーの合計数 **Paid**
- **User**
- **製品階層 使**
- **用状況 Ping**
- バージョン

キーメトリクス、KPI、PI

以下の指標の説明は、「**Product Team Performance Indicator**」のページにあります。アクション月間

- アクティブユーザー数（**AMAU**
- ステージ・マンスリー・アクティブ・ユーザー（**SMAU**
- セクション 月間アクティブユーザー数（セクション
- **MAU**） セクション合計月間アクティブユーザー数（セ
- クション**CMAU**） 合計月間アクティブユーザー数（
  CMAU

各指標には**3**つの異なるバージョン（記録、推定、予測）があり、その説明は

- **Product Team Performance Indicator**ページ
- **the Sisense Style Guide** 現在、使用量を特定した記録された**ping**メトリクスは、**Centralized Dashboard**にチャート表示されていますが、今回の**Estimated values**の最初のバージョンに取り組んでいます。

xMAU」と「有料xMAU」の違い

有料xMAUの定義

上記の各指標は、**xMAU**と有料**xMAU**について計算されます。有料**xMAU**は現在、「**28**日間のローリング期間中に、**Usage Ping**データを介してセルフマネージドの有料インスタンスにロールアップしたユーザー、*または***GitLab.com Postgres Database Imports**を介して**SaaS**の有料ネームスペースにロールアップしたユーザー」を月間アクティブユーザーと定義しています。**(**例として「**PaidStageMonthlyActiveUsers-PaidSMAU**」を参照してください**)**。

現在、**GitLab** ビジネス機能は、どの名前空間やインスタンスが **OSS**、**EDU**、内部プロジェクト、または有料プランタイプを持つが **ARR** を貢献しないその他のサブスクリプションに属するかを識別する標準的な方法を持っていないため、**xMAU** の現在の実装では、これらのサブスクリプションに関連するユーザーを「有料」として含めています。

xMAUおよび**Paid xMAU**の算出について

xMAUと**paid xMAU**を計算するための主なデータソースは、**Version App**と**GitLab.com**の**Postgres**データベースの**2**つです。下の表は、これらの計算にどちらのデータソースが使われているかをまとめたものです。

デリバリー	**xMAU**	有料**xMAU**
SaaS	バージョンアップのお知らせ	**GitLab.com Postgres Table****。
セルフマネジメント	バージョンアプリ	バージョンアップしました。

メモ

***:SaaS**の**xMAU**では、**gitlab.com**用に生成されたペイロードを使用しています。これらのペイロードは、**uuid** = `ea8bf810-1d6f-4a6a-b4fd-93e8cbd8b57f` のインスタンスにリンクされているので、簡単に識別できます。

**:SaaSの有料XMAUでは、GitLab.comのpostgresテーブルを使用する必要があります。GitLab.com インスタンスから生成された Usage Ping ペイロードは、インスタンスレベルでのハイレベルな統計情報を提供します。これは集約された数値であり、製品の階層やプランの種類、ネームスペースなど、さらに細かく分類することはできません。

- SaaSの有料xMAUを生成するためには、データウェアハウスに格納されているGitLab.comデータベースのレプリカにxMAUカウンターを複製する必要があります。
  - バッチカウンタは、シンプルなSQLで生成されたカウンタです。カウンタの生成に使用されたSQLクエリはアクセス可能で、簡単に再現できます。
  - 残念ながら、これはすべてのカウンターに対してできることではありません。Redisのカウンタは、SQLで生成されたカウンタではありません。Redisカウンタは、ページビューやフロントエンドのインタラクションなど、Postgres DBにはないアクションも追跡します。
- そのため、GitLab.comのPostgres Replicaを使って再現できるのは一部のメトリクスのみです。つまり、今のところ、Monitor StageのようなSaaS Paid xMAUメトリクスの一部を計算することはできません。

***:セルフマネージの有償 xMAU を計算するために、使用量Pingペイロードのエディション・フィールドを使用し、EEP、EES、およびEEUエディションの使用量 Ping のみを選択します。エディションの値はライセンスが生成され時点で、licenseDotデータベースのライセンステーブルのプラン列から得られ

データモデルとスニペット

2種類のデータソリューションを作成しました。

- xmau_202011は、PMが自分のXMAU指標を素早くグラフ化し、グループや個人のダッシュボードに表示したり、ダッシュボードに埋め込んだりすることができるスニペットです。詳細はこちら
- mart_monthly_product_usageテーブルです。PMがデータをさらに調査したい場合は、トップユーザーが誰であるかを理解したり、他の次元（案件の規模、会社の規模、製品の階層...）で結果を分解したりすることができます。

Product Adoption Dashboardは、1つのメインスニペットを使って作成されます。

xmau_202011

注：このリンクは、このスニペットを使ったSisenseビューにリダイレクトされます。`RUN SQL`ボタンを押すと、データセットを見ることができます。

以下の説明には、いくつかの例があるとより理解しやすいでしょう。

- 開発部門のCMAU推定値
- リリース管理グループの有料GMAU推定値 プランステージの
- SMAU推定値（エディション別

このスニペットは、主に推定および記録されたXMAUに推定値を与えるために使用され

ます。使用するには4つのパラメータを宣言する必要があります。

- xmau_type: すべてのXMAUデータを表示するか、有料データのみを表示するかを決定します。許容され
- る値は *「All」* または「paid」です。 xmau_level: これらの値の中から1つを選択します。`cmau'、'umau'、'smau'、'gmau'`
- フィルター：これはxmau_levelに選ばれた値に依存します。
  - CMAU: `All`を宣言するか、セクション名を選択することができます（キャメルケース、可能な値は`'dev'、'enablement'、'ops'、'secure_protect'`）。
  - SMAU: セクション名(キャメルケース、可能な値は`'dev'、'enablement'、'ops'、'secure_protect'`です)、またはステージ名(キャメルケース)のいずれかを選択します。ステージ名の一覧を以下の表に示します。
  - GMAU：ステージネーム（キャメルケース、ステージネームのリストはこちら）またはグループネーム（camel_case）のどちらかを選択できます。グループ名のリストはこちらの表にあります
- target: チャートにターゲットラインを設定することができます。空欄のままだと何も表示されません。ターゲットとして設定したい値を入力することで静的なターゲットラインを作成することもできますし、毎月の成長目標（例えば10%）を10進数の値（この例では0.1）で入力することで動的なターゲットラインを作成することもできます。

このスニペットは、限られた寸法のコンパクトなテーブルを返します。

- created_month: レポート作成月
- product_tier: こちらの定義を参照してください。これは、インスタンスにリンクされたサブスクリプションのproduct_tierではなく、インスタンスが属するproduct_tierを表します。('All'、'target'も潜在的な値です)

- の配信を行っています。**SaaS**またはセルフマネージド（「**All**」、「**target**」も潜在的な値です。
- の内訳です。配信='**SaaS**'には**SaaS**、配信='セルフマネージド'には録音されたセルフマネージドとに分かれます。
  推定自己管理。これにより、当社の設計基準に従って**XMAU**チャートを作成することができます（例はこちら）。

なお、「All」や「target」も潜在的な値です。
- 版である。CE、EE、SaaS（「All」、「target」も可能性のあ

る値です） そして、1つのメジャー。

- **mau_value:** 特定のディメンションのアクティブ・ユーザーの数

推定アルゴリズムの詳細については、現在の方法論と私たちのビジョンについての詳細をこちらでご覧いただけます。

月間製品使用量の推移

このモデルでは、使用状況のpingデータと、ライセンス、salesforce、zuoraのデータを結合します。つまり、販売/財務と製品のデータを同時に扱うことができるのです。例えば、次のような質問に答えられるようになります。

- XMAU計画に最も貢献している大学上位10校（課題を作成したユーザー数
- 特定のステージの月別および製品層別の採用率は、SMAUを業界
- 別に分割して検証する

dbtモデルは、各カラムの定義がしっかりと書かれています。また、この新しいモデルを使って答えられる質問を示すダッシュボードも作成しました。

課題を作成したユーザー数の月別推移をプロットする基本的なSQLクエリ。

```
SELECT
  report_month,
  main_edition,
  delivery,
  SUM(月次指標の値)
FROM legacy.mart_monthly_product_usage
WHERE metrics_path = 'usage_activity_by_stage_monthly.plan.issues' GROUP
BY 1,2,3
```

CIパイプライン機能を使って、EDU/OSSサブスクリプションのトップ10を抽出する基本的なクエリです。

```
SELECT
  host_name,
  license_id,
  ping_id。
  サブスクリプション名_slugify,
  ping_product_tier,
  main_edition,
  monthly_metric_value
FROM legacy.mart_monthly_product_usage
WHERE metrics_path = 'usage_activity_by_stage_monthly.verify.ci_pipelines'
  -- スニペット[last_month]は、最後の暦月のみから値を取得するために使用されます AND
  reporting_month = [last_month]
  AND DELIVATION = 'Self-Managed'
  is_program_subscription は、EDU/OSS プログラムの一部であるすべてのサブスクリプションに対して TRUE
に設定されるブーリアンフラグです。
  AND is_program_subscription
ORDER BY monthly_metric_value
DESC LIMIT 50
```

有料のセルフマネージドインスタンスでデプロイメントを作成したユーザ数をSalesforceの業種別に分割して月ごとにプロットする基本的なクエリです。

```
SELECT
  reporting_month,
  IFF(ultimate_parent_industry IS NOT NULL, ultimate_parent_industry, 'Unknown') AS
ultimate_parent_industry,
  SUM(月次指標の値)
FROM legacy.mart_monthly_product_usage
WHERE metrics_path = 'usage_activity_by_stage_monthly.release.deployments' AND
  is_paid_subscription
GROUP BY 1,2
```

エンティティ・リレーションシップ・ダイアグラム

私たちの目標のひとつは、分析に必要なすべてのデータを簡単に提供できる1モデルを作ることです。しかし、ソリューションの開発を続けていると、このモデルでは得られない情報が出てくることがあります。ここで、エンティティ・リレーションシップ・ダイアグラムの理解が役立ちます。このモデルは、アクセスしているレイヤーを作成するためにどのテーブルが結合されているかを示しています。このモデルは、より深く掘り下げて、さらなる洞察を得ようとするときに役立ちます。

		Diagram/EntityGrainPurpose
ファクト 月間使用量Pingデータ ERDとデータフロー	host_id, instance_id, reporting_month, metrics_path	さまざまなインスタンスおよびサブスクリプションのディメンション別に、機能の使用状況を把握することができます。

▶ データの分類

xMAU分析をサポートするデータの中には、[オレンジ](/handbook/engineering/security/data-classification-standard.html#orange)または[イエロー](/handbook/engineering/security/data-classification-standard.html#yellow)に分類されるものがあります。これには、アカウントからのORANGEの顧客メタデータ、SalesforceやZuoraからのコンタクトデータ、GitLabのNon公開の財務情報などが含まれ、これらはすべて公開されるべきではありません。このダッシュボードからデータを共有する際には、その詳細が GitLab の組織内に留まるようにし、外部に共有する際には適切な承認を得られるように注意しなければなりません。また、行やレコードレベルの顧客メタデータを扱う際には、個人のデバイスやラップトップにデータを保存しないよう、常に注意を払う必要があります。これらのデータは、[Snowflake](/handbook/business-technology/data-team/platform/#data-warehouse)と[Sisense](/handbook/business-technology/data-team/platform/periscope/)に残すべきであり、特別な承認がない限り、これらのアプリケーションを通してのみ共有されるのが理想的です。

ORANGE

- 説明行またはレコードレベルの顧客および個人データ。オブジェク
- トです。
  - `dim_billing_accounts`
  - `dim_crm_accounts`
  - `usage_ping_mart`

▶ ### ソリューション・オーナーシップ

セルフサービス・ダッシュボード・ソリューション

	ダッシュボード目的
エグゼクティブ・オーバービュー - TBD	このダッシュボードは、すべてのメトリクスの現在のステータスをエグゼクティブ・オーバービューとして表示します。
Dev Section Analysis - TBD	このダッシュボードでは、関連するメトリクスの現状をセクションごとに紹介しています。
DRAFT: ハンドブック更新のための一元化されたダッシュボード	このダッシュボードのチャートは、ハンドブックのエンベッドに使用されます。

これらのスニペットを使って、特定のxMAUチャートのターゲットを更新するには？

当社の標準化されたスニペットとビジュアライゼーションを使用したすべての埋め込みPI/xMAUチャートでは、ターゲットの更新は非常に簡単な4ステップのプロセスです。

1. 興味のあるPIにアクセスし、sisenseのリンクpi_pageをクリックします。
2. チャート編集」をクリックします。
3. 最後の値を変更する(これが目標値です。10進数でやれば一定の割合で増えます。完全な数値を入力すると、その数値が目標ラインになります。) ターゲット
4. 保存ボタンをクリックすると、保存されます。

トラステッドデータソリューション

トラステッドデータフレームワーク

手動でのデータ検証

- 手動使用Ping検証ダッシュボード バージョン
- 信頼できるデータ ダッシュボード

---

layout: handbook-page-toc title:"予測されるXMAUのアルゴリズム"

予測アルゴリズムの説明

パフォーマンス指標のページによると、現在xMAUには3つのバージョンがあります。予測されるxMAUは、現在の成長率で3年後に使用量がどのようになるかをプロダクト・リーダーシップに示すためのものです。

最初に提案された解決策は、次の式を適用して、M月の予測xMAUを計算することです。

```
予測xMAU(M月) = 予測xMAU(先月) + (現在のMoM成長量 x month_difference(先月とM月の間))
```

MoM growth Amountは、前四半期の平均月間絶対成長量として算出されます。つまり、仮定すると、過去4

ヶ月間のEstimated Plan SMAUは以下のような数字になります。

月	エスティメイト SMAU
M-3	94
M-2	95
M-1	97.5
M	100

平均MoM成長量(M - (M-3) / 3)が2で、現在の値が100です。12ヶ月後のSMAU予測値を算出したいと思います。

```
予測プランSMAU(先月＋12ヶ月)＝推定プランSMAU(先月)＋(2×12)＝100＋24
= 124
```

上記の計算式により、Plan SMAUの12ヶ月後の予測値は124となります。

SaaS、Self-Managed CE、Self-Managed EEの成長率の調整

SaaS、CE、EEの間の成長傾向は大きく異なる可能性があります。これを考慮して、アルゴリズムでは、あるグループのCE、EE、SaaSの利用量を考慮し、それぞれの配信タイプの平均成長率に基づいて成長率を調整しています。

そこで、「**Predicted SMAU for Plan in 12 Months**」を計算してみましょう。以下のような前提条件があります。

配信と版	SMAU	前月比成長率
CE	**50**	**2%**
EE	**20**	**4%**
SaaS	**30**	**3%**
合計	**100**	

そのため、**CE**、**EE**、**SaaS**については、上記で説明した同じ計算式を適用して、それぞれ異なる**Predicted SMAU**を算出します。

```
予測SMAU、CE = 50 x (1+ (0.02 x 12)) = 50 x 1.24 = 62
予測SMAU、EE = 20 x (1+ (0.04 x 12)) = 20 x 1.48 = 30
予測SMAU、CE = 30 x (1+ (0.03 x 12)) = 30 x 1.63 = 41

予測SMAU = 予測SMAU, CE + 予測SMAU, EE + 予測SMAU, SaaS = 62 + 30 + 41 = 133
```

そのため、**12**ヶ月後の**SMAU**予測値は**133**となります。この

ロジックを使った**WIP**ダッシュボードは、こちらでご覧い

ただけます。

次のステップ

私たちは、この問題に取り組むためのさまざまな方法を検討しています。私たちは、さまざまな選択肢をまとめたダッシュボードを作成しました。

追加のオプションを検討。

- 一定の**MoM**成長率を使用：つまり、毎月同じ成長率を適用します。つまり、**Predicted(Month ＋1）＝予測値（月）×MoM**成長率
- **Prophet**という外部の予測用**Python**ライブラリを使って、前月のデータに基づいて成長を予測する。

---

layout: handbook-page-toc title:"予測されるXMAUのアルゴリズム"

xMAU バリエーション - Trusted Data Framework

xMAUの**KPI**は、ビジネスの健全性とその成長を評価する非常に重要な指標です。**KPI**は、月に**1**度開催されるプロダクト・キー・レビューで精査されます。また、ハンドブックの製品パフォーマンス指標のページにも掲載されています。

これらの理由から、データチームは一連のツール（ダッシュボード、アラート、カスタムテスト）を作成し、**KPI**の直接の所有者が**KPI**の変動を容易に監視し、極端な傾向や疑わしい傾向がある場合には警告を受けることができるようにしました。

SiSenseのアラート

SiSenseにはアラート機能があり、クエリの結果が定義された値/閾値よりも高い、同じ、または低い場合に、ユーザーに通知を送ることができます。

KPIオーナーがメトリクスの変動を把握するのに役立つ**xMAU**メトリクス用の**SQL**アラートセットを作成しました。

- **[td_xmau] Est.CMAU**変動監視。このアラートは、**CMAU KPI**の月次変動が**±10%**を超えた場合に発生します。
- **[td_xmau] Est.UMAU**変動監視。このアラートは、**UMAU KPI**の月次変動が**±10%**を超えた場合に発生します。

- **[td_xmau] Est.SMAU Variations Monitoring**です。このアラートは、いずれかの**Stage**で**MoM**の絶対値の変化が**10%**を超えた場合にトリガされます。**SMAU**値が**1000**未満の**Stage**は除外されます（これらの**Stage**は**SMAU**メトリクスが新たに計測されており、当初は極端な変動を受けると想定しています）。
- **[td_xmau] Est.GMAU**変動の監視。このアラートは、グループのいずれかが**MoM**の絶対的変化が**10%**を超えた場合にトリガされます。**SMAU**値が**1000**未満のグループは除外されます（これらのステージでは、**SMAU**メトリクスが新たに計測されており、当初は極端な変動を受けるものと想定しています）。

これらのアラートは現在、毎週月曜日と毎月**1**日に送信されています。

SiSenseのダッシュボード

データチームが作成した**TDxMAU**ダッシュボードは、迅速な理解に役立つ一連のビジュアルをまとめたものです。
これらの**KPI**の健康状態を

次のステップ

現在、アラートは**1**人のチームメンバー（**Mathieu Peychet**）にのみ送信されています。アラートを受信するユーザーのリストを作成する必要があります（私たちの提案は、**Product Intelligence PM**と**Product Analytics Team**に送信することです）。

また、このアラートシステムをより良く、よりスマートにするために、他のソリューションも試しています。

- 現在、推定**xMAU**の計算を**dbt**に移行しています。
- **GreatExpectations**は、カスタムデータテストを作成するために設計されたオープンソースのソリューションで、試してみる予定です。

layout: handbook-page-toc title:"データ開発" 説明"このページでは、データ開発のライフサイクルを定義しています"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

GitLabでのデータ開発

GitLabでは、インサイトとビジネス上の意思決定を促進するデータソリューションを構築するために、**3**つの異なる、しかし相互に関連するアプローチを展開しています。これらのアプローチは互いに補完し合い、ビジネス、解決すべき問題、問われている質問に応じて、必要なスピード、品質、信頼性のレベルで結果を提供することに重点を置いています。これらのアプローチは相互に補完し合い、進化していくものであり、必要に応じて初期の段階で開発したものを後の段階で活用することができます。初期の段階で開発されたデータソリューションは、ビジネス上の必要性が十分にあれば、後の段階で改善・強化することができます。すべての分析は、確立されたデータ分析プロセスに従って行われます。

この**3**つのアプローチ「**Ad-Hoc**」「**Business Insights**」「**Trusted Data**」を紹介します。

	アドホック	ビジネスインサイト	信頼できるデータ
使うとき方向性／緊急性	分析	ルーチン／オペレーション分析	ミッションクリティカル分析
SiSenseによるビジュアライゼーション	任意	必須	必須
エンタープライズディメンションを利用して構築モデル	オプション	任意	必須
データカタログ	への登録 任意	必須	必須
信頼性の高いデータの開発をプロセス	フォローするオプ	任意	必須
ション			
信頼性の高いデータを実施テストを	を使用してテスト	任意	必須
オプション			
ソースクによる監査可能性（	システムへのリンオプション	任意	必須
信頼できるデータ」	ブランド化**N/A**	**N/A**	必須
'Business Insights'	ブランド**N/A**	必須	**N/A**

アドホックなデータ開発

アドホックは、分析作業の典型的な最初のステップであり、**1**回限りまたは限定的に使用するレポートやダッシュボードを提供することになります。アドホック開発は、既存のデータソリューションでは質問に答えられない場合に行われます。その場限りの分析のために開発されたコードは、長期的なソリューションに活用するために書かれたものではなく、結果を迅速に提供するためのものです。その場限りの分析を行うために、**Analyst**は通常、**Enterprise Data Warehouse**に対して**SQL**クエリを書いて実行したり、**Sisense**や**Python**などのツールを使って分析するためのデータを抽出したり、あるいは既存のダッシュボードを活用したりします。場合によっては、テキストファイルやスプレッドシート、その他のデータソースから新しいデータを入手する必要があります。

ほとんどの場合、アドホックレポートは当面のビジネスニーズを解決し、それ以上の対応は必要ありません。しかし、アドホック分析の結果、追加の分析を必要とする結果が得られることもあります。また、アドホック分析の結果が、より信頼性の高いソリューションに発展させるのに十分なほど重要な場合もあります。このような場合には、ビジネスインサイトソリューションまたはトラステッドデータソリューションの作成が決定されます。

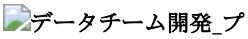
ビジネスインサイト データ開発

ビジネスインサイトは、安定した信頼性の高いレポートが必要とされるものの、構造化されたエンタープライズ次元モデルがまだ利用できないソリューションの大半を占めています。ビジネスインサイトソリューションは、それぞれのメトリクスの**SSOT**としての役割を果たし、レポート全体の中で重要な役割を果たしています。

ビジネスインサイトソリューションは、データテスト、コードレビュー、データカタログへの登録などの品質検証が含まれる点で、アドホックレポートとは異なります。ビジネスインサイトソリューションは、EDMの一部を利用することができますが、完全にEDMに基づいているわけではありません。しかし、Trusted Data Solutionと比較すると、Business Insightsソリューションは、完全なテストカバレッジとEDMカバレッジがありません。

信頼されるデータ開発

Trusted Dataは、企業が利用可能な最も完全で、信頼性が高く、正確な分析を提供します。組織が成熟し、アナリティクスの価値が高まるにつれ、Trusted Dataは進化し、開発の厳しさも進化していきますが、基本的なステップは一貫しており、要件収集、設計、反復的なワイヤーフレーム、テスト、運用監視などが含まれます。



されるデータソリューション基準

すべてのTrusted Dataソリューションは、以下の基準を満たす必要があります。

1. ビジネス上の問題が定義され、明確な収益への影響が確立されている。
2. 開発を管理するためにデータプロジェクトエピックが作成される
3. 要件と成功基準がEpicに取り込まれ、追跡される
4. v1.0およびv1.1のスコープが定義されており、リリースサイクルが前もって設定されている（例：毎週、隔週、毎月）。
5. ダッシュボードのワイヤーフレームをLucidまたはSisenseで作成し、ユーザーと共有し、"最終ドラフト "に向けて繰り返し作業を行います。
6. 次元データモデルは、エンタープライズ次元モデルバスマトリックスに設計・統合されています。
7. 信頼できるデータテストの作成と展開
8. ソリューションは、ソースシステムへのデータ検証を含むユーザーアクセプタンステストの段階に入ります。
9. ソリューションがデータカタログに登録されていること
10. 必要なトレーニングやユーザーイネーブルメントを含む、ソリューションの展開

---

layout: handbook-page-toc title:"Data Team Data Management Page" 説明"データ管理ページは、エンタープライズデータプラットフォームと関連する活動の管理、セキュリティ、および統治に関する内容をカバーしています。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .hidden-md .hidden-lg}。

データマネジメントページへようこそ

このページでは、EnterpriseDataPlatformおよびそれに関連する活動の管理、セキュリティ、およびガバナンスに関するプラクティスとポリシーについて説明します。Enterprise Data Platformの技術的なコンポーネントは、GitLab Tech Stackに記載されています。

データセキュリティの実践

エンタープライズデータプラットフォームは、多くのシステムから収集したデータを取り込み、処理し、保存します。すべてのデータが同じ重要性を持っているわけではありません。私たちはCriticalSystemTierフレームワークとDataClassificationStandardを使用して、どのデータが最も重要で、どのようにセキュリティを確保するのが最適かを判断しています。

Sisense

- Sisenseでロールベースのデータアクセススキームを導入してい
- ます。ユーザーアクセスはOktaで管理
  ロールとスペースでデータアクセスを管理
  各ユーザーにはSisense Roleが割り当てられ、ダッシュボードやレポートへのデータアクセスが可能になります。

- **Sisense**スキーマは**Snowflake**データアクセススキーマと相互に作用し、ユーザーがどちらのシステムからもデータへの「バックドア」を持つことがないようにします。

追加のコントロールは以下の通りです。

- 未使用のダッシュボードをアーカイブ化
- システムへのアクセスはAPIキーで管理

スノーフレーク

役割ベースのデータアクセススキームをSnowflakeに導入します。

- ユーザーアクセスはOktaで、アクセスリクエストはGitLabで管理されています。
- 各ユーザーには、さらに1つ（職能に応じた役割）[/handbook/business-technology/data-team/platform/#snowflakepermissions-paradigm]が割り当てられ、この設定はPermifrostで管理されます。
- SnowflakeスキームはSisense Data Accessスキームと相互に作用し、ユーザーがどちらのシステムからもデータへの「バックドア」を持つことがないようにします。

追加のコントロールは以下の通りです。

- データの分類基準に基づき、データはデータベースとスキーマで管理される すべてのクエリ/
- ユーザー/プロセスには、事前に定義されたウェアハウス、またはコンピュートリソースが割り
- 当てられる（パスワードはローテーションされる）[/handbook/business-technology/data-team/platform/#passwords].

一般的なデータ・セキュリティ・コントロール

- データコントロールを定義する目的では、Enterprise Data PlatformはTier1システムです。
- 重要：お客様のプライベートREDデータは、Enterprise Data Platformに永久保存することはできません。

コントロール	RED	ORANGE	YELLOW
一般的なデータ管理			
データ登録リスト	必須	必須	推奨
休息時の暗号化	必須	必須	必須
トランジットでの暗号化	必須	必須	必須
プライバシーの見直し	必須	推奨	必要なし
データ保持の手順	必須	推奨	必要なし
データインフラコントロール			
マルチファクタ認証	必須	必須	必須
役割ベースのアクセス	必須	必須	必須
アクセスロギング	必須	必須	推奨
データウェアハウスコントロール			
四半期ごとのSnowflakeユーザー監査	必須	必須	必須
四半期ごとのSiSenseユーザー監査	必須	必須	必須
四半期ごとの変更管理レビュー	必須	推奨	必要なし
四半期ごとのREDデータスキャナー	必須	N/A	N/A
エンドポイントデバイス			
マルウェア対策	必須	必須	必須
フルディスク・エンクリプション	必須	必須	必須
四半期ごとのデータパージ	必須	必須	必須

データインフラ：データウェアハウスの一部としてデータにアクセスまたは処理し、エンドユーザーがデータを利用できるようにするためのあらゆるシステムが含まれます。

- データウェアハウスの制御。エンタープライズ・データ・ウェアハウスは、ティア1システムです。
- エンドポイント・デバイス。データウェアハウスにアクセスするすべてのエンドポイントは、Tier 1に分類されます。

四半期ごとのデータヘルス・セキュリティ監査

四半期ごとの監査では、適切な担当者が正しいデータアクセス設定を行い、データパイプラインが正しく実行されているかなど、システムのセキュリティを検証します。

このプロセスは、「四半期データの健全性とセキュリティ問題のテンプレートによっ

てサポートされています。ここでは、活動のチェックリストのサンプルを紹介します

。

- ☐ スノーフレーク
  - Snowflakeから離脱した従業員の活動を停止させる
  - オフボーディングされたGitLabチームメンバーのすべてのSnowflakeアカウントは、オフボーディングされた日から無効化されるべきです。このアクティビティは、オフボーディングされたGitLabチームメンバーのアクティブなアカウントがあるかどうかをチェックします。その後、アクティブなアカウントはすべて非アクティブになります。
  - 監査を実施した時点から過去60日以内にログインしていないアカウントをSnowflakeから無効にします。
  - 60日以上ログインしていない指定ユーザーのSnowflakeアカウントは無効化されます。非活性化後、ユーザーにはその旨が通知されます。GitLabチームメンバーが再びアクセス権の付与を希望する場合は、通常のARを作成する必要があります。マネージャーの承認後、アカウントが有効になります。
  - すべてのユーザーアカウントに多要素認証が必要であることを確認する。
- ☐ Sisense
  - Sisenseからオフボーディングされた社員を無効にする。
  - オフボーディングされたGitLabチームメンバーのすべてのSisenseアカウントは、オフボーディングされた日から無効化されるべきです。このアクティビティは、オフボーディングされたGitLabチームメンバーのアクティブなアカウントがあるかどうかをチェックします。その後、アクティブなアカウントはすべて非アクティブになります。
  - 監査を実施した時点から過去60日以内にログインしていないアカウントをSisenseから無効にします。
  - 60日以上ログインしていないSisenseアカウントは無効化されます。無効化された後は、ユーザーに通知されます。GitLabチームメンバーが再びアクセス権の付与を希望する場合は、通常のARを作成する必要があります。マネージャーが承認すると、アカウントが有効になります。
  - すべてのユーザーアカウントに多要素認証が必要であることを確認する。
- ☐ 信頼できるデータ
  - Golden Record TDテストをすべて確認し、合格していることを確
  - 認する。データサイレンを確認し、REDデータの存在を確認する。
  - TD: Sales Funnel ダッシュボードへのビジネスロジック変更のレポートを四半期ごとに作成する。新しいディメンション、新しいファクト、新しいマートの追加、結合の変更、新しい計算フィールドの追加などのビジネスロジックです。

---

layout: handbook-page-toc title:"Data Quality" 説明"GitLab Data Quality Programは、GitLabの生産性や効率性に影響を与えるデータ品質の問題を特定、監視、修正することを目的としています。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .hidden-md .hidden-lg}。

データクオリティープログラム

GitLab Data Quality Programは、データの継続的な改善を通じて、GitLabの生産性と効率性を向上させることを目的としています。

このプログラムは、データの問題を特定し、監視し、修正するための信頼性の高いプロセスを構築することで、この目標を達成します。データクオリティーの範囲は**GitLab**が管理するすべてのデータあり企業内データの利用可能性にのみ制限さます

EDWは、大規模かつ複数のソースシステムにまたがる大量のデータをスキャンし、データ品質の問題を検出する機能を備えているため、データウェアハウス。

データ品質に関する問題をお持ちですか？Data Quality Issue Workflowをご覧ください。{: .alert .alert-success}。

データ品質問題の種類

従来のデータ品質プログラムでは、品質問題を、完全性、正確性、一貫性、妥当性、一意性、完全性など、いくつかのタイプに分類しています。このようなニュアンスは、データ品質の専門家ではない人と接するときに混乱を招くことがあります。この問題を単純化するために、GitLab データクオリティプログラムでは、以下のデータクオリティ問題の種類を認識しています。

- 不正確なデータInaccurate Dataとは、表すべき実世界の値と一致しないデータのことです。不正確なデータの例としては、3桁の米国郵便番号があります。
- 欠損データ欠損データとは、*本来存在するはず*のNULLまたは空のフィールドやレコードのことです。欠損データの例としては、住所録の郵便番号がNULLであることが挙げられます。
- 重複したデータ重複データとは、同じデータが繰り返されるべきでないときに繰り返されるデータのことです。重複データは、データの報告方法に基づいて自然に発生することがあるため、特定するのが難しい場合があります。重複データの例としては、CUSTOMERマスターテーブルにある2つの（ほぼ）同一の顧客レコードが、両方とも1人の「現実世界」の顧客にリンクされている場合が挙げられます。

データ品質システムの構成要素

データ品質システムは、時系列で問題点を把握するスコアカードと、データの既知の問題点を特定する検出ルールで構成されています。

- データ品質スコアカード - データ品質スコアカードは、データカスタマおよびデータクリエータが使用するダッシュボードです。ダッシュボードには、サブジェクトエリアの個々の検出ルールのステータスによって測定された、サブジェクトエリアの全体的な品質が表示されます。特定の目的のために、特定の独立したデータ品質スコアカードを作成することができ、今後も作成される予定です。例えば、当社は「データ品質スコアカード-製品使用データ」を積極的に開発しており、Zuora課金システムの品質を測定するために、別の「データ品質スコアカード-Zuora」を開発する予定です。
- データ品質検出ルール - データ品質検出ルールは、フィールドや行のデータの品質を事前に定義された条件と比較してチェックするSQLベースのテストです。検出ルールを実行するには、データがすでにEnterprise Data Warehouseに存在している必要があります。検出ルールは列挙され、1つのSQLステートメントには1つのテストしか表現されません。検知ルールの例は以下の通りです。
    - 検知ルール1: 不正確なデータ - Account LocationレコードのStateフィール
    - ド 検知ルール2:重複したデータ - アカウントマスターレコードのアカウント名
    - 検出ルール3：データの欠落 - 新しいUsage Pingの送信にはライセンスキーが必要です。

運用プロセス

毎週、検出ルール「バッチ」が実行され、出力が永続的なテーブルに保存される。永続的テーブルには、実行日、検出ルール識別子、トランザクションIDが含まれ、ソースシステムへのリンクが可能になる。この永続的テーブルは、スコアカードを作成する際のベースとなります。

データ品質の問題を解決する

リメディエーションとは、品質問題を解決、修正、または排除するプロセスのことです。修正は「データ作成者」、つまりソースデータの作成に責任を持つ人やチームが担当します。品質問題を特定すること、または特定を支援することは、「データカスタマー」の責任である。

製品データ品質スコアカード

目的 - Product Data Quality Scorecard は、製品使用データに関するデータ品質の問題を定量化します。スコアカード

・ダッシュボードには、以下の情報を表示するビジュアライゼーションがあります。

- 製品のデータ品質検出ルールのそれぞれの合否の割合。合格したレコードの割合は、条件またはデータ品質検出ルールを満たしたレコードの総数の割合を取ることで計算されます。計算に使用される式は以下の通りです。
  ((passed_record_count/processed_record_count)100)

同様に、「失敗したレコードの割合」は、条件やデータ品質検出ルールを満たすことができなかったレコードの総数に対する割合を取ることで算出されます。***((failed_record_count/processed_record_count)100)***

検知ルールの合格と不合格は、合格したレコードの割合としきい値制限の割合を比較して決定されます。現時点では、しきい値は**50**に設定されています。正確なしきい値は**DRI**が決定する必要があります。

通過したレコードの割合がしきい値を超えた場合、検出ルールのステータスは緑になります。例えば、通過したレコードの割合が**72%(50%**以上**)**の場合、**72%**のレコードがデータ検出ルール**/**条件を満たしていることを意味します。

通過したレコードの割合がしきい値以下の場合、検出ルールのステータスは「赤」になります。例えば、合格したレコードの割合が**40%(50%**未満**)**の場合、**60%**のレコードがデータ検出ルール**/**条件を満たしていないことを意味します。これらのレコードには注意が必要で、データはソースチームによって修正される必要があります。

- トレンド分析チャートは、**1**週間の間の各データ品質検出ルールの合否率の変化を示します。

トレンド分析チャートのデータポイントをダブルクリックすると、**TD: Product Data Quality Detailed Dashboard V1.0**が表示され、各検出ルールで影響を受けた**ARR**の合計値とともに、データ行と集計値の詳細情報が表示されます。

- 各日の集計結果には、**Daata**品質検出ルールのそれぞれについて処理された行の総数と、ルール**/**条件を満たす（パスする）行の数、ルール**/**条件を満たさない（フェイルする）行の数が、ルール実行日によって追跡される各日について表示されます。

ダッシュボード「**TD: Product Data Quality Scorecard - Overview V1.0**」と「**TD: Product Data Quality Detailed Dashboard V1.0**」のデータは、毎日更新されます。

クイックリンク

TD: Product Data Quality Scorecard - Overview V1.0

TD: Product Data Quality Detailed Dashboard V1.0

現在、製品の使用状況データについて確認されているデータ品質検出ルールは以下の通りです。

検出ルール**ID**	ルール	説明**DRI**
1	ホスト名のインスタンスタイプがない	
2	サブスクリプション**ID**がないライセンス	
3	ライセンスが見つからないサブスクリプション	
4	ライセンスの開始日が将来になるセルフマネージドプランのサブスクリプション	
5	ライセンス開始日がライセンス終了日よりも長いセルフマネージドプランのサブスクリプション	
6	過去に購読終了日を迎えた期限切れのライセンス**ID**	
7	名前空間**ID**がない**SaaS**サブスクリプション	

追加リソース

ガイド&ブック

- データの前に立つ
- 非侵襲的データガバナンス
- データ・ライフサイクル・マネジメント（DLM）または同等の堅牢なアプローチ。

SaaSツール

FivetranもStitchも、マネージドサービスであるため、独自のデータ品質チェックを行っています。これらのデータの抽出時に問題が発生した場合は、ベンダーのサポートチームにご相談ください。

カスタム

サードパーティのツールからの抽出については、独自の**API接続やAirflowで管理しています。これらのすべてが、データ品質チェックのためのメソッドを備えているわけではありません。**

BambooHR

BambooHRのデータはカスタムコードで抽出しています。データの品質チェックには、API からの 200 レスポンスの確認や、JSON データに最低数のレコードが存在することが含まれます。

Postgres パイプライン

私たちの Postgres_Pipeline (gitlab.com、customers.gitlab.com、license.gitlab.com、version.gitlab.com からのデータを扱う) は、送信元と送信先のデータベースの行数が一致しているかどうかをチェックします。

トランスフォーメーション・データ・クオリティ

私たちは、倉庫内のすべての変換にdbtを使用しています。私たちは、すべての新しいdbtモデルのテストを要求し、必要に応じて定期的にテストを更新します。これらのテストは、抽出テストやチェックと同様に、上述のデータ品質哲学に沿って書かれるべきです。

データ パイプライン ヘルス ダッシュボード

課題を見る

第1回目のイテレーションでは、以下を中心に追加しました。

- **キーテーブルの日々のレコードの挿入と更新の速度をテストする SQL ステートメント（行数テス**
- **ト） キーテーブルのキーフィールドの集計値をテストする SQL ステートメント（列値テスト**
- **キーテーブルのキーレコードの存在をテストするSQL文（ゴールデンレコードテスト**
- **これらの結果をシンプルに可視化するワイヤーフレームのダッシュボード**

---

layout: handbook-page-toc title:"Data Team Services" description:"Data Team Services"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

スペシャルイベントのオンコール

Special Event On-Callは、データチームが提供するサービスで、エグゼクティブ・プランニング・セッションなどの短期間の特別なイベントに必要な迅速な対応能力を提供します。オンコールサービスは、サポートに多大な労力と調整が必要なため、ほとんど利用されていません。データチームは、既存の仕事やスタッフの空き状況により、オンコールのリクエストに応えられない場合があります。

どのような仕組みになっているのでしょうか？

オンコールのリクエストが確認されると、データチームはイベント専用のSlackチャンネルを設定します。イベント参加者は、このチャンネルを使ってリクエストを送信し、レスポンスを受け取ります。解決までの時間は、リクエストの複雑さによって異なります。データチームは新しいリクエストに対して、(a)リクエスト完了までの推定時間、(b)トラッキング用の課題へのリンクを返信します。データチームがリクエストの解決に長時間を要する場合は、リクエスト者に通知します。

イベント終了後、データチームは次のことを行います。

- 作成された課題のリストとそのステータスを含むイベントサマリーレポートの提供
- フォローアップが必要な課題の特定
- Slackチャンネルの閉鎖

オンコールサービスの依頼

1. イベントの10営業日前までに、DataTeamプロジェクトで課題を作成します。
2. 以下の内容を「説明」に追加します。
    1. イベントの開始日と終了日
    2. 毎日の開始時刻と終了時刻
    3. 取り組むべき質問の種類と主題カテゴリ（財務、販売、成長など）について
    4. 想定される質問の総数
    5. Slackチャンネルを使用しているリクエスターの推定数
3. 詳細が確定したら、データチームがスタッフのモデルを作成します。
4. データチームは、オリジナルの問題でお客様に回答します。
    1. コミュニケーションに使用するSlackチャンネル
    2. スタッフの配置計画（名前とカバー時間を含む）（個人にDMを送らず、すべてのコミュニケーションを確立されたSlackチャンネル内で行ってください。

社内データチームのプロセス

1. データチームのマネージャーは、オンコールのリクエストを確認し、既存の仕事の都合やスタッフの空き状況を考慮して、サポートできるかどうかを判断します。
2. リクエストに対応できない場合は、データチームのマネージャーがリクエスト者に対応し、代替のサポートプランを共同で検討します。
3. リクエストがサポートされる場合は、データチームのマネージャーが行います。
    1. 名前とカバー期間を含むスタッフモデルの作成
    2. 影響を受けるデータチームのメンバーに対して、人員配置計画とカバー率の要件を調整する
    3. 影響を受けるプロジェクトの関係者に、それぞれのプロジェクトへの潜在的な影響を通知する
    4. 新しいSlackチャンネルを「data-oncall--」という形式で作成します。
    5. オリジナルのIssueをSlackチャンネル名とスタッフのモデルで更新する
    6. スタッフ配置計画とローテーションスケジュール（名前を含む）をslackチャンネルにピン留めする
    7. 原稿依頼者への機種・プランの確認

オンコール・イベントサポートの流れ

データチームのオンコールサポートプロセスでは、コミュニケーションを合理化し、責任を明確にするためにティアリング方式を採用しています。

- ティア1メンバーは最初の対応者であり、イベントSlackチャンネルの監視、リクエストの確認、リクエストへの対応、課題の作成、可能であれば課題の解決を行います。
- Tier2メンバーは、Tier1メンバーからの要請に応じて、Tier1メンバーが対応できないリクエストをサポートします。

ティア1とティア2のメンバーは、オンコールイベント中も定期的に連絡を取り合う必要があります。休息が必要なTier 1チームメンバーは、Tier 1の責任を引き継ぐ資格のあるチームメンバーを見つける責任があります。

ティア1の責任

1. 事前に設定した取材期間中に、イベントSlackチャンネルをモニターします。
    1. 新規のリクエストに対しては、投稿後30分以内に「Request Received, We Are Working On It」のような返答をする。

2. 最初のリクエストから1時間以内に、より正式な回答を調整して伝え、以下を含む。
3. リクエスト者に代わって作成される新しい課題へのリンク（まだ作成されていない場合
4. 問題解決に要する時間の目安（例：「1時間以内に回答します」など
5. 問題が特に複雑であったり、解決に数時間を要する可能性がある場合には、明確なメッセージを表示します（例：「このリクエストはすぐに対応することが難しく、解決には時間がかかります」）。

2. 受信したリクエストは可能な限りすべて解決してください。それができない場合は、Tier2のメンバーに接続して助けを求めてください。
3. 定期的に、Issueコメントを使ってリクエスト者に最新情報を提供し、特に緊急性が高い場合は、イベントのSlackチャンネルに最新情報を提供します。

Tier2の責任

1. ティア1メンバーのサポートができること。
2. 定期的にイベントのslackチャンネルをチェックし、全体の状況を把握してください。
3. 長時間の休息が必要な場合は、Tier1メンバーにアラートを出す。

---

layout: handbook-page-toc title:"データチームの方向性"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

このページには、将来の見通しに関する内容が含まれており、現在の状態または予定されている機能セットや能力を正確に反映していない可能性があります。

四半期目標

FY22-Q1

FY22-Q1は、データチームにとって非常に重要な四半期となりました。これは、フュージョンチームによる新しい組織モデル、Trusted DataソリューションとSales Funnel Dashboardの最初のメジャーリリース、製品使用状況データのGainsightへの統合など、FY21の初期に開始された多くのイニシアチブをチームが完了したためです。当チームは、当四半期にすべての主要なコミットメントを成功裏に達成し、第2四半期に向けて成功と規模拡大のための体制を整えることができました。

- FY22-Q1Retrospective
- FY22-Q1ReportCard
- FY22-Q1 Handbook Jamboree

		ThemeObjectiveNotes and Key Results
データプログラムの成熟	目的1：FY22フュージョンチームの定義、Project Compassのサポート、アンケートによる社内関係者の満足度の報告、優先順位と成果物の整合性の確認。	Fusion Teams、XMAU Handbook Page、Gainsight SM Usage Data、Marketing Data Martを展開しました。
レベル2- コーポレート・インテリジェンス	目的2：GTMパフォーマンス管理の効率化	Shipped TD: Sales Funnel dashboard, Shipped Analytics Hub MVC
レベル2- プラットフォームの成熟度	目的3：エンタープライズデータプラットフォームの信頼性と安全性の向上	四半期監査のv1を作成、Data Pump v1を出荷

FY22-Q2

FY22-Q2DataTeamHandbookUpdates

- **FY22-Q2 Retrospective** `Coming Soon`
- **FY22-Q2 Report Card** `Coming Soon`
-

- **FY22-Q2 Handbook Jamboree** `Coming Soon`

テーマ	目的
Level 2-Corporate	**IntelligenceObjective 1: Accelerate GTM Teams and Project Compass**
Level 2-Corporate	**IntelligenceObjective 2: Enable GitLab to be Public Company**
Level 2-Platform Maturity	**Objective 3: Improve Product Decision Quality and Accuracy with Trusted Data**（信頼できるデータによる製品決定の質と精度の向上

FY22-Q3

テーマ	目的
Level 2-Corporate	**IntelligenceObjective 1: Accelerate GTM Teams and Project Compass**
Level 2-Corporate	**IntelligenceObjective 2: Public Company Readiness**
Level 2-Product & Customer	**IntelligenceObjective 3: Accelerate R&D Teams with Trusted Data**

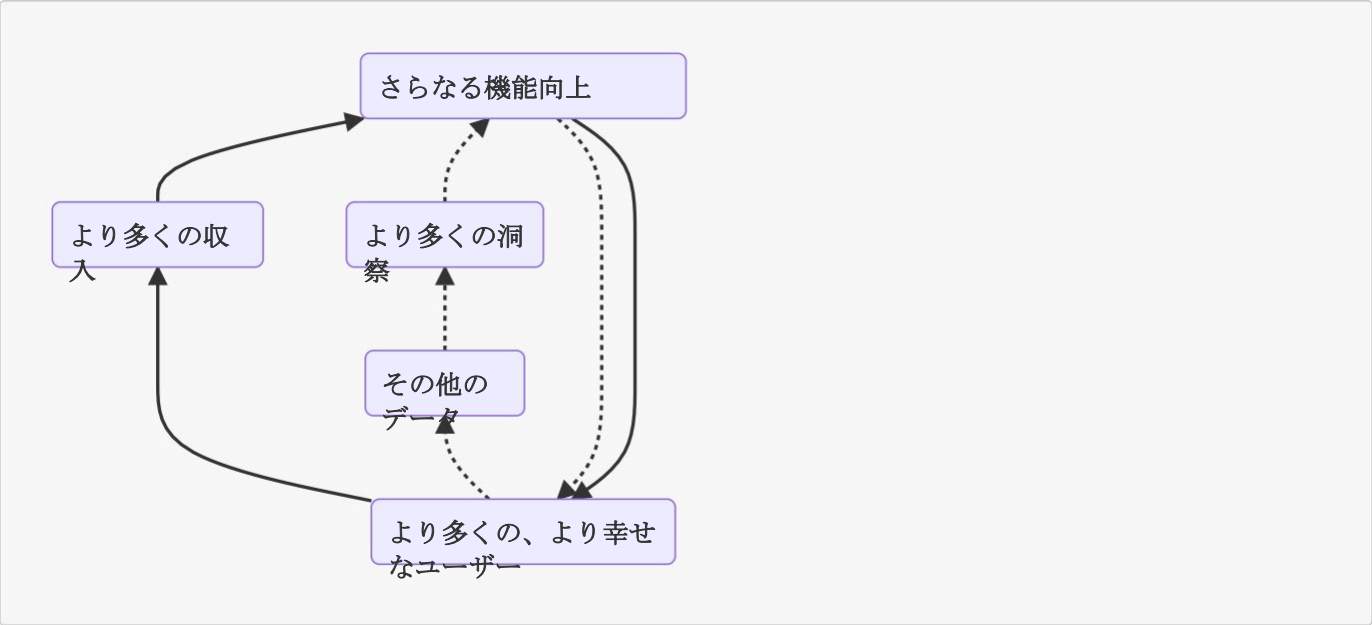
戦略

ミッションを達成し、責任を果たし、**GitLab**が成功した上場企業になるための重要なステップとして、私たちは**EDP**（**Enterprise Data Platform**）を構築しています。**EDP**は、単一の統一されたデータおよび分析スタックであり、セルフサービスデータやデータクオリティーなどの幅広いデータプログラムを備えています。**EDP**は、**GitLab**の**KPI**、部門横断的なレポートや分析、さらにはチームメンバー全員が信頼できるデータを使ってより良い意思決定を行えるようにします。**EDP**は、データパブリッシングやプロダクトなどの機能により、**GitLab**の分析能力をさらに加速させます。これらの機能は、ビジネスシステムや**GitLab**製品に統合され、お客様が使用するために強化され、集約されたデータを提供します。この加速は、**GitLab**のオープンコアやデベロップメントスペンドのフライホイールのように、「データフライホイール」の開発によって行われます。

データフライホイール

カスタマー&プロダクトインテリジェンス フライホイール

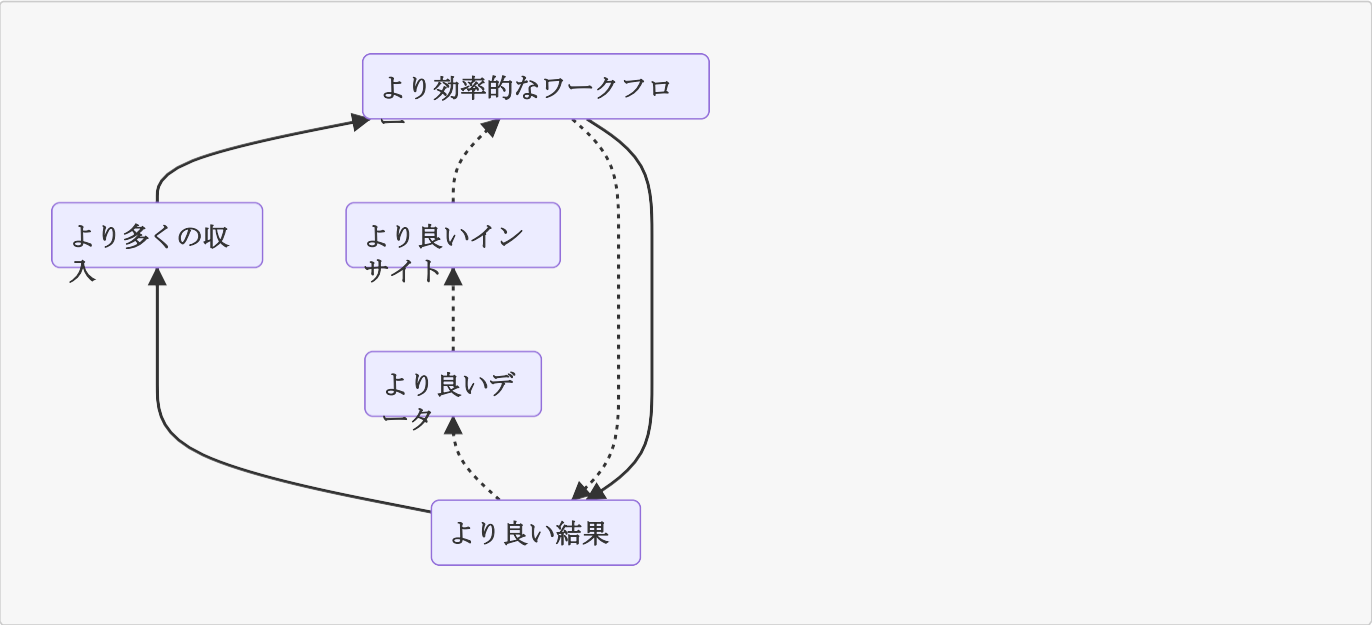
カスタマー&プロダクトインテリジェンスフライホイールは、カスタマーエクスペリエンスの向上に重点を置き、ユーザーと製品のインタラクション、カスタマーユースケース、製品開発、製品採用、およびカスタマージャーニーのほとんどの側面に関わるデータと分析を網羅しています。



コーポレート・インテリジェンス

コーポレート・インテリジェンス・フライホイールは、（社内の）ビジネス効率の向上に焦点を当てており、これはビジネスワークフローの計測、監視、および改善によって達成されます。コーポレート・インテリジェンス・チームの一般的なアウトプットは以下の通りです。

パフォーマンス・ダッシュボード、バランス・スコアカード、KPI、MBOなど、データを活用した関連フレームワーク。

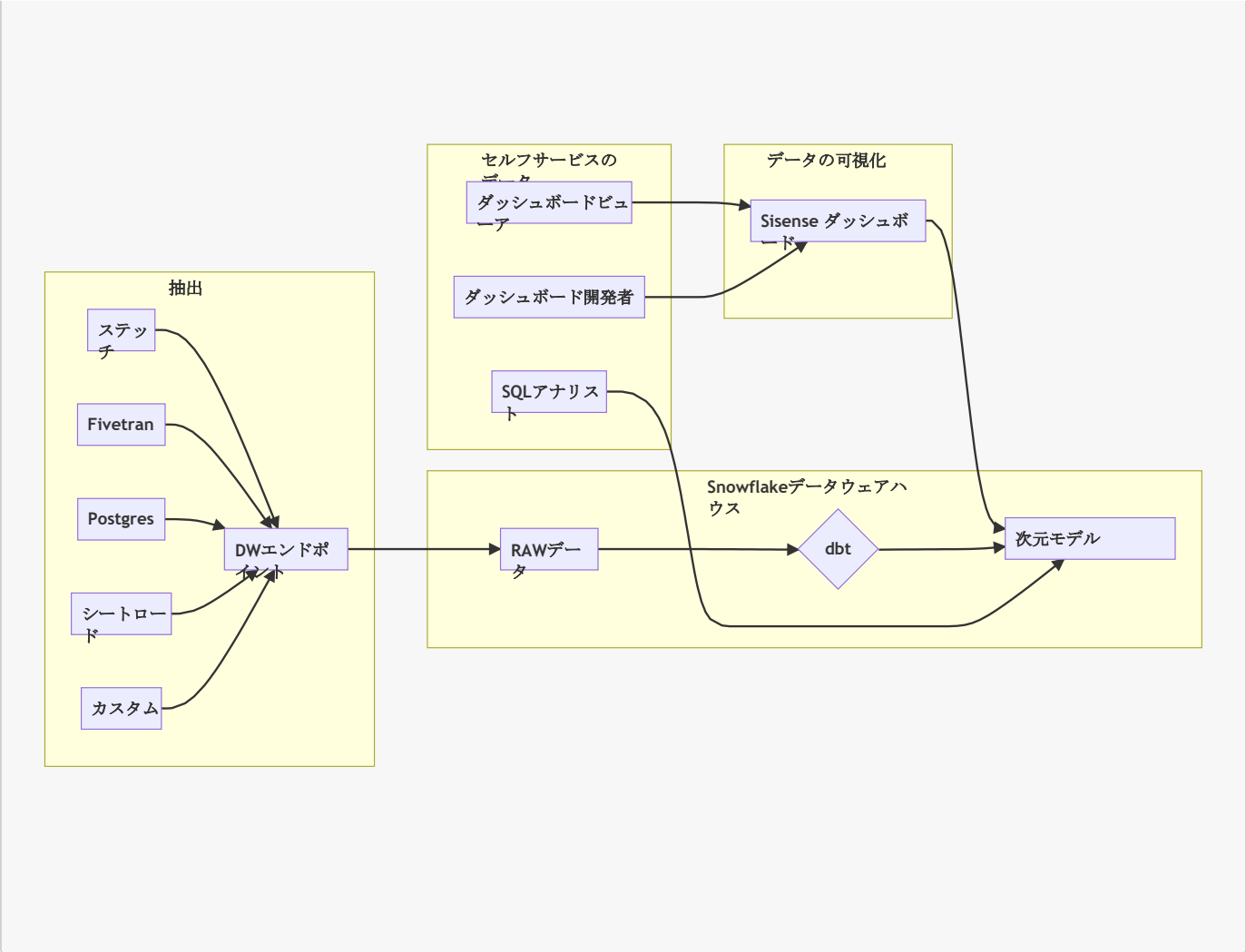


短期的な方向性

私たちの短期的な目標は、GitLabのKPIをサポートし、GitLabの最も重要な2つのビジネスプロセス（*リードからキャッシュまでのサイクルと製品リリースから採用までのサイクル*）をカバーするSelf-Serve Dataを提供することで、データケイパビリティモデルのレベル2に到達し、GitLabを公開企業にすることです。

優先順位の高い順に、私たちが積極的に構築しているEDPの要素は以下の通りです。

1. Sisenseのダッシュボードや Snowflakeでの SQL分析など、レポートや分析のための唯一の情報源となるエンタープライズ次元モデルです。
2. 信頼できるデータフレームワークにより、すべてのデータソリューションが徹底的にテストされ、自動化された継続的な検証が行われます。
3. KPIを含むパフォーマンス分析をサポートするためのデータ・ビジュアライゼーション。
4. セルフサービスデータプログラムは、すべてのGitLabチームが信頼できるデータに確実にアクセスできるようにするためのプログラムで、データチームの関与は必要ありません。セルフサービスデータプログラムには、次の3つのオプションがあります。
    1. ダッシュボード・ビューアー - GitLabチームのメンバーがSisenseポータルにログインし、事前に作成されたダッシュボードにアクセスできます。
        1. ダッシュボード開発者 - Sisenseの認定を受けたGitLabチームメンバーがSisenseポータルにログインし、エンタープライズディメンショナルモデルで利用可能な信頼できるデータをソースとした独自のダッシュボードを構築します。
        2. SQL Developer - SQL Analysisの資格を持つGitLabチームメンバーが、Snowflakeでホストされている Enterprise Dimensional Modelから信頼できるデータを照会するための独自のSQLを作成します。



長期的な方向性

当社の長期的な方向性は、マスターデータ管理、データレイク、高度な分析など、成熟したエンタープライズデータプラットフォームに見られる機能をEDPに追加することで、「年単位」で測定されます。また、レベル2に到達した後は、次のようになります。

- オープンソースのデータプロジェクトに貢献する方法を探したい
- データのパイプラインや処理コンポーネントとしてMeltanoを使用したい。
- EDPの一部をGitLab.comと統合し、GitLabの顧客に詳細な分析機能を提供したい OktaのBusiness@Workのような
- DevOps業界のベンチマーク・レポートを提供したい
- レベル3に到達するために必要な要素を確保するために、データ技術スタック全体を再検討する。

成功の測定

短期的な方向性に対する進捗状況を以下の方法で測定します。

1. データチームのKPI
2. データバリューピラミッドに沿った当社の結果のビジネスインパクト
3. 当社が提供するデータ機能は、データ機能モデルに対応しています。
4. The Data Team Quarterly Report Card

長期的な進捗を測る基準はまだ定めていません。データチームのKPI

1. 過去最高のセルフサービス・データ顧客数を実現
2. 月間 アクティブなセルフサービス・ダッシュボード開発者の数
3. 月間アクティブなセルフサービス型SQL開発者の数
4. ダッシュボードのトラフィックに占めるユーザー生成コンテンツの割合（月間

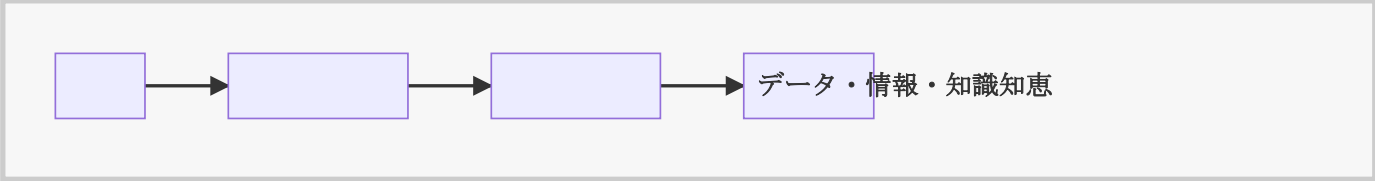
完全なエンタープライズデータプラットフォーム

次の表は、大企業が必要とする広範なデータと分析を解決できる、成熟したエンタープライズデータプラットフォームの機能を表しています。記載されているすべての機能が、GitLabの短期的なニーズや既知の長期的なニーズを満たすために必要なわけではありません。特定の機能を実装するかどうかは、明確なビジネスニーズによって決定され、最終的な結果は参照例とは大きく異なる可能性があります。

データアーキテクチャ	データセキュリティ	データ品質
記述式	診断	アドバンスド・アナリティクス
報告	ダッシュボード	セルフサービス
運用データストア	データウェアハウス	データレイク
データモデル規格	エンタープライズ次元モデル	データマート
リファレンスデータ管理	データエンリッチメント	マスターデータマネジメント
データパイプライン	データトランスフォーメーション	リアルタイムデータ
データのエクスポート	データ出版	データ製品
データタクソノミー	データカタログ	データポータル

データバリューピラミッド

私たちは、すべてのGitLabチームがデータバリューピラミッドを上（下の図では左から右）に移動し、基本的なメトリクスやカウントを知恵に変えて、お客様のためにより良い製品を作り、ビジネスをより効率的に運営し、ビジネスモデルに新しい機能を追加するのを支援したいと考えています。データバリューピラミッドでは、現在、主に「データと情報」の段階で仕事をしています。



データケイパビリティモデル

データケイパビリティモデルは、ギットラボの戦略をサポートするための目標状態の要件を特定するために使用されます。

GitLabが上場企業になるためには、Lead-to-CashとPublic-Facingの指標がケイパビリティモデルのレベル2に達する必要があります。

		レベル特性ベネフィット
(5)Prescriptive	製品に組み込まれたリアルタイムの複合的な分析、行動や認識の形成、データ分析は戦略的な差別化要因である。	新しいデータ製品、意思決定のROI向上
(4)Predictive	データサイエンス」 起こりそうなことを洞察し、広めていくとエフォートレスアナリティクスの生産、エンタープライズデータの品質とガバナンス	信頼できる顧客生涯価値。拡大・解約予測、製品組み込み型アナリティクス
(3)Strategic	広範囲かつ容易にドリルダウン可能な分析、ドリルダウン可能なクロスファンクショナル・スコアカード、ダッシュボード、エンタープライズ・データ・ウェアハウス	カスタマー360&ヘルススコア予測可能で信頼できるデータレポーティング、堅牢なセルフサービス、データ・アット・スケール
(2)Advanced: 参考文献	運用可能な自動化されたレポートとダッシュボード、自動化されたテストによる信頼性と検証されたデータ、手動および自動化された統合の混合、一部のデータサイロを含むコア統合データ	信頼できるデータ、セルフサービスのデータ、重要なパフォーマンス指標、拡張のための安定したプラットフォーム

		レベル特性ベネフィット
(1)リアクティブ	静的なリストやレポート、履歴やラグ（過去**30**日、**90**日、**365**日）に強い関心がある、予測不可能な速度、最小限のクロスファンクショナル分析、データサイロ	ヒストリカル・タブラル・レポート、データ・ビジュアライゼーション
(0) なし	レポート作成に一貫性がない、結果が広く信頼されていない、安定した分析インフラがない	

layout: handbook-page-toc title:"データチャンピオンプログラム" 説明"Discover how GitLab uses a Data Champion Program in concert with the Data Team to promote data literacy and acumen"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

このページには、将来の見通しに関する内容が含まれており、現在の状態または予定されている機能セットや能力を正確に反映していない可能性があります。

概要

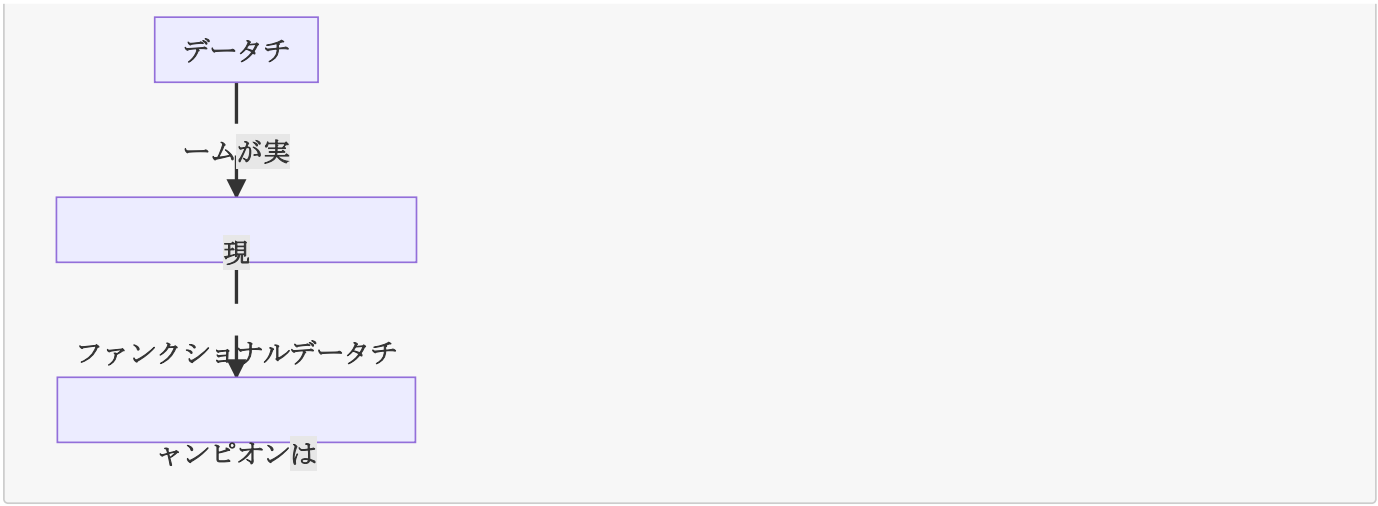
データチャンピオン・プログラムは、既存のチームメンバーで組織全体のデータ能力を開発するための最新のソリューションです。すべてのデータニーズを中央のデータチームに依存するのではなく、データチャンピオンプログラムは、中央のデータチームの専門家が作成した専門知識、コンテンツ、プログラムを活用して、機能別チーム内でデータを開発するインサイドアウト、トレーナー育成のアプローチを採用しています。

データ・チャンピオン・プログラムの目標は以下の通りです。

- データを会社の重要な資産として活用し、全員がデータを使って最善の意思決定ができるようにする。
- セルフサービス・ダッシュボードの開発や**SQL**分析など、機能チームがデータを自給自足できるようにするためのパスの作成
- データ駆動型の技術やアプローチの導入を促進する データの質とデータの説
- 明責任を向上させる
- 活発な**Data**コミュニティの発展に貢献する

アプローチ

データチャンピオン・アプローチでは、各機能別チームの中から、**(a)**そのチームのデータ**DRI**として、**(b)**データチームの安定したカウンターパートとしての役割を果たす個人を特定します。データチームのリーダーとデータチャンピオンは共同でデータサクセスプランを策定し、ビジョン、責任、目標を明確にし、機能チーム独自のデータニーズ、適性、既存のデータ知識を考慮します。この計画では、**GitLab**データ資産（レポート、ダッシュボード、データモデル）、セルフサービス機能、データロードマップをカバーするトレーニングスケジュールを確立します。データチャンピオンは、チーム内でのデータ能力の向上を支援するために、地域のデータコンピテンシーセンターを通じてデータプラクティスを促進する責任があります。



ファンクショナルチームメンバー

データチャンピオンの責任

1. データチームとの連携によるデータサクセスプランの策定
2. 機能別チームのデータDRIとして、データ品質問題を含むすべてのデータ要求を把握する。
3. データイシューとエピックの優先順位付けと積み重ねを行い、データイシューボードを最新の状態に保つ。
4. イシューボードを通じてデータチームに優先事項を伝える
5. プロダクトマネージャーのためのデータ」など、各チームの対象者に合わせたデータハンドブックページの作成
6. データに関するニュースやプラクティスを定期的に伝え、カスケードしていく
7. ソースシステムのデータ品質向上に貢献
8. ソースシステムの変更を監視し、必要な更新をEnterprise Data Warehouseに組み込む手助けをする

要求事項

- 週に1〜2時間程度のプログラムへの参加（取り組みの進展に応じて時間を調整
- ） 個人およびチームのデータコンピテンシーの向上への取り組み 中級SQLの経
- 験
- データビジュアライゼーションの経験
- データモデリングの経験；Zuoraの例、Salesforceの例

プログラムターゲット

	エリア現在の状態	ターゲット
の状態データ機能チームの優先順位付け	Data TeamData Champion, supported with Value Calculator	
ファンクショナルチームのデータイシューボード	データチーム	データチャンピオンとデータチームの維持
機能チームのデータDRI	Does Not Exist（	存在しない）データチャンピオンは、所属する機能別チームのデータDRIとして活動します。
*機能*チームハンドブックのデータ	ページが存在し	ないデータチャンピオンとデータチームの維持

	エリア現在の状態	ターゲット
の状態通信する機能チームへのデータニュース	定期的なコミュニケーション席し、データを渡します。	はありません データチャンピオンは、データチームのミーティングに定期的に出 機能チームに関連するニュース
Sisense トレーニング	オンラインチュートリアルとデータチームによる定期的なライブトレーニング	データチャンピオンが機能チームのためにカスタマイズし、必要なトレーニングを提供します。
データの収集と処理	データチームは、すべての**Handbook**ドキュメントを管理し、**dbt**ドキュメントの更新をサポートします。	データチャンピオンは、ソースシステムの品質やモデルの変更に関する最新情報を入手し、必要な更新情報をデータチームに伝達して実装する（ハンドブックの維持を含む
データのプライバシーと保護	プライバシー審査を担当するデータ、法務、セキュリティ、コンプライアンスの各チーム	データチャンピオンは、データ、法務、セキュリティ、コンプライアンスの各チームが、プライバシー評価を必要とする可能性のあるデータ要素を理解し、適切な文書を維持できるように支援します。

layout: handbook-page-toc title:"データプログラムレベル2リファレンスソリューション" このページについて

{:.no_toc}

- **TOC {:toc}**

目的

このページには、将来の見通しに関する内容が含まれており、現在の状態または予定されている機能セットや能力を正確に反映していない可能性があります。

公開企業は、主要な財務、顧客、成長の指標を確実かつ予測可能な形で共有するとともに、ビジネスパフォーマンスを継続的に改善するために、リードタイムからキャッシュタイム、製品のアイデアから採用までのプロセスを分析する必要があります。これらの活動は、データ機能モデルのレベル**2**で定義された機能によって支えられています。このページでは、現実的な例を示し、今後の開発の参考とするために、「製品ジオロケーション分析」のレベル**2**データソリューションを紹介します。

ソリューション概要 - 製品ジオロケーション分析

自社の製品が世界中のどこで使われているかを理解することは、顧客や製品のグローバル展開をより深く理解するための重要なステップであり、ロケーションを意識したインサイトを構築することにつながります。

このデータソリューションは、**3**つの**Self-Service Data**機能を提供します。

1. ダッシュボードビューアー - 国、地域、月、年ごとの**GitLab**デプロイメントを視覚化する新しい**Sisense**ダッシュボード（セルフサービスダッシュボード）です。
2. ダッシュボード開発者 - 新しいダッシュボードを構築し、既存のダッシュボードをジオロケーションデータにリンクさせるための完全な次元モデルコンポーネントを含む新しい**Sisense**データモデルです。
3. **SQL Developer** - 新しい**Enterprise Dimensional Model**の対象領域

データプラットフォームの観点から、このソリューションは実現します

。

1. **GeoLocation**分析のための**Enterprise Dimensional Model**の拡張機能です。
2. 新しい`dim_country`テーブル
3. **Data Pipeline Health**ダッシュボードのテストおよびデータ検証拡張機能
4. **ERD**、**dbt**モデル、および関連するプラットフォームコンポーネント

最後に、これは過去**1**年間に完了した以下のようないくつかのアドホックな問題に対する長期的な自動化ソリューションで

-

す。顧客とユーザーのデータを国別に抽出

- 各国ユーザーデータ

- SheetloadでGeoLite2フリーデータベースをアップロード

知識評価と証明書

Self-Service Data 認定プログラムは、Learning and Development 認定プログラムに基づいています。Self-Service Dataプログラムでは、対象となるDashboard DeveloperまたはSQL Developer Knowledge Assessmentを完了した場合に、個別の証明書が発行されます。ナレッジ・アセスメントへのリンクは、以下の各セクションにあります。

データの分類

ORANGE

- IPアドレス

ソリューション・オーナーシップ

- ソースシステムのオーナー@rparker2
- ソースシステムのサブジェクト・マター・エキスパート。@jeromezng
- データチームのサブジェクト・マター・エキスパート。@rparker2

主な用語

- 地域 - データはGitLab Sales Territoriesを使って可視化されます。AMER, APAC, EMEA
- Country - データはISOCountryによって可視化されます。
- IP-ジオロケーションマッピング - IPアドレスを地図上の地理的位置にマッピングすること。

キーメトリクス、KPI、PI

- 国または地域別、月別、年別の使用Ping数
- 国または地域別の GitLab.com ページビュー数（月別、年別） KPIs - 定義な
- し
- PI - 定義されていない

セルフサービス・データ・ソリューション

セルフサービス・ダッシュボード・ビューアー

	ダッシュボード目的
世界の製品の	成長国、地域、時間ごとのGitLabの採用状況を可視化します。
Geolocation	DataのData Health Dashboardこのソリューションをサポートする
ために使用されるGeolocationデータの	データヘルス。

セルフサービス・ダッシュボード開発者

データスペース	説明
Global	以下に示す「製品ジオロケーション分析」モデルと1-1の関係を持つデータモデルを含みます。

セルフサービス・ダッシュボード開発者認定証

証明書を取得するには、Self-Service Dashboard Developer Knowledge Assessmentで100%の評価を得て、新しいダッシュボードのスクリーンショットをアップロードする必要があります。Knowledge Assessmentを完了すると、回答がメールで送信され、このメールが証明書となります。

セルフサービスのSQL分析

キーフィールドとビジネスロジック

- データはusage-pingとsnowplowから取得しています。
- Usage-pingは、ホストのユーザーがどこから来るかではなく、インスタンスがどこにHOSTされているかについての情
- 報を含みます。Snowplow は、ユーザーがどこから来ているかについての最良の情報を含んでいます。ユーザーの中にはVPNや同様のプロキシ・ソリューションを使ってGitLabを導入している人もいるので、これらのソースから得られるIPとジオロケーションのマッピングは正確ではないかもしれません。しかし、Snowplowは私たちが持っている最高のデータソースであり、私たちの最高の真実の情報源でもあります。

エンティティ・リレーションシップ・ダイアグラム

Diagram/EntityGrain	目的	キーワード
製品 ジオロケーション分析	Activity By 日	分析に使用できるディメンションとファクト 国別、地域別、時間別のGitLab利用状況。	dim_date, dim_country, fct_country_activity_by_day
DAY	DIM_DATED	すべての日付の中心となるディメンションです。	
	dim_countryISO	CountryCentral すべての国と地域のためのディメンションです。 ISO-3166とGitLab Sales Territoriesをソースとしています。	

リファレンスSQL

SisenseやdbtのすべてのプロダクションSQLは、読みやすさと保守性のためにSQLスタイルガイドを遵守しなければなりません。

NORAMの日別国別ページビュー

```
SELECT
  f.date_key            AS date_key,
  dc.country_name       AS country_name,
  SUM(f.num_page_views) AS number_of_page_views
FROM fct_country_activity_by_day f
JOIN dim_country dc
  ON f.country_key = f.country_key
WHERE dc.reporting_region = 'NORAM'
```

2020年の国別トップ100ネームスペース

```
SELECT
  dc.country_name       AS country_name,
  f.namespace_key       AS namespace_key,
  SUM(f.num_page_views) AS number_of_total_page_views
FROM fct_country_activity_by_day f
JOIN dim_country dc
  ON f.country_key =
f.country_key WHERE dd.year =
2020
GROUP BY dc.country_name, f.namespace_key
PARTITION BY dc.country_name LIMIT 100
```

セルフサービス型SQL開発者認定証

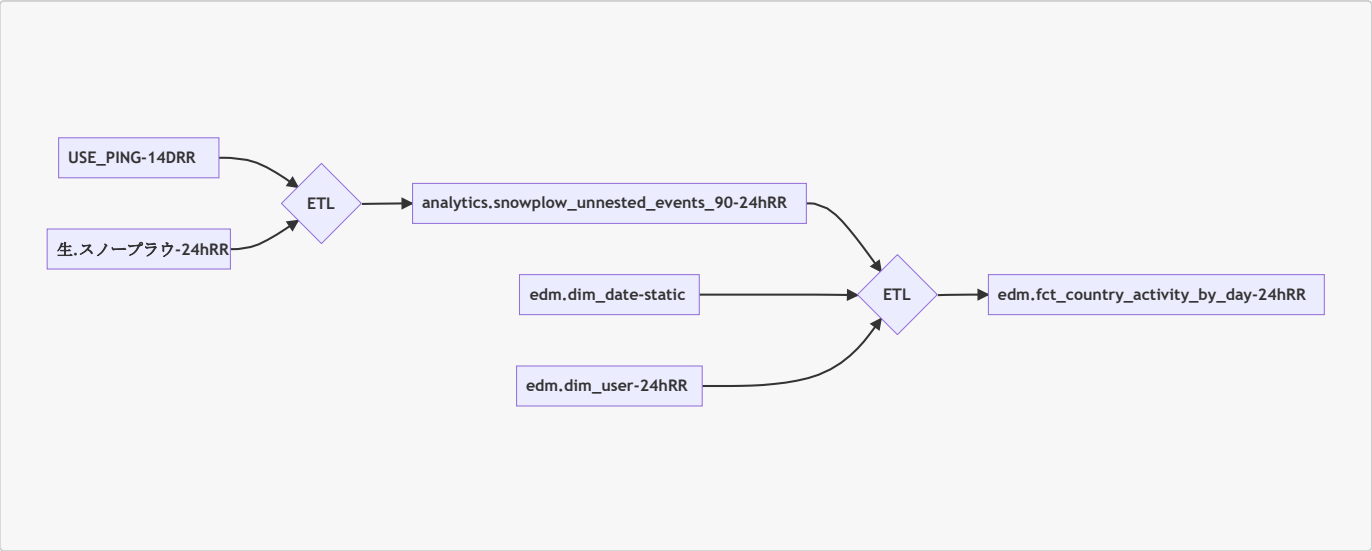
証明書を取得するには、Self-ServiceSQLDeveloperKnowledgeAssessmentで100%の評価を得て、新しいSQL文のスクリーンショットをアップロードする必要があります。知識評価が完了すると、回答がメールで送信され、このメールが証明書となります。

データプラットフォームソリューション

全体的なソリューションは、当社のEnterprise Dimensional Modelガイドラインに準拠しています。

データの系統

完全なリネージグラフについては[dbtのドキュメント]を参照してください。



DBTソリューション

[dbtソリューション]は、RAWのソースデータから次元モデルを生成します。

トラステッドデータソリューション

エンタープライズ次元モデルの検証

バリデーション	予想される結果
1	マップされた国の総数が300を超えないこと。
2	APACからのトラフィックの割合がAMERを超えないこと。
3	>40,000 # fct_country_activity_by_day の行が新たに追加されました（日別）。

ソースデータパイプラインの検証

バリデーション	予想される結果
1	直近14日間で、新しいUsage_pingデータがアップロードされました。
2	Usage_pingデータで表されるアカウント数の合計は、*予想される結果*よりも多い。
3	除雪機のデータで表されるアカウント数の合計は、*予想される結果*を上回ります。

---

layout: handbook-page-toc title:"Self-Service Data"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

概要

このページには、将来の見通しに関する内容が含まれており、現在の状態または予定されている機能セットや能力を正確に反映していない可能性があります。

データ民主化は、データチームにとって共通の目標ですが管理すべきデータの多様性、量、速度、正確を考えると、達成するのは難しいかもしれません最終的には、効果的なデータ民主化ソリューションはすべて、データ顧客に焦点を当て、データを見つけやすく、理解しやすく、実用的なものにするソリューションを提供する必要があります。

- *簡単に見つけられる*こと。データ お客様が自分に関連するデータを簡単に見つけられること
- が*理解しやすいこと*。データは、ビジネスに適した用語や概念を用いて、一貫した方法でラベル付けされ、整理されている必要があります。
- *アクション可能であること*。意思決定や行動を支援し、結果を出すためのインサイトを提供する、または支援するデータであること。

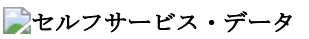
GitLabセルフサービス・データプログラム

GitLab Self-Service Dataプログラムは、信頼できるデータを使って**GitLab**がより速く行動できるようにするとともに、データチームが拡張できるように、**4**つの機能を提供します。

- データカタログ - データカタログは、データカスタマーが対象分野に基づいてデータ定義、ダッシュボード、モデルを見つけるのに役立ちます。
- ダッシュボード・ビューアー - すべての**GitLab**チームメンバーは、**Okta**からアクセスできる常時接続のポータルを通じて、事前に構築されたダッシュボードに*アクセスできます*。
- ダッシュボード開発者 - **GitLab**チームメンバーで、独自のデータ可視化チャートやダッシュボードを作りたい人向け
- SQL開発者 - **GitLab**チームメンバーで、**SQL**に精通しており、**SQL**ベースの分析を行いたい人向け

セルフサービス機能の概要

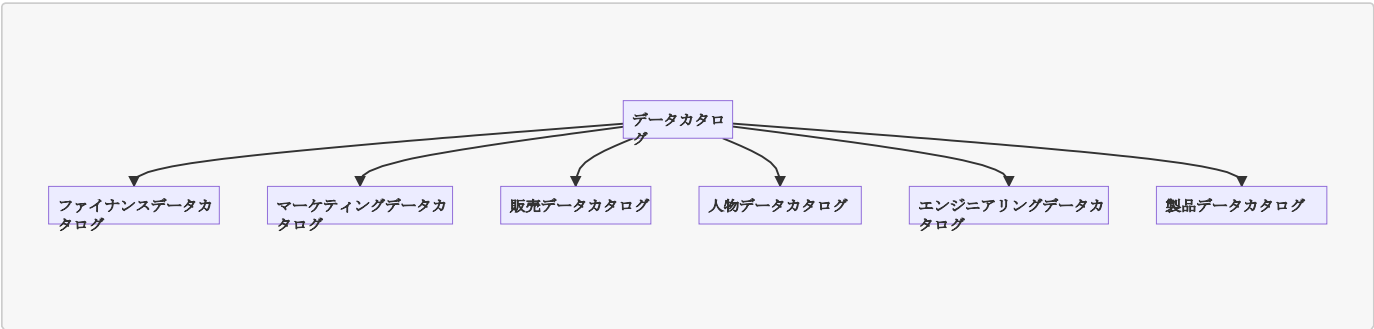
	ダッシュボードビューア	ダッシュボード開発者	SQLデベロッパー
アクセスデータカタログ	はい。	はい。	はい。
既成の**Sisense**ダッシュボードにアクセス	はい。	はい。	はい。
新しい**Sisense**ダッシュボードの構築	いいえ	はい。	いいえ
SQLを書いてデータを調べる	いいえ	いいえ	はい。
認証が必要	いいえ	はい。	はい。

 セルフサービス・データ

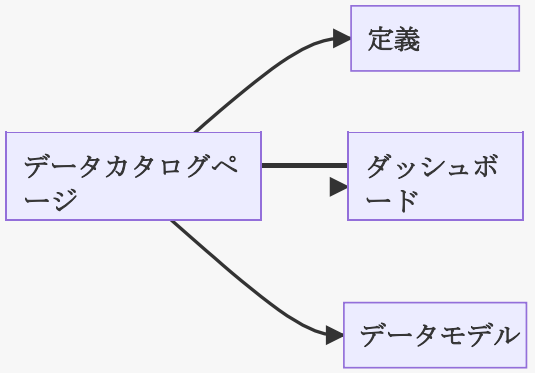
データカタログ

データカタログはハンドブックに掲載されており、機能ごとに整理されたデータダッシュボード、モデル、定義のインデックスが含まれています。ハンドブックの一部として、データカタログは、共通の基準と単一のソース・オブ・トゥルース（真実の）データへの調整を促進します。データチームのハンドブックセクションの一部として、データカタログは関連するコンテンツで定期的に更新されます。

データカタログ構造



データカタログハンドブック ページ内容



セルフサービス・ダッシュボード・ビューアー

セルフサービス・ダッシュボード・ビューアーは、GitLabチームのメンバーがSisenseで公開されている事前に作成されたダッシュボードにアクセスできるようにします。前提条件として

- ☐ **None**

Access Requests:

- ☐ **必須ではありません - すべてのGitLabチームメンバーは、オンボーディング時にアクセス権が与えられます。**

セルフサービス・ダッシュボード開発者

セルフサービス ダッシュボード デベロッパーは、認定GitLabチームメンバーに、Enterprise Dimensional Modelを利用して明確に定義され、検証されたKimball標準のデータモデルに基づいて、Sisense Data Discoveryを使用して独自のダッシュボードを構築する機能を提供します。

前提条件

- ☐ データモデルの読解力
- ☐ レポートやチャートのデータビジュアライゼーションを開発した経験がある
- ☐ 方 レビュー Sisenseトレーニング
- ☐ ハンドブックに掲載されている「Self-Service-ready Data Subject Areas」の中から、「Self-Service Dashboard Developer Knowledge Assessment」に合格する。

アクセスリクエスト。

- ☐ 新規アクセス・リクエストを開き、シングル・パーソン・アクセス・リクエストを使用する
- ☐ 完成した知識評価へのリンクを含める Snowflakeへのア
- ☐ クセスを要求する

セルフサービスのSQL開発者

Self-Service SQL Developerは、認定GitLabチームメンバーにSnowflakeデータウェアハウスへのSQLプロンプトを提供します。現在、すべてのデータが標準的な形式でモデル化されているわけではないため、調査を行ったり、「RAW」データやモデル化されていないデータを読んだり、経験豊富なチームから学んだりするために、追加の時間が必要になる場合があります。データチームがEnterprise Dimensional Modelを構築し、SQLの例やデータモデルを含むレベル2ソリューションを追加で提供することで、SQL開発者の生産性が向上し、SQL分析がより簡単かつ迅速に行えるようになります。

前提条件

- ☐ 中級SQL
- ☐ データディクショナリをナビゲートする能
- ☐ 力 データモデルを読んでSQLを書く能力
- ☐ Data Team SQL Style Guideの遵守
- ☐ ハンドブックに掲載されている「Self-Service-ready Data Subject Areas」の中から、「Self-Service SQL Developer Knowledge Assessment」に合格する。

アクセスリクエスト。

- ☐ 新規アクセス・リクエストを開き、シングル・パーソン・アクセス・リクエストを使用する
- ☐ 完成した知識評価へのリンクを含める Snowflakeへのア
- ☐ クセスを要求する

---

layout: handbook-page-toc title:"データチームドキュメンテーションガイド" 説明"このドキュメンテーション方法は、現在データチームでは使用されていませんが、異なるタイプのドキュメンテーションをどのように考えるかの良いガイドとなっています。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

このドキュメント作成方法は、現在データチームでは使用していませんが、さまざまなタイプのドキュメントをどのように考えればよいのか、よい指針となります。

ペルソナ

データチームのハンドブックには、たくさんの情報が詰まっています。すべての情報が、すべての人のために書かれているわけではありません。社内でも出身地が違えば、求めている情報も違います。

読者の皆様に参考にしていただくために、大多数のユーザータイプを想定した4つのペルソナを作成しました。ペルソナは以下のとおりです。

ビルダー🛠

{:#builder}には

ビルダーの主な属性

- データのクエリと変換の両方で強力なSQL能力を持っている 優れたデータ運用
- 能力を持っている
- データへの直接アクセス（Snowflakeの
- Rawデータ） 強いドメイン知識
- ドキュメンテーションや分析の貢献者／協力者 典型的な役割

  データエンジニア
  コアデータアナリ
- スト
-

ユーザー💻を

{:#user}になります。

ユーザーの主な属性

  データを照会するための低〜中レベルのSQL能力
- 基本的なデータの流暢性
- データへの直接アクセス（Snowflakeの
- Rawデータ） 強いドメイン知識
-

典型的な役割です。

  ディストリビューション・ア
- ナリスト/エンジニア・プロダ
- クトマネージャー

- エンジニアリング・マネージャー

コンシューマー📊

{:#consumer}です。

消費者の主な属性

- **限られたSQL能力 限られた**
- **データ流暢性 強いドメイン**
- **知識**
- **データへのアクセス制限（Sisenseの**

み） 代表的な役割

- **ピープルマネージャー ビ**
- **ジネスパートナー**
- **Non-data / technical individual contributor**

チャンピオン🏆

{:#champion}になります。

チャンピオンの主な属性

- **限られたSQL能力と限られた時間**
- **ドメイン知識は豊富だが、分析的洞察は直属の部下に頼る データへのア**
- **クセスが限られている**

典型的な役割です。

- **取締役/VP/CXO インベ**
- **スター**

みんな🌎

{:#everyone}になります。

これは、すべてのペルソナに関連するものがある場合のための、包括的なグループ分けです。

ドキュメントの種類

TheDocumentationSystem」では、ここでは説明しきれないほど詳しく説明されていますが、一般的な考え方として、ドキュメントには4つのタイプがあるとされています。これらはそれぞれ異なっており、異なる目的を持って書かれています。それは以下の通りです。

チュートリアル

{:#tutorials}

のチュート

- **リアルです**
- **ね。**

は学習志向
初めての方でも安心して始められる
ように、レッスンでは

例え：小さな子供に料理を教えること

ハウツーガイド🏁を

-

{:#howto}を参照してください。

ハウツーガイドです。

目標を持っている

-
-
-

-

- 特定の問題を解決する方法は、一連の
- ステップであることを示しています。

類語：料理本のレシピ

説明はこちら💡。

{:#explanation}

の説明です。

- は、理解重視の説明です。
- 背景や文脈を提供する **Analogy:** 料理

の社会史に関する記事 Reference 🗄

{:#reference}になります。

リファレンスガイドです。

- は、情報を重視し、機械
- が正確で完全であること
- を説明します。

類語：参考となる百科事典の記事

使用方法

ペルソナタイプとドキュメントタイプの両方のヘッダーの横に絵文字があるのがわかります。この絵文字は、ハンドブックのデータチームの部分で使用しており、読者が各セクションの対象者や目的をすぐに理解できるようにしています。例えば、次のようなものです。

- 🛠🏁 データチームの誰かのためのハウツーガイドであることを示します。例としては、**Snowflake**でのプロビジョニングの方法などです。
- 💻🛤 これがユーザーペルソナの参考資料であることを示します。例としては、**Sisense**で作業するためのヒントやコツのセクションです。
- 🌎💡 全てのペルソナについての説明であることを示します。例としては、データチームの憲章が挙げられます。

絵文字は、そのセクションに興味を持つすべての人を網羅しているわけではないことにご注意ください。私たちの目的は、読者が何を読もうとしているのかを説明するためのクイックガイドを提供することです。

データチームハンドブックへの寄稿

データチームのすべてのチームメンバーは、ハンドブックに貢献することが奨励されています。`@data-team`のみんなは、データチームのハンドブックページのコードオーナーですが、義務があります。

1. すべてのハンドブックの変更はピアレビューされ、ピアが承認を与える前にマージされることはありません。
2. ハンドブック変更のためのすべての**MR**には適切な名前がついていますが、これは**Handbook Changelog**を通じて変更を追跡するために重要です）。)データチームのハンドブックの変更であることを示してください。
3. **MR**が直接影響を与える可能性のあるチームメンバーにタグを付け、チームメンバーがレビューするために**24**〜**48**時間の時間を確保する。ハンドブックへのわずかな変更であっても、少なくとも**1**人のチームメンバーに確認してもらうのが良い方法です。
4. データチームのデモミーティングでは、あなたのハンドブックへの貢献について自由に議論・デモすることができます。

---

layout: handbook-page-toc title: 'GitLab Experimentation Best Practices' description:"実験によって、私たちは学び、正しい体験を顧客に提供し、顧客とGitLabにとってより良い価値を生み出すことができます。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

なぜエクスペリメント・ベスト・プラクティスなのか？

実験によって、私たちは学び、お客様に正しい体験を提供し、お客様とGitLabにとってより良い価値を生み出すことができます。実験は簡単なように見えますが、組織がベストプラクティスに従わなければ、不正確な結論を出すリスクがあまりにも高くなります。

例えば、実験を行う際によくあるミスステップは、不正確な結論や判断につながる可能性がありますので、以下にご紹介します。

1. **ピーキング（記事はこちら）。テスト期間を前もってロックしておかないと、A/Bテストが終わる前に結果を確認して行動を起こすことで副作用が発生するピーキング問題が発生してしまいます。A/Bテストの中間結果を意思決定の準備をした状態で見る回数が多いほど、基準がないのに統計的に有意な差を示してしまう確率が高くなります。**
    1. **2つのピーキングケースはp値を2倍にします。**
    2. **5回の覗き見でp値が3.2倍になる。**
2. **シンプソンのパラドックス: テスト飛行の途中でテストグループの配分を不均衡に変更した場合に発生します。テストグループの配分比率を変更すると、テストグループ内の潜在的なセグメントの比率が変わり、結果に誤差が生じます。より正式には、シンプソンのパラドックスとは、ある母集団における2つの変数の間の関連性が、その母集団を小集団に分けたときに現れたり、消えたり、逆転したりする統計現象のことです。**

実験方法から得られる価値を最大化し、GitLabでの決定の不正確さを減らすために、GitLabでのすべての実験においてベストプラクティスに従うことを推奨します。このドキュメントでは、Gitlabで採用すべきベストプラクティスの概要を説明します。

実験の計画

仮説を立てる。アップサイドの定義

実験を始める前に

- **改善しようとしているビジネスチャンスの定義**
- **どのようにしてビジネスチャンスを拡大しようとしているのか、なぜそのようなアプローチを取りたいのかを説明してください。**
    裏付けとなるデータや分析があれば、それを文書化する。
コントロールのUIとフローを文書化し、ワイヤーフレーム/コンセプトをテストする

目標。目標がなければテスト測定は意味がない

- **実験で動かしたい主要な指標を定義します。これは、ロールアウトのシナリオを定義するために使用する指標です。**
    - **理想的には、1つの指標だけであることが望ましい（最大2つの指標）。**
    - **理想的には、ビジネス上のKPI（例：ARRの増加、有料ユーザーのサインアップなど）**
    - **に結びつけるべきです。 サンプルサイズを計算するために、予想される変化の大きさと方向を定義します。**
    - **テストのワンテール/ツーテールと期間を決定 → サンプルサイズに基づき、KPIが変化すると予想される方向性に基づく。**
    - **例**
        - **テストコンセプト新規ユーザー登録時のステップ数の削減**
        - **主な指標はまた、有料ユーザー数などの収益につながるKPIに結びつけることが理想的です。**
- **副次的なメトリクスを定義する。一次指標がなぜ、どのように影響を受けたのかを検証し、詳細に理解するための指標があります。**
    - **二次指標は、一次指標の動きが本物かどうかをクロスバリデーションするのに役立ちます。**
    - **5%や10%という高いアルファ値を使用しているため、一次指標への影響が偶然のものである可能性があり、二次指標は一次指標の変化が本物かどうかを検証するのに役立ちます。テストのケイデンスが改善されれば、この点はより重要になるでしょう。**

        - 例
            - テストコンセプト。新しいユーザー登録フローに、より目立つトライアル登録ボタンを導入する。主
            - な目標D90コンバージョン率の向上。新規ユーザー登録の際にトライアルに申し込む人が増えること で、製品の価値をより深く理解してもらい、D90コンバージョン率を5%向上させる **ノイズの可能性
            - ：主要な指標が5%向上しても、トライアルに申し込む新規ユーザーの数が増えなかった場合、D90コンバージョンの向上はノイズである可能性があり、さらなる検証が必要です。
- **ガードレールの指標を定義する。これは、短期的な利益のために長期的にはビジネスを害することがないようにするためのものです。
  **
    - これらは、実験がもたらす長期的な影響を追跡する指標に基づいています。例
    -
        - 新規ユーザーの登録フローでトライアルを増やす実験をする場合、D90コンバージョンやトライアルから有料へのコンバージョンをガードレール指標として設定するとよいでしょう。
        - もうひとつの例は、あるソーシャルメディア企業で、「通知無効率」をガードレール指標として定義しています。プッシュ通知の増加によりセッション/DAUが1%上昇するごとに、「通知無効率」の増加はX%以内に収まるべきだというガードレールを設定しています。
        - 別の例。あるSaaS企業では、ガードレールの指標として12ヶ月間の解約率を設定し、12ヶ月後に分析を再検討することがあります。

プレアナリシス：前もって計算しておく

- データを用いて問題、機会、アプローチを検証する（例：記述的分析
    - 例特定のメールタイプのメールヘッダーコピー変更をテストしても、このメールのボリュームが少なければ、有意な価値はないかもしれません。有意性を達成することはできないでしょう。このような場合には、事前・事後テストを行うか、よりトラフィックの多いメールタイプを選んでテストを行うことを検討してください。
- 片側／両側、α（5%）、検出力（80%）を用いて、必要なサンプルサイズを算出する。実験期間（どのくらいの期間実験を行うか）を前もって決めておく。
    - ロールアウトの決定に高いアルファ値を使用する必要があるかどうかを事前に判断します。
- 実験を行うための費用対効果の分析を行う。組織内で実験のコストが高く、技術リソースが限られている場合、正しいアイデアを優先させるために、それぞれのテストから得られる潜在的なアップサイドを理解できなければなりません。
    - 会社のKPIの1つとして、この変更による年間のアップサイドの可能性を計算する。それが不可能な場合は、KPIへのアップサイドをログ値で定義し（0.01%, 0.01%, 0.1%, 1%, 10%, 100%インパクトのように）、異なるアイデアの間で大まかな比較ができるようにします。
    - 段階的な機能変更と段階的な変更の推進

実験のセットアッププラン。テストのセットアップ方法

- 実験から得たいビジネス上の疑問に答えるために必要なバリエーションの数を決定する（例：A/B/nまたはMVT（多変量解析）のセットアップ
- どのような仮説の質問に答えられるか、答えられないかを含む分析計画を文書化します。
- 必要な機器の定義（既に利用可能なデータに加えて、追加で必要なトラッキングデータ）。
    - 開発者が新しいトラッキングについて議論し、労力やパフォーマンスへの影響を考慮して実装できるように、トラッキングのニーズを「重要」または「あったらいいな」と指定する。
- セグメンテーションや除外基準を含む、テスト対象となる母集団の定義
- いつ実験を行うかを実験カレンダーに記載し、競合をチェック クリティカルパスで実験を行う場合や、大規
- 模なユーザーに影響を与える場合には、立ち上げ時の重みや立ち上げプランを定義する
    - 重要な分野でリスクの高いテストを行う際には、1%から5%/10%、そして50%へと段階を踏んでいきます。
        - シンプソンのパラドックスに注意して、重量配分が比例して比較可能なテスト期間のみを分析してください。
    - ランピングの成功基準の指定
        - 例プライマリー指標に大きな影響はなく、5%程度の変化でも検出できる（＝感度）。

リスクとクリティカル・パスに基づくテストグループのサンプル配分に関する一般的なガイダンス

失敗のリスク、機能や影響を受ける人々の重要性を考慮した実験打ち上げの重み付けマトリックス	クリティカル・パス・ページまたはクリティカル・プロダクト・フィーチャー

		LMH
	故障時の潜在的なコードリスク	
	L50	
		10%
		5%
M		
		10%
		10%
		5%
H		
		10%
		5%
		1%

実験検証。

Experiment Validation プレローンチ。すべてが整っているか確認

- 開発チームとデータエンジニアチームが機能を構築した後、開発環境または本番前の環境で、レポートデータとUIが意図したとおりに動作するかどうかを検証します。
  - 推奨：開発、QA、アナリストのうちいずれか2名にトラッキングとレポーティングを検証して
  - もらう 推奨：PM、QA、アナリストのうちいずれか2名にUI機能を検証してもらう
- コントロールとテストエクスペリエンスのスクリーンショットを、今後の参考のために実験ドキュメントに貼り付けます。

Experiment Validation ポストローンチ。期待通りの結果が得られているかを確認する

- 本番開始から1〜2日後、実験の報告データが有効かどうかを確認する 母
  - 集団の割り当てに偏りがないかどうかを確認する
- 1%の重さでスタートした場合、2〜3日後に最初の読み込みを行い、事前に取り決めた計画通りに5%や10%の重さに設定します。
  - 1%の実験の目的は、あくまでも「モノが壊れない」ことであり、結果を読み取ることではありません。
- 5%や10%で立ち上げた場合は、当初の計画通り50%にウェイトを変更する
  - 平日と週末では顧客層が異なる可能性があるため、週単位の季節性がある場合は1週間単位、月単位の季節性がある場合は1ヶ月単位で分析してください。
  - 1%のテストは、実験的な経験によって生み出された影響を読み取ることを目的としていないため、上記は1%には適用されません。

実験分析とコミュニケーション。

- 課題や実験データに偏りがないことを確認する
  - 可能であれば、これらの検証を自動化する 割り当ての母集団バイアス 統計的有意性を
- 用いたテスト結果の分析
  - セカンダリーメトリクスを活用して、テストが成功した理由や方法を検証し、より理解を深めます。
  - 展開シナリオを理解し、人口コホートによるパフォーマンスの有意差を見つけるために、主要なディメンション別に結果を分析する。
    - 多重比較を行うと、統計的ノイズが増えるので注意が必要です。そのため、ディメンションレベルの分割では分析前に仮説を立てるか、修正P値を使用してください。
- このテストと同時に実施された、事前に特定された相互作用実験との相互作用結果を分析します。
- 統計的に信頼できる結果を伝える。

    - 有意な影響があった場合、信頼区間（@80%）をつけて結果を伝える。
    - 有意な影響がない場合は、一般の人々がどの程度の観察された変化率で、統計的に有意な結果と結論付けられるかを理解できるように、感度を持って結果を伝える。
- 結果をハンドブックに記載し、テストの詳細な説明、ランプの概要、主要な結果、メトリクスのスナップショット、インサイト、次のステップを記載します。
- 結果をパートナー（技術部門およびビジネス部門）および分析仲間に提示し、追加の洞察を集め、学習内容を他の人に伝えます。フォローアップの分析と洞察を文書化する。
- PMのために最終的な結果に基づいて、実験がロールアウトされたり、コードから破棄されたりすることを確認します。

実験から得られた知見を制度化する。

- ビジネスパートナーと一緒に、詳細な結果をより広い組織のグループで発表し、他のチームがどのように学習から利益を得ることができるか、価値を最大化するために他のチームがどのような行動を取ることができるか、協力することができるかについての会話を促進します。
- 失敗した同じアイデアを繰り返さないために、すべての結果を文書化し、機能や製品タグで検索できるようにする。これにより、複数の実験からメタアナリシスを行い、より幅広い洞察を得ることができます。
    - 例）類似企業において、過去の20数例のデータを活用して、通知タイプの増加と通知無効化の関係を把握した。
    - 例比較対象となる企業では、過去の実績（〜50件）のメタデータを用いて、異なるメールタイプのボリュームを増やすことによるセッションの上昇を把握しました。
    - 例：同業他社では、過去8回の分析結果を活用して、クロスマーチャンダイジングの追加スポットの増分価値を把握しました。
    - Growthチームはすでに、実験の結果を一元化されたハンドブックのページに記録しています。このやり方をGitlab 全体で採用し、一元化したページを作るべきです。

実験のガバナンス。

- 自動通報システム。
    - 自動化された警告システムを有効にして、実験による主要メトリクスへの著しいマイナスの影響を監視します。これにより、結果を覗き見する必要がなくなります。
        - P <= 0.01またはP <= 0.5を2日間連続で使用しています。
    - 実験の結果、一貫して有意なポジティブな結果がx日間継続した場合、自動アラートを有効にする（ランダムウィンを減らすために3日間とした）。
- 長期にわたる実験を定期的に見直し、製品のパフォーマンスに影響が出ないようにする。
- セットアップ、トラッキング／データの失敗、誤った実装、コンフリクト／実験の相互作用などによる実験の失敗を記録する。これにより、実験プラットフォームの健全性を監視することができます。
- 実験カレンダーを作成することで、コンフリクト管理を可能にし、実験の量と速度を把握することができます。

オポチュニティ・アクション・アイテムを検討する。

注）短期とは、数時間/数日分の努力を意味します。長期的には、数週間、数ヶ月分の努力が必要です。

現在の実験プラットフォームの機能に関する初期のフィードバックと観察に基づいて、これらは私たちが検討することを推奨するアクションアイテムのリストです。ギャップがまだ存在し、その優先順位を確認するために、影響を受ける関係者と協議する必要があります。

- サンプルサイズの要求を検出力計算で標準化する。(優先度H)
    - 短期的にはアルファ（5%または10%）、片側/両側、検出力80%の標準的なサンプルサイズ計算機を定義する。
- 結果のダッシュボードに有意性と信頼度を確実に反映させる（優先度H
    - 短期的にはZ検定の公式やPythonの関数を使って、Sisense実験分析フレームワークに直接、有意差計算と信頼区間を組み込む。
- P値のしきい値を調整して、テスト結果を次元別に分割する機能。(優先度M)
    - 長期的には実験のメトリック/ファネル/ディメンションを自由に評価し、ディメンションやフィルターによる分割分析を可能にするための代替ツールやデータトラッキングを検討する。
- サンプルサイズの問題を解決する（優先度H
    - 短期的には実験のサンプルサイズが小さくても対応できるように、実験のサンプルサイズ分析を片側または両側で定義する。

短期的にはセカンダリー指標を活用して、プライマリー指標での方向性の読み取りに自信を持つ。

-
  -
-
  -
-
  -
-
  -
  -

    - 短期的には十分なサンプルを収集した後に実験を中断し、後にガードレール・メトリックへの影響を測定することで、ガードレール・メトリックを長期的な影響測定に使用する。
    - 長期的。ベイズ実験解析フレームワークを活用し、サンプルサイズが小さい問題に対するデータの不確実性を伝える。
- テスト結果のバイアス検証の自動化。**(優先度M)**
    - 短期的には実験結果に重みや母集団の偏りがある場合に、自動的にフラグを立てる方法の検討 長期的結果を
    - 分析する際に、自動化されたバイアス警告を導入する。
- 勝った実験と負けた実験の自動アラート。**(優先度L)**
    - 短期的には負けている実験や常に勝っている実験について、**P**値や結果が有意に残っている日数に基づいて自動化されたアラートを作成する方法を検討する。
- 実験内および実験間の無作為化に偏りがないか分析する。**(優先度M)**
    - 短期的には既存のシステムの限界に基づいて、実験内および実験間の無作為化バイアスを検証する必要があるかどうかを検討し、分析する。
    - 長期的なもの実験内および実験間のバイアスを継続的に監視する自動化システムを構築する。
- 実験カレンダー。**(プライオリティーM)**
    - 短期的には現在の実験カレンダーを理解し、実際の実験割り当てデータと比較することで、テストカレンダーの精度を検証する自動化された方法を模索する。
    - 長期的には組織全体で実行されているすべての実験と、実験の期間を記録する単一のシステムがある。実際のテスト割り当てデータに対する自動検証を可能にする。

分析用に作られたツール。

- 検出力と信頼度の変化を利用したサンプルサイズ計算機
- **WIPSisense**実験ダッシュボードへの有意差検定の組み込み（概念実証

リファレンス

- **Growth**の実験ページ。**Growth**はすでに実験結果を一元化したページに記録しています。
- 成長実験ダッシュボード（**Dave**による）は、発売後に実験データの精度を迅速に検証するのに役立ちます。**7**ステップの**A/B**テストプロセス 計画、設計、**QA**に時間を費やす必要がある
- 低トラフィックサイトの**A/B**テスト
- テスト計画の重要性

用語の説明。

- デフォルトの体験。実験に割り当てられていないときに参加者が受ける経験。これはほとんどの場合、対照体験と同じである。
- 実験カレンダー。過去の実験、現在の実験、今後の実験を、実験対象者、ウェイト、実験期間とともに表示するシステム。これにより、コンフリクトの計画を立てたり、相互に作用する実験間の相互作用を分析したりすることができます。
- 感度。与えられた母集団とその指標に対して、有意な閾値を検出できる最小の変化は何か。

---

layout: handbook-page-toc title:"Data Team Calendar - Meetings" description:"GitLab Data Team Calendar"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

ミーティング

データチームのGoogleカレンダーは、チームミーティングのためのSSOTです。また、データスペースの関連イベントも含まれています。このカレンダーには誰でもイベントを追加することができます。このカレンダーに掲載されているイベントの多くは、毎月のキーレビューを含め、出席を必要としない、参考のためのイベントです。データチーム全体のイベントを作成する場合は、Googleカレンダーで勤務時間を確認し、事前に勤務時間外のミーティングについて話し合っておくとよいでしょう。就業時間外のミーティングが必要な場合は、誰がミーティングをするかを交代することを検討してください。

ノーミーティング・フライデー

金曜日に絶対にミーティングをしないとは約束できませんが、チームは *「No Meetings Fridays」* を採用しています。

メーカーのスケジュールの偏り

データチームのメンバーのほとんどは、マネージャーのスケジュールよりもメーカーのスケジュールに強く共感しており、私たちはこれを支持しています。

- 個々のコントリビューターが会議に参加する時間を20%以下に制限すること
- ミーティングを一日や一週間に分散させるのではなく、連続した単位にまとめ、定期的に必要なミーティ
- ングのほとんどを火曜と木曜に行う。

ミーティングプロトコル

チームは*GitLabミーティングのプラクティスとスタンダード*に従っています。

- ミーティングには必ずアジェンダがあります。
- 会議のテーマは優先順位の高い順に並んでいます。
- アジェンダには、トピックや質問に沿ってメモが追加されます。
- すべてのアジェンダは、*データチームミーティング*の共有ドライブに保存されます。

定期的なミーティング

デイリースタンドアップ

データチームのメンバーは、日々のスタンドアップにギークボットを使用しています。これらのスタンドアップノートは、#data-dailyに投稿されます。Geekbotが「今日は何に取り組もうと思っていますか？と聞かれたら、チームメイトが積極的にブロックを解除できるように、具体的な内容を答えてみましょう。Salesforceの作業」ではなく、「`sfdc_opportunity_xf`モデルにオポチュニティのオーナーを追加する」と考えてみてください。Geekbotがメッセージを送ってきたら、すぐに返事をしなければならないというプレッシャーはありません。あなたがその日に取り組んでいることがチームに伝わるような応答をGeekbotにしてください。そうすれば、優先順位が変わったり、必要な追加情報があったりした場合に、チームがあなたの理解を助けることができます。

火曜日 企画・結果発表会

毎週火曜日の14:00-17:00（UTC）には、ビジネスに特化したフュージョンチームとプラットフォームエンハンスメントチームの成果を出すことに焦点を当てています。データチームのリーダーは、ほとんどのセッションに定期的に出席し、すべてのミーティングの議題を確認することが求められます。私たちは2週間のマイルストーン間隔で仕事をしているため、この*火曜日のミーティング*の焦点は週ごとに変わります。

時間(CET/UTC/ET/PT)	ミーティング
1600/1400/1000/700	GTM Fusionチームミーティン
グ1700/1500/1100/800	ProductFusionチームミーティ
ング	
1800/1600/1200/900	データ・エンジニアリング・チーム・ミーティング

DEMOミーティング

毎週木曜日は、データスタックに焦点を当て、開発中のソリューションを紹介しています。

- デモミーティング。ソフトウェアや関連コンテンツのデモを行うことは、ソリューションへの関心を高め、知識を深める

のに最適な方法です。デモには、**SQL**の例、ハンドブックのページ、**dbt**フロー、**ERD**、ダッシュボード、その他の関連コンテンツが含まれます。画面共有やウォークスルーも推奨します。必要に応じて、ビジネス関係者もデモに招待します。

時間 (CET/UTC/ET/PT)	ミーティング
1700/1500/1100/800	データチームのデモ

月例会議

時間(CET/UTC/ET/PT)	デイ	ミーティング
Varies	最初のフルウィーク	オールハンズ&レトロスペクティブ
Varies	先週の金曜日	Handbook Jamboree
Varies	Varies	月刊ピザパーティー

---

layout: handbook-page-toc title:"Merge Request Roles and Responsibilities" description:"GitLab データチーム MR の責務"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

マージリクエストの役割と責任

レビュアー

GitLab データチームのすべてのメンバーは、同僚やコミュニティの貢献者のマージリクエストに対してコードレビューを行うことができますし、そうすることが推奨されています。マージリクエストをレビューしたい場合は、誰かがあなたに割り当ててくれるまで待つこともできますが、公開されているマージリクエストの一覧を見て、フィードバックや質問を残すことも大歓迎です。

なお、すべてのチームメンバーはすべてのマージ要求を確認できますが、マージ要求*を受け入れる*ことができるのはメンテナに限られています。

レビュアーの責任は

- 技術的な実装を確認するために
- コードがビジネス目標を達成するために
- 作成したデータモデルのデータ品質をチェックする

ために Codeowner

コードオーナーシップとは、プロジェクトメンバーとプロジェクト内の特定のフォルダやファイルを結びつけるGitLabの機能です。これは、「このコードについて誰に聞けばいいのか」「このコードの変更を誰がレビューすべきか」という質問に答えるためのものです。

コードオーナーになることは、プロジェクトメンテナーになるための過程の一部です。もしあなたがファイル（例えば新しいdbtモデルセット）の唯一の作成者であれば、あなたはそのファイルの事実上のコードオーナーとなります。オーナーシップの範囲を広げたい場合は、以下の手順に従ってください。

1. CODEOWNERSファイルに、変更したいオーナーシップを記載したMRを作成します。
2. 参加したいエリアをすでにカバーしている他のコードオーナーと協力して、担当したいコードを変更する少なくとも5つのMRでペアを組む。
    - MRは、コードベースの小さな変更だけでなく、アーキテクチャ的なものや、完全に機能する追加機能を生み出すものも反映させる必要があります。この基準に達するには、5つ以上のMRが必要になる場合があります。
    - あなたは、コードオーナーに割り当てられたすべてのMRの一次レビュー者と
    -

なります あなたは、マージする権限があるかのようにMRをレビューしま
す

- MRの準備が整い、あなたがそれをマージしたと感じたら、それをコードオーナーに割り当て、あなたがMRをマージしただろうとコメントします。

    - コードオーナーは必要に応じてフォローアップレビューを行い、**MR**をマージするか、あなたのためにメンテナーに割り当てます。
    - 作成したオリジナルの**MR**でドキュメントを作成する
3. **5MR**の閾値に達した時点で、コードオーナーは申請者のマネージャーと協力して
4. 拒否された場合、**MR**を終了し、**3**ヶ月以上の期間、上司と協力して再応募する
5. 承認された場合、その**MR**をメンテナに割り当ててマージします。マージされると、コードオーナーの承認リストに自動的に追加されます。

メンテナ

データチームのプロジェクトにおけるメンテナーは、どの職種とも同義ではありません。ここでは、データチームはメンテナの責任についてエンジニアリング部門が示した前例を参考にしています。データチームのすべてのプロジェクトには少なくとも**1**人のメンテナがいますが、ほとんどのプロジェクトには複数のメンテナがいて、アナリティクスのように**dbt**とオーケストレーションに別々のメンテナがいるプロジェクトもあります。

メンテナの責任は次のことを保証することです。

- データチームのプロセスに従うこ
- と **MR**での最終確認

データチームのメンテナーになるには

メンテナになるためのガイドラインはありますが、具体的なルールはありません。メンテナは、**GitLab Data** プロジェクトのコードベースについて高度な知識を持っている必要があります。あるプロジェクトのメンテナになる前には、そのコードベースをよく理解し、一つ以上のドメインの専門知識を身につけ、そして私たちのコーディング標準を深く理解しなければなりません。これらの条件が両方とも満たされたとき、あなたはメンテナになる準備ができたと言えるでしょう。

1. あなたがレビューした**MR**は、追加で必要な大幅な変更をすることなく、一貫してメンテナのレビューを通過しています。
2. 作成した**MR**が、大幅な変更を要求されることなく、レビュアーやメンテナのレビューを一貫して通過する。

それらの主観的な要件が満たされていれば、自分をメンテナとして追加するプロセスとなります。

1. 関連するプロジェクトに **"Add as project maintainer "**というタイトルで課題を作成します。
2. 以下のように、課題にドキュメントを追加します。
    - なぜあなたがメンテナーを引き受けることができるのかを説明してください。
    - メンテナリングの範囲を説明する（プロジェクト全体、**dbt**、**python**など
    - あなたが作成し、レビューした最近の**MR**で、あなたの準備ができていると思われるもの。
3. 課題が作成されたら、正式なレビューのためにペアになりたいメンテナをタグ付けしてください。
4. メンテナは、少なくとも**10**個のマージ・リクエストであなたとペアを組ませます。
    - **MR**は、アーキテクチャの変更や、多くのファイルやディレクトリに変更を加えた完全に機能する機能リリースなど、多様な範囲を示している必要があります。
    - あなたは、これらの**10**個の**MR**の主なレビュアーとなります。
    - 融合する力を持っているかのように、**MR**の見直しを行う。
    - **MR**の準備が整い、あなたならそれをマージしただろうと思ったら、それをメンテナに割り当て、あなたなら**MR**をマージしただろうとコメントします。
    - メンテナは必要に応じてフォローアップレビューを行い、あなたのために**MR**をマ
    - ージします あなたが作成した課題の**MR**をドキュメント化します
5. **10MR**の閾値に達した時点で、メンターは申請者のマネージャーと協力して決定します
6. 拒否された場合、問題を解決し、**3**ヶ月以上の期間、上司と協力してから再度申請してください。
7. 承認されたら、**MR**を作成してチームページのエントリーにメンテナスを追加します。
8. **MR**を上司に割り当て、関連するプロジェクト（インフラ、アナリティクスなど）やエリア（**dbt**、エアフローなど）の既存のメンテナに言及します。
9. 関連するグループ **(**例**: dbt)** の既存のメンテナが重大な異議を唱えず、少なくとも半数のメンテナがレビュアーの準備ができていることに同意すれば、私たちは自分たち自身で新しいメンテナを獲得したことになります。
10. プロジェクトのオーナーが、プロジェクトでのあなたの特権を増やします

マージリクエストのワークフロー

データチームは、データアナリティクスチームとデータエンジニアリングチームの間にある1つのチームとして運営されています。そのため、各MRには最低でも3人の関係者がいることが予想されます。以下のシナリオをご覧ください。

1. DA著者、DAレビュー、DEメンテナのマージ
2. DEの著者、DAのレビュー、メンテナのマージ
3. DEの著者, DEのレビュー, メンテナのマージ

データエンジニアリングチームはデータプラットフォームに責任を持っているので、すべてのMR依頼にはデータエンジニアが含まれていなければなりません。

---

layout: handbook-page-toc title:"新しいデータソース" 説明"新しいデータソースを追加する方法"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

このページでは、データウェアハウスに新しいデータソースを追加するプロセスについて詳細に説明します。データチームは、データウェアハウスに新しいデータソースを追加するあらゆるイニシアチブを採用しています。私たちは、データが価値につながることを想定していますが、データウェアハウスに新しいデータを追加するには、コストがかからないわけではありません。

開発（データチーム、サポートに関わる他のチーム、要求者であるお客様のリソースの割り当て）と、データパイプラインの稼働維持（ストレージ、コンピュータリソース、インシデント管理、監視など）には、時間とお金がかかります。

新しいデータソースの追加をリクエストする前に、以下の点を考慮してください。

- 妥当なビジネスケースがあるのか？
  - ビジネスケースとしては、規制上の要件を満たすことが必要な場合もあります。
  - ビジネスケースは、投資コストよりも潜在的な価値がはっきりしているため、単純明快な場合もあります。ほとん
  - どの場合、価値が不明瞭であったり、コストが不明瞭であったりするため、定量化するのは困難です。そのような場合は、データチームと率直に話し合ってみてください。私たちには経験がありますし、正当化の手助けもできます。また、これは必ずしも科学的な計算である必要はありません。
- データウェアハウスにデータが入った時点では、作業は完了していません。データは生データ層に「ただ」入っているだけで、Sisenseのデフォルトではアクセスできません。データはdbtを介してEDM（Enterprise Dimensional Model）にロードする必要があります。フォローアップが必要で、このページで説明されているプロセスの上に行われます。
  - 当社はデータプラットフォームへの貢献を歓迎しているため、下流のモデリングはビジネスチームが担当することも可能です。dbt、SQL、データモデリングに関する幅広い知識が必要です。
  - 下流のモデリングはデータチームが担当することもあります。計画を立て、会社の優先順位に沿って優先順位を設定する必要があります。これは、データウェアハウスに新しいデータソースを追加する範囲ではないため、後から手配する必要がある。
- データウェアハウスに新しいデータソースを追加することは、1回きりの作業ではありません。データがデータウェアハウスに抽出されると同時に、定期的に（週に1回、1日に1回、1日に複数回など）データが更新されます。これは、実装後に何かが起こったり、うまくいかないことがあることを意味します。このプロセスを必要に応じてサポートするために、ソース側（ビジネスと技術）からDRIが必要になります。
- データはEDMの外でも使うことができます。例えば、ファンクションアナリストが生データ層で使用することができます。生データにアクセスするには、ARを上げる必要があります。
- データがEDMに入ってしまうと、Sisenseではダッシュボードを作成して作業を行うことになります。そのためには、いくつかの技術的な知識が必要です。
- データプラットフォームは、ソースシステムにアクセスする必要があります。当たり前のことのように聞こえますが、これは必ずしも簡単なことではありません。

新しいデータソース

新しいデータソースを追加する場合のワークフローは次のようになります（注：これは私たちがすべての作業に使用している既存のワークフローであり、ここでは新しいデータソースを追加するという文脈で明示しています）。

新しいデータソースを追加するためのプロセス。

新しいデータソースのステージ （ラベル）	サマリー
ワークフロー::1 - トリアージ	すべての初期情報は要求者によって提供され、データチームによって評価されます。
ワークフロー::2 - バリデーション	今後の開発のためのソリューションデザインと完全な作業内訳が作成されます。 四半期に一度、データチームのOKRが定義されます。作業量や優先順位に応じて、次の四半期のOKRに新しいデータソースを組み込むことができる。
ワークフロー::3 - スケジューリング	新しいデータソースの追加は、次の四半期のOKRのリストに入っています。すべての作業内訳活動が添付されたエピックが作成されます。
ワークフロー::4 - スケジュール	開発が進んでいる 開発は検討中
ワークフロー::5 - 開発	開発が阻害される
ワークフロー::6 - レビュー	
ワークフロー::X - ブロッキング	

ワークフロー::1 - トリアージ

新しいデータソースのリクエストには、必ず問題が発生します。これは、すでに抽出されているが、拡張が必要なデータソースや、まったく新しいデータソースにも当てはまります。新規データソースのテンプレートをご利用ください。すべての初期情報はリクエスト者によって提供され、データチームによって評価されます。データがデータウェアハウスにまだ存在していない場合は、クロスチェックが行われます。

ワークフロー::2 - バリデーション

新しいデータソースに関するすべての詳細は、完全な作業内訳を作成することを目的として、表示されます。データの範囲

要件に基づいて、抽出する必要のあるデータポイント（どのテーブルかなど）が決定される。総データ範囲が与えられなければならない。

抽出液

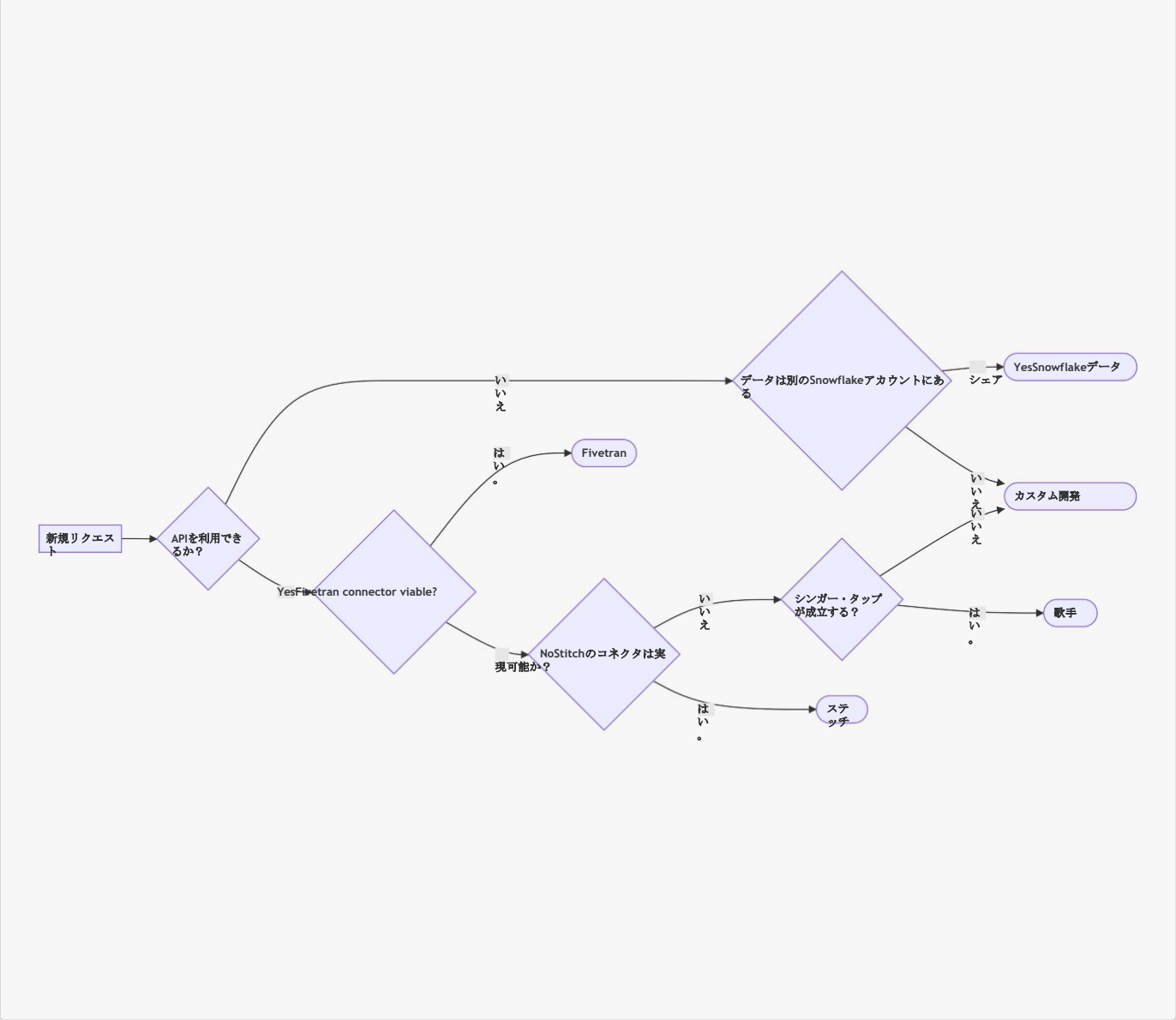
データチームには、ソースシステムからデータを抽出するためのさまざまな機器が用意されています（ランダムに発注されます）。カスタムメイド（Python経由

- フィベラ
- ン・メルターノ
- スノーフレークデータの共有 ステッチ

どの楽器を使うかの判断は、常に以下のような組み合わせで行われます。

1. 実装のための努力。
2. メンテナンスに力を入れています。
3. 伸び縮みする能力

どの機器を使用するかは、データチームが決定します。適切な抽出ソリューションを決定するために、次のような決定図が使用されます。



カスタム開発とは、**GitLab**データチームが設計・開発したソリューションのことです。その例として、現在の**PGP**や**Zuora Rev Pro**の抽出が挙げられます。

アクセスリクエスト

データチームにソースシステムへのアクセス権を提供しておくと便利ですが、今すぐアクセスリクエストを出す必要はありません。

作業内訳

検証の最終目標は、ソリューションデザインと完全な作業内訳、**DRI**とサイズの見積もりを添付することである。実行しなければならないすべての作業が記述されています。私たちは、すべての作業が課題に変換され、**T**字型のサイズが`M`/`5/8`以上にならないようなレベルを目指しています。

MNPIデータ

データ・ソースに**MNPI**データがあり、このデータがデータ・ウェアハウスに向けて抽出されようとしているかどうかを判断する必要があります。データ・ソースに**MNPI**が含まれており、このデータが抽出される場合は、課題のラベルを`新しいデータ・ソース` `MNPI`に変更します。

ワークフロー::3 - スケジューリング

ビジネスケース、実装にかかる労力、データチームの作業量、データチームの優先順位に基づいて、実装のスケジュールが決定されます。スケジューリングについては、**GitLab Data Team Planning Drumbeat**に従います。これは、四半期ごとに

データチームは、新しいデータソースのリクエストがいつピックアップされるかを決定します。新しいデータソースのリクエストが残っている場合
スケジューリングは、データチームが注目していないということではありません。それは、まだスケジュールが組まれていないということなのです。

1. 次の四半期のOKRの定義は行われませんでした。四半期に一度、データチームは次の四半期のOKRを設定します。
2. ビジネスケース、導入の手間、データチームの作業量、データチームの優先順位などの理由から、次の四半期のOKRには適合しませんでした。

ワークフロー::4 - スケジュール

新しいデータソースを実装する要求が、次の四半期のOKRの範囲内にある場合、その要求はスケジュールされます。課題が添付されたエピックが作成されます。作業内訳に基づいて、すべての課題が作成され、対応する課題のウェイト、ワークフロー「4 - スケジュール」のラベル、適切なマイルストーン(わかっている場合)が割り当てられ、エピックに添付されます。

ワークフロー::5 - 開発

作業の実行が始まると、その課題は開発中であり、通常の開発ライフサイクルに従います。

開発期間中、データエンジニアはデータへのアクセスについて、すべての関係者と調整を行います。データへのアクセスは、データやユースケースに応じて、生のスキーマに提供することができます。

ワークフロー::6 - レビュー

作業の実行が終了すると、課題はレビューに入り、通常の開発ライフサイクルに沿ったものになります。

ワークフロー::X - ブロッキング

外部からの介入を必要とするために実行が継続できなかった場合、問題はブロック化されます。明確な問題提起がなされ、適切な人材が最短でアサインされなければならない。

---

layout: handbook-page-toc title:"Data Team - Planning Drumbeat" 説明文"GitLab Data Team OKR and Milestone planning process"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

データチーム・プランニング・ドラムビート

データチームのPlanning Drumbeatは、四半期ごとに行われる事前設定のシーケンスです。プランニングドラムビートは、GitLabの会計年度と四半期の構造に従っています。データチームのプランニングは以下のようになっています。


プランニング・ドラムビートによるオリジナルへのリンク

マイルストーンの命名規則：マイルストーンFYxx-Qxx-MSxx。例：FY22-Q02-MS01
LT＝リーダーシップチーム（@rparker2, @iweeks, @dvanrooijen2, @mlaanen）。
DT = データチーム（プロダクトフュージョンチーム、GTMフュージョンチーム、データ

エンジニアリングチーム） 全体的な野心は、なぜ、いつ、何をするのかを構造化する

ことです。

- OKRは「なぜ」につながる

- ワークブレイクダウンで何をするか
- マイルストーン計画でいつ何をするか

このようなトップダウンのアプローチは、私たちの仕事のやり方にも通じるものがあります。

OKRプランニング

データチームのOKRは、ビジネス・テクノロジーのOKR、財務部門のOKR、CEOのOKRと連携し、サポートする部門のOKRと一致することを目指している。エンタープライズ・データ・ウェアハウスの開発と運用に必要な技術とデータ・インフラストラクチャーの作業の性質上、これは必ずしもそうではありません。

FQの終わりまでに、データチームは自分たちのKRを成功させるため、また他のチームがKRの成功を測定するのを支援するために必要なすべての行動の概要を示す。これはイシューを使って行います。イシューでは、関連するアクションのそれぞれについて、全員が予想されるすべてのステップを説明します。これは、チームが予測している障害や懸念を提起する絶好の機会です。これらは将来の参考のために記録しておきます。

これらのOKRは、セントラルデータチームが四半期に行う作業の60%を占めています。残りの時間は、緊急に発生した問題と、アドホック／探索的な分析に分けられます。

OKRアライメントの*実例*は以下の通りです。

1. FY21-Q2CEO Objective 2、データチームのObjective 2と連携し、成長戦略をサポートするための自動化されたデータパイプラインを提供する。
2. FY21-Q2CEOの目標3、KR4、KR5は、データチームの目標1に合わせて、財務ARRデータマートを提供する。

OKRレビュー

OKRレビューでは、データリーダーシップチームが、現在のFQ KRの現状をレビューします。OKRの結果は評価され、必要に応じて次のFQ OKRに伝達されます。

作業内訳

作業内訳の成果は、次の四半期に実施する作業、成果物、責任の詳細な説明であり、以下のように決定されます。

1. 今後のOKRの定義
2. OKRレビュー
3. 新しい／将来を見据えたインサイト
4. チームの可用性
5. チームメンバーの抱負

仕事の内訳はチームワークであり、全員が貢献することが奨励されています。T

シャツのサイジングの考え方

当社では、新しい課題や大作、長期的な構想を実現するために必要な作業量を迅速に把握するため、T-シャツサイジングアプローチを採用しています。T-shirtサイジングアプローチは、詳細な作業計画を作成するための作業分解をサポートするように設計されていますが、全体のスコープを管理するために十分なレベルの詳細を提供することも目的としています。

| サイズ｜専任者の時間｜重さ（課題点）｜例｜ | :--:| :--:| :--: | XS | < 半日 | 1 | 既存のハンドブックページの更新#データの調査・対応。新しい信頼できるデータテスト。S｜1日｜2-3｜新しいハンドブックページ；典型的なトリアージ問題；既存モデルの上に新しいダッシュボードを追加。M｜1週間｜5-8｜新しいモデルを必要とする新しいダッシュボード。StitchやFivetranを使った新しいデータソース。L｜2〜3週間｜13｜新しいファクトテーブルの実装とテスト。完全なXMAUソリューション。XL｜1〜2ヶ月
| 新しいシステムへの新しいデータポンプ。複雑なソースAPIを持つ新しいデータソース。| XXL｜2-4ヶ月｜52+｜
新しいデータソースを使った新しい次元のモデル対象領域。

マイルストーン計画

データチームは、マイルストーンと呼ばれる2週間の間隔で作業を行います。1四半期に1つのマイルストーンは、1四半期が13週間であるため、3週間かかります。3週間かかるマイルストーンとは、大型連休を挟む場合や、チームの大半が休暇やContributeに参加している場合などです。マイルストーンは、水曜日に始まり、火曜日に終わります。これにより、金曜日に土壇場でマージすることを抑制し、チームはマイルストーンの先頭でマイルストーン計画会議を行うことができます。

マイルストーン計画では、以下の点を考慮する必要が

- あります：休暇のスケジュール

- 会議のスケジュール チー
- ムメンバーの空き状況
- チームメンバーの仕事の好み（スペシャリティは好みとは異なる マイルスト

ーン計画のタイムラインは以下の通りです。

- ミーティングの準備 - 責任者**Milestone Planner** 未解決の問題を調
  - 査し、具体化する。
  - チームロードマップとの整合性に基づき、マイルストーンに課題を割り当てます。
  - 注：この段階では、必要な場合を除き、課題が個人に割り当てられることはありません。

	日現在のマイルストーン	次のマイルストーン
0 - 第1水曜日	マイルストーンスタート ロールマイルストーン	-
7 - 第1火曜日	中間点 マイルストーンからずれてしまう恐れのある問題は、アサイニーが提起しなければならない	-
10 - 第2金曜日	**MR**の最終提出日について **MR**は、ドキュメントとテストを含めて、すぐにマージできるようにする必要があります。 金曜日（「家族と友人の日」の場合は木曜日）には、**MR**の合流を行わない。 マイルストーンの最終日	マイルストーンはほぼ最終段階 マイルストーンプランナーは、次のマイルストーンに向けて課題の優先度とチームのキャパシティを確認します。
13 - 第2月曜日	準備ができた**MR**はマージ	-
14 - 第2火曜日	されることができる 会議 の日 未完成の課題は、マイルストーンから削除するか、次のマイルストーンに回す必要があります。	マイルストーン計画 **Sync-meeting**では、現在のマイルストーンのレトロパースペクティブを実行し、作成されたマイルストーン計画に従って次のマイルストーンの調整/開始を行います。すべての未完成の課題は、マイルストーンから削除するか、次のマイルストーンにロールオーバーする必要があります。

プランニング・ドラムビートの目的

このプロセスの目的は、ベロシティの理解を深めることで、作業の計画と見積もりの能力を向上させることにあります。マイルストーン計画では、課題点を設定しますが、これはマイルストーン間で平均値を共有するため、一貫性を測るのに適しています。そして、完成した仕事にコミットできると考えられる時期に基づいて、マイルストーンに引き込みます。そして、優先順位に応じて課題の優先順位を決定します。

一度に担当する課題は**2**つ以下になるでしょう。このアプローチ

には、以下のような多くの利点があります。

1. 最優先のプロジェクトを確実に完了させることができる
2. リーダーシップが阻害されている問題を特定することができます。

3. データ機能の外部から優先順位を設定されている専門アナリストを含むデータチームの作業にリーダーシップの視点を提供します。
4. チームメンバーの安定したスループットを促進する
5. ステークホルダーに対して、自分の要求がどこに優先しているのかを明確にする。
6. チーム全員に、自分の任務がどこにあるのか、優先順位を明確にする。
7. 次のマイルストーンを計画する際に、課題がすでにランク付けされているため、プレッシャーを軽減することができます。 計画の鼓動と実行は、チームの努力の賜物です。

---

layout: handbook-page-toc title:"Data Triage Guide" 説明"GitLab データ・トリアージ・ガイド"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

データトリアージ

Triager

データチームでは、各国のタイムゾーンを考慮して、以下のようなトリアージスケジュールを実施しています。

UTCデー	データアナリスト	データエンジニア
日曜日	@ken_aguilar	-
月曜日	@chrissharp	VEDPRAKASH2021
火曜日	@chrissharp	@paul_armstrong
水曜日	@michellecooper	rbacovic
木曜日	@ken_aguilar	ラドゥーラ
金曜日	@jeanpeguero	@jjstark

休暇中や優先度の高いプロジェクトに従事しているチームメンバーは、カバー範囲を探し、カバー範囲を引き継ぐチームに連絡する責任があります。これはデータチームのGoogleカレンダーで更新する必要があります。毎週同じ人に*月曜日の仕事*を押し付けないように、データエンジニアは時々交代でトリアージの日を設け、その場限りのコラボレーションを行います。

チームに専任のトリアジャーがいることで、傍観者の影響にも対応できます。スケジュールは日々のオーナーシップ情報を明確に共有していますが、オンコールのポジションではありません。オーナーシップを明確にすることで、チームの他の全員が1日の大半を深い仕事に費やす余地を作ることができます。トライアングルは、このような追加の時間的要求があってもうまく達成できるような仕事の種類に合わせて一日の計画を立てるようにします。

Data Triagerは、Dataチームへのリクエストや問題に最初に対応する人です。

- **Data Analyst triagerは、GitLabチームメンバーからのリクエストに応えることを主な仕事としています。#data-triage**に投稿される**issue**を介して、または**#data**の**slack**で直接リクエストに応えます。
- データエンジニアのトリアージ担当者は、主に当社のデータプラットフォームに関する問題を解決する責任があります。これらの問題は、**#analytics-pipelines** スラックチャンネルか、**DE-TriageErrors** ボードに投稿されます。これらのエラーから作成された課題は、**DETriageErrors**の課題テンプレートを使用する必要があります。
  - 割り当てられたトリアージの日には、データエンジニアは主にアクティブな問題や、**DE-TriageErrors**ボードに表示された問題に集中しなければなりません。
- データアナリストとデータエンジニアのトライアングルは、それぞれプラットフォームの問題と**GitLab**チームメンバーのリクエストを担当します。

- トリアージの責任についての詳細は、データエンジニアのトリアージトレーニングセッションのビデオをご覧ください。

他の GitLab チームメンバーからデータチームのプロジェクトに入ってきた問題の多くは、理解、見積もり、優先順位付けのために追加の情報やコンテキストが必要です。このような質問をして、問題を早期に解決することがトリアジャーの優先事項です。

注：Data Analyst triagerの場合

- は、すべての質問に対する答えを知っているとは限らない。
- 対象分野の専門家（SME）や機能的なDRIである他のチームメンバーにcc（言及）してフィードバックを求めるべきで
- ある。 躓いた問題を文書化し、チームメンバー全員に知識を広めるのを助けるべきである。

データトリアージの方法

DataTeamプロジェクトで課題を作成します。タスクと任務は課題テンプレートに記載され

ています。よくある質問と一般的な問題をお読みください。

トリアージ・ボット

トリアージの一部は、Analyticsプロジェクトに設置されたGitLab Triage Botによって支援されます。このボットは1時間ごとに実行され、ポリシーファイルで定義されたルールに基づいてアクションを起こします。GitLabTriageのREADMEには、ルールのフォーマットや定義に関するすべてのドキュメントが含まれています。

triage botのポリシーファイルの変更は、CIジョブ "dry-run:triage "を実行して、ログ出力を確認することで、MRでテストする必要があります。このCIジョブはドライランです。つまり、プロジェクト内で実際には何もアクションを起こさず、ポリシーが実際に実行された場合に何が起こるかを出力します。

エンドオブデイ・ラップアップ

日々のトリアージをより良く、より効率的に行うために、その日のうちに作業をまとめます。以下の情報は、データアナリストとデータエンジニアが毎日提供しています。

- タイムトラッキング。その日のトリアージにかかった時間と、どのような活動が行われたかを記録すること。一般的な考え方としては、どこに作業負荷がかかっているかを理解し、それらの活動を最適化することです。分析目的に役立つように作業内容を記載してください。例えば
  - データインジェストの問題解決に2時間、データ
  - 変換の問題解決にソース×1時間
  - 壊れたSisenseダッシュボードの解決に1時間
  - トリアージと（新しい）問題の再ルーティング
    に1時間
  - Slackの質問への回答に0.5時間
- グラウンドホッグの問題定期的に発生する問題は、私たちを悩ませ、無駄な時間を費やしています。何度も出てくる問題をリストアップしてみてください。これは、現在の風景の中で対処すべきスポットを見つけるためです。

トリアージラウンドアップは、各マイルストーンの終わりに、データリーダーシップチームによって行われ、マイルストーンのトリアージの取り組みを集約します。情報提供の目的は、それを有用なものとし、Triageを向上させることであることを念頭に置いてください。

GitLab.comのDB構造の変更

定期的に変更される最も重要なデータソースのひとつが、GitLab.comのデータベースです。日々の業務を中断させないためには、データベースへの変更を追跡・確認する必要があります。GitLab.comデータベースへの変更はすべて、db/structure.sqlファイルに行われます。データチームは、db/structure.sqlに変更が加えられた場合、MRにラベルを適用することで、Danger Botを介して通知を受けます。

- `Data Warehouse::Impact Check`というラベルがDanger Botによってデータチームの行動を促すために追加されます。

  トリアージでは、Triagerは`Data Warehouse::Impact Check`というラベルでMRをチェックします。

以下のアクションは、データチーム・トライジャーによって実

行されます。すべての**MR**が判定される

-
-

- **SQL**ファイルの変更によって操作が中断されていない場合は、ラベルは`Data Warehouse::Not Impacted`に変更されます。
- **SQL**ファイルの変更により操作が中断された場合。ラベルは
  - `Data Warehouse::Impacted`に変更されます。
  - 新しい課題が`GitLab Data Team`プロジェクトに開設され、正しい**DRI**に割り当てられ、元の**MR**にリンクされます。
  - 影響は課題の中で決定されます。
  - 任意の**MR**は、ロードの問題、下流の**dbt**処理、**Sisense**の使用を克服するために作成されます。
  - **GitLab.com**の**MR**の**Merge**によると、**Merge**は計画的に行われます。
  - すべての関係者に通知されます。

判定マトリクス。**

	ChangeCall to Actionが必要*。
新しいテーブルの作成	❌
表削除	✅
テーブル名の変更	✅
フィールド追加	❌
削除されたフィールド	✅
フィールド名の変更	✅
フィールドのデータタイプの変更	❓
制約条件の変更	❓

*デフォルトでは、すべてのテーブルやカラムをロードしていません。そのため、新しいテーブルやカラムが追加されても、特定のビジネス上の要求があった場合にのみ、これらのテーブルを読み込むことになります。現在の構造に変更を加えることで、操作が中断される可能性があるかどうかを判断する必要があります。

判断材料は豊富ではありません。すべてのMR**を注意深くチェックする必要があります。**

GitLab Postgres データベースにアクセスできない

gitlabにクローンされた**Postgres**データベースにアクセスできない場合、**airflow**のタスクログに以下のエラーが表示されます。

```
sqlalchemy.exc.OperationalError:(psycopg2.OperationalError) FATAL: The database system is starting
up\n
b'FATAL: The database system is starting up\'D
```

以下の手順で行います。

1. **DE Triage**テンプレートを使用して課題を開きます。
2. `gitlab_com_data_reconciliation_extract_load`、`gitlab_com_db_extract`、`gitlab_com_db_incremental_backfill`、`gitlab_com_scd_db_sync` という名前の **gitlab.com** の **DAG** をすべて一時停止します。その理由は、警告を抑えて不要なリソースを使わないためです。
3. アラートチャンネルを見て、「**GitLab Job has failed**」を検索してアラートを見つけます。サンプルのアラートは以下のような内容です。

```
Firing 1 - GitLab Job has failed
GitLabジョブの "clone "リソース "zlonk.datalytics.dailyx "が失敗しました。
:chart:プロメテウスのグラフを見る :label:ラベ
 ルです。アラートネーム: JobFailed
 Alert_type: Symptom (症状)
```

```
環境：GPRD 環境：GPRD
Fqdn: blackbox-01-inf-gprd.c.gitlab-production.internal
Job: clone
モニタ：default
プロバイダ：gcp
地域：us-east
リソース: zlonk.datalytics.dailyx
Severity: s3
Shard:
default
Stage: main
Tier: db
タイプ:
zlonk.postgres 少ない
表示
```

4. 問題を調査するために `@sre-oncall` の **slack** ハンドルに連絡し、 **incidentdeclare** を使用してインシデントリクエストを上げてください。これにより、**SRE**オンコールチームが対応するための本番のインシデント問題が作成されます。また、インシデントをより広く認知してもらうために、`@gitlab-data/engineers`にも連絡してください。
5. インフラの課題と、提起されたトリアージの課題をリンクします。
6. 問題が解決するか、`@sre-oncall`の担当者または**DBRE**チームの誰かから確認されたら、接続の安定性を検証するために、失敗したタスクの**1**つを単独でクリアして再実行してみてください。
7. **DAG**の場合、`gitlab_com_scd_db_sync`、`gitlab_com_data_reconciliation_extract_load`、`gitlab_com_db_incremental_backfill`は、**24**時間以内に**1**回しか実行されないので、失敗したタスクをクリアして、実行するようにします。
8. **DBT**がソースの更新に間に合わない日に実行された場合は、ソースの更新が遅れていることをトリアージテンプレートを使って**#data**チャンネルに通知します。

Zuora Stitch Integration シングルまたはセットのテーブルレベルリセット

テーブルを完全に埋め戻すために、**Stitch**で**Zuora**データパイプラインのテーブルをリセットすることは、どのような場合にも起こり得ます（例：ソースに新しいカラムが追加された、テクニカルエラーが発生したなど）。現在、**ZuoraStitch**統合はテーブルレベルのリセットを提供してないため、統合内のすべてのテーブルのリセットを実行する必要がますこの余分なコストとリスクが発生ます

このため、以下の手順でテーブルレベルのリセットに成功しています。この例では、**Zuora**購読テーブルを使用していますが、**Stitch Zuora**データパイプラインの他のテーブルにも適用できます。

ステップ**1**： **-** 既存のテーブルの名前を変更し、バックアップを識別するために日付の接尾辞を付けます（推奨フォーマットは**YYYYMMDD**）。

```
alter table "raw". "zuora_stitch". "subscription "の名前を "raw". "zuora_stitch".
"subscription_20210903 "に変更します。
```

Step 2:-定期的な統合を一時停止する。

 通常の統合を一時停止する

ステップ**3**： **- Stitch**で新しい統合**Zuora-Subscription**を作成します。

この設定では、抽出頻度を**30**分に、抽出日を**2012**年**1**月**1**日に設定し、すべてのデータが抽出されるようにしました。

 複製するのはサブスクリプションテーブルの

み ステップ**4:-**新しく作成した統合を実行しま

す。

新しく作成した統合機能を手動で実行し、完了するのを待ってください。完了すると、ホームページに正常に表示されます。完了したら、次のステップに進む間、データがずれないようにするため、新しく作成した統合タスクを一時停止します。

Step 5:-記録を確認する

新規に作成したテーブル "RAW". "ZUORASUBSCRIPTION". "SUBSCRIPTION "で、統合UIでstitchに読み込まれたと表示される行数とテーブルに読み込まれた行数が同じかどうかをクロスチェックします。

Step 6:-メインスキーマにテーブルを作成します。

新たに読み込んだデータをZUORA_STITCHスキーマに移動します。なぜなら、新しい統合ではテーブルが
画像の上に記載されているZUORASUBSCRIPTIONです。

```
create table "raw". "zuora_stitch". "subscription" clone "raw". "zuorasubscription". "subscription";
```

注：テーブルのポストクローンに主キーが存在するかどうかをチェックし、存在しない場合はリンクに主キーをチェックし、それらのカラムに制約を追加します。

ステップ**7:-**レコード数のチェックを行い、新しいテーブルのレコード数が少なくなっていないかを確認します。

```
select count(*) from "RAW". "ZUORA_STITCH". "SUBSCRIPTION_20210903" where deleted =
'FALSE'; select count(*) from "RAW". "ZUORA_STITCH". "SUBSCRIPTION" ;
```

ステップ**8:-**新しいスキーマをドロップする

```
drop schema "raw". "zuorasubscription" cascade ;
```

ステップ9:-一時的なZuora-Subscription統合を削除し、通常の統合を有効にする ステ

ップ10:-通常の統合を実行し、検証する

これは、テーブルに以前発生したエラーがなくなり、データがテーブルに入力されていることを確認するためです。2つの異なる抽出器による重複したIDをチェックし、データが正しくテーブルに入力されていることを確認します。

```
select id, count(*) from "RAW". "ZUORA_STITCH".
"SUBSCRIPTION" group by id
カウント(*)が1の場合
```

メモ 詳細はMRを参照してください。

トリアージFAQ

Data Triageは24時間365日のサポート、もしくは24時間サポートしなければならないシフトですか？
通常の勤務時間内に、トリアージテンプレートに記載されているトリアージ当日のタスクリストを実行する必要があります。

何か問題が見つかった場合、直接本番で修正するのか、それともインシデントの一部として捉え、定義された時間内に解決するのか。
トリアージの日には、その場にいたデータチームのメンバーが、すべての失敗、質問、エラーを探します。

- Slackチャンネル; #data-prom-alerts #analytics-pipelines and #data 新たに追
- 加された課題
- Sisense の TDF ダッシュボード

これには、最後の人がサインオフしてからのすべての失敗が含まれ、それ以降、その人がサインオフするまでのすべての失敗について問題が発生します。データのパイプラインが破損し、データの読み込みや更新に遅延が予想される場合。その場合

関係するチームに[トリアージテンプレート]を使って通知しなければなりません(https://gitlab.com/gitlab-data/analytics/-/issues/new?

異なる種類の問題については、ETAがありますか？
パイプラインが壊れている場合は、修正する必要があります。現在、私たちはデータ資産のSLOの定義に取り組んでいます。データ抽出パイプラインについては、こちらに包括的な概要が掲載されています。

トリアージの日に通常の勤務時間、つまり米国時間の午前11時まで働いたとします。通常の勤務時間後にパイプラインが切断され、データの入手が遅れた場合はどうなりますか？
そうですね、私たちの存在のメリットは、時間のオーバー幅が広いことです。トリアージに参加している人が米国のタイムラインに先行していれば、問題をタイムリーに解決できるというメリットがあります。デメリットは、米国のタイムラインをその日のうちにフルカバーできないことです。これは将来に向けての注意点です。

一般的な問題のトリアージ

このセクションでは、一般的な問題とその解決策について説明します。

エアフロータスクの失敗！？

DAG `gitlab_com_db_extract`
タスク `gitlab-com-dbt-incremental-source-freshness`

背景を説明します。このエキスは、GitLab.com環境のコピー（レプリケーション）データベースに依存しています。これがレプリケーションの遅延の原因となっている可能性が高いです。

セットアップの詳細はこちら

考えられる手順、解決策、対処法- レプリケーション・ラグの確認
- 必要に応じてDAGを一時停止する
- データギャップの確認
- 埋め戻し作業
- DAGのリスケジュール

注：GitLab.comのデータソースは非常に重要なデータソースであり、一般的に使用されています。ビジネス関係者に適宜お知らせください。

便利な正規表現

これらの用語が存在しない行にマッチ

```
^(?!.*(<一番目に探す用語>|<二番目に探す用語>)).*$
```

例：Airflowのログをきれいにするため。

```
^(?!.*(テストでの失敗|データベースのエラー)).*$
```

---

layout: handbook-page-toc title:"Data Team Learning and Resources" 説明"GitLab データチームライブラリ"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

Powered by GitLab Team Members

- How Data Teams Do More With Less By Adopting Software Engineering Best Practices - Thomas's talk at 2018 DataEngConf in NYC
- ナッシュビルで開催された「2019 Music City Tech Conference」でテイラーが行った同様の講演のスライド

- LocallyOptimisticAMA:August2019withTaylorMurphy オー
- プンソースアナリティクスの力を示す4つの例 GitLabCIを
- 使った初めてのdbtプロジェクトのデプロイメント
- クラウドネイティブな世界でのデータオプス
- データ分析の3つのレベル-データ組織の成熟度を評価するためのフレームワーク GitLabを
- 使ったDataOpsの導入方法
- GitLab Dataチームの運営で得た教訓
- Views on Vue Podcast with Jacob Schatz and Taylor Murphy
- HowtodoDataOpswithGitLab-CustomerCall (GitLab Internal)
- GitLabforML-CustomerCall (GitLab Internal)

おすすめの読み物、聴き物、見物

- AIのニーズの階層 データ
- ・メタ・メトリクス
- エンジニアはETLを書くべきではない
- スタートアップ創業者のためのアナリティクスガイド
- Functional Data Engineering - バッチデータ処理のための最新パラダイム
- KeepitSQLStupid、DataEngConfSF'18でFishtownAnalyticsのConnorMcArthurがAIのためのdbt DevOpsを
- 説明した講演です。
- データサイエンティストはDevOpsから何を学ぶか
- 一人のアナリストが語る「GoodtoGreat」への道
- データの価値とは？第1回、第2回、第3回
- データプラクティスの構築
- スタートアップのデータチームにデータエンジニアは必要か？
- データサイエンスは今までとは違う(注：GitLabにまだデータサイエンティストがいないの
- はこのためです) WhyDon'NeedDataScientists
- リソースWrittenbydbtCommunityMembers あな
- たの会社はデータドリブンには向いていません
- か？あなたにとって「セルフサービス」アナリ
  ティクスとは
- 組織内のデータサイエンスチームを統合するためのモデル
- 成熟したアナリティクス・ワークフローの構築(注：「アナリティクスはソフトウェア工学のサブフィールド」という
  前提を説明しています。)
- YouTubeのDataOpsプレイリスト

データニュースレター

- アルゴス＆エ
- シックス カロ
  ージカ
- The Carpentries
- DataEng Weekly
- DataElixir
- Data is Plural
- データサイエンス・ラウンドアップ・
- ニュースレター データサイエンス・
- ウィークリー
- LantrnsAnalytics（プロダクトアナリティ
- クス）ミュージック＆テック
- Normcore Tech
  NumLock News
  OneShotLearning
  SFData
-

データブログ

-

Airbnb
AskGoodQuestions
BufferBlog

- カロージカ
- FishtownAnalyticsBlog
- GoDataDriven
- MBAMondays
- ModeAnalyticsBlog
- Multithreaded
- Sisense Data Blog
- Silota
- WesMcKinneyBlog デー
- タ運用
- レティナAIブログ
- StitchFixAlgorithmsBlog
- Five Thirty Eight
- Data.gov

データビジュアライゼーションリソース

- データを使ったストーリ
- ーテリング
  DataRevelations
- データ・ビジュアライゼーシ
- ョン・カタログ EagerEyes
- FiveThirtyEight'sDataLab
- FlowingData
- データからViz
- Gravy Anecdote
- へ JunkCharts
- MakeaPowerfulPoint
- Makeover Monday
- Perceptual Edge
- PolicyViz
- TheFunctionalArt
- The Pudding
- Visualising Data
- VizWiz
- WTF ビジュアライゼーション

データ スラック コミュニティ

- データビズ協会
- データサイエンス・コミュ
- ニティ dbt
- 大いなる期待 Locally
- Optimistic Measure
- メルタノ
- オープンデータコミュニティ
- Pachyderm
- プリフェクトパ
- イカロリナス
- R for Data Analysis
- SoftwareEngineeringDaily
- TheDataSchool

テクニカル・ラーニング・リソース

- クリス・アルボン
- Mode SQL チュートリアル

- **dbt Tutorial**
- **Technically.devPostonSQL**
- **ElementsofDataScience**
- **MachineLearningResources (GitLab Internal)**
- **Codecademy**
- **DataQuest**
- **KhanAcademy**
- **HackerRank(Exercises)**
- **Udacity**
- **StanfordUniversityMini-Courses**
- **The Data School by Chartio**
- **W3Schools**

---

layout: handbook-page-toc title:"Data Team Organization" description:"GitLab データチームの組織"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

データチームの編成

データチームは、卓越した技術と主題を持つセンターで構成されており、フュージョンチームと呼ばれるデータソリューションを提供するビジネスに特化したチームとビジネスに関与するチームによって補完されています。データプラットフォーム&エンジニアリングチームは、データソース、パイプライン、分析ツールに加えて、高性能なコンピュートとストレージのレイヤーを提供することで、データフュージョンチームを補完しています。そして、データチームを補完し、高度な分析機能を提供するのが、データサイエンスチームです。チームの構成や、GitLabの他のチームとの連携については、How Data Works at GitLabをご覧ください。

データ&アナリティクス・デモ

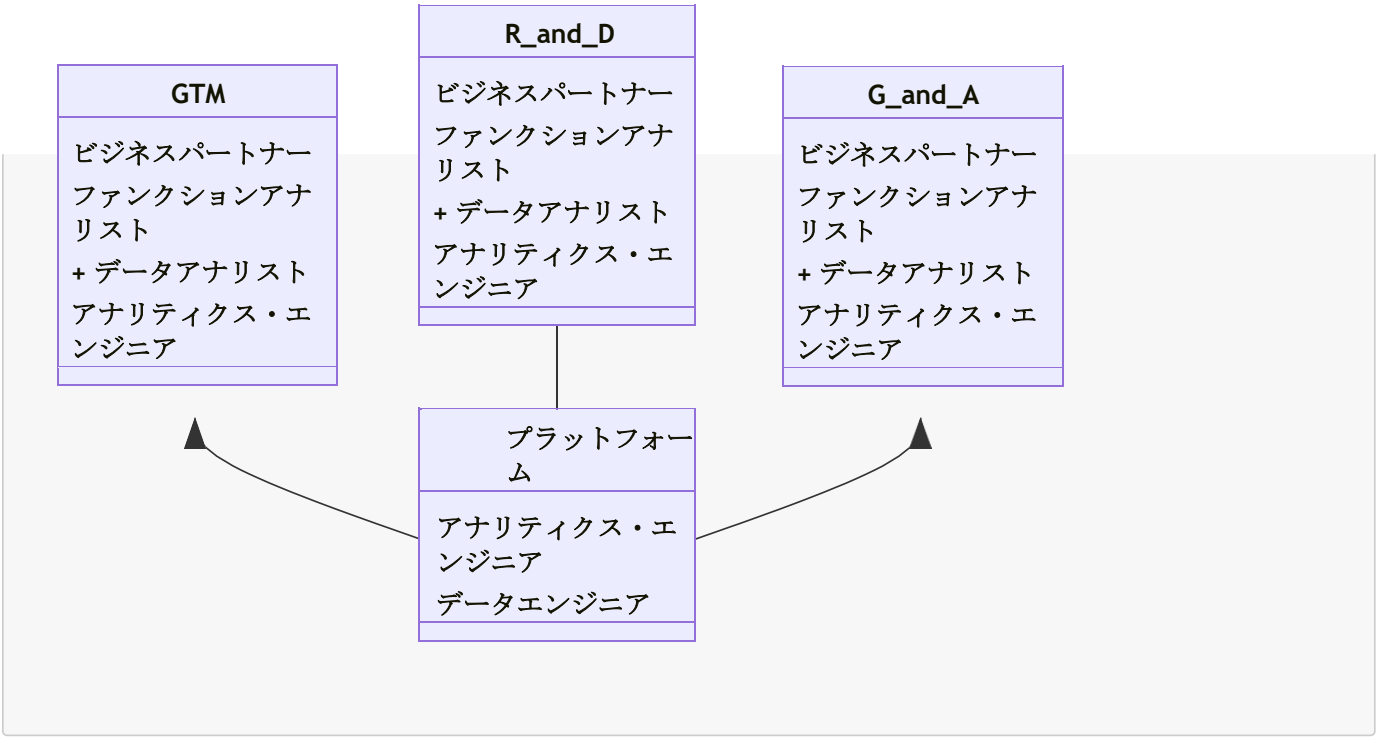
データ&アナリティクス・デモは、データプログラムに関わるすべての人が、進歩や革新を共有し、協力し合い、ただ楽しむための素晴らしい方法です。データ&アナリティクス・デモ毎週木曜日され、録画GitLabUnfilderedDataTeamのプレイリストに投稿れ。

データプログラム採用動画

この度、GitLabDataRecruitingのビデオを作成しました。ぜひご覧ください。

データフュージョンチームの組織

データフュージョンチームは、ビジネスとデータチームの間のチームメンバーで構成されています。現在のデータフュージョンチームについては、フロントページでご紹介しています。



フュージョン・チーム・アサイメント

		GTMR&DG&A
iweeks ***Lead/DRI***	@mlaanen ***Lead/DRI***	@iweeks ***DRI***
	@ken_aguilar@snalamaru	@pempey *リード*
	Jeanpeguero@chrissharp	@jjstark データプラットフォームチーム 安定したカウンターパート
@michellecooper	@jeanpeguero **GTM Fusion Team** 安定したカウンターパート	
@paul_armstrong データプラットフォームチーム 安定したカウンターパート	@rbacovic データプラットフォームチーム 安定したカウンターパート	

データプラットフォームチーム 安定したカウンターパート

GitLab Stable Counterpartの原則に従い、すべてのFusion TeamにはData Platform Team Stable Counterpartが割り当てられています。データプラットフォームチームのステイブルカウンターパートは、データプラットフォームチームとフュージョンチームの間で、自分の時間、仕事、優先順位を分けています（一般的に平均50%ずつ）。安定したカウンターパートは、フュージョンチームの方向性や優先順位を把握しており、必要に応じてデータプラットフォームチームと議論します。例えば、安定したカウンターパートが割り当てられた時間内に処理できる以上の需要がある場合や、アーキテクチャの方向性を変更する必要がある場合などです。安定したカウンターパートは、これを認識し、フラグを立て、該当するステークホルダー（一般的には、データプラットフォームチームとFusionチームのリード/DRI）に対処します。

フュージョン・チーム・オペレーション

データフュージョンチームを成功させるためには、以下の要素が不可欠です。

1. DataforGTMSeries」や「DataforR&DSeries」を通じて、ビジネスパートナーやデータチャンピオンとの定期的かつ透明性の高いエンゲージメントを行います。
2. データチームによる作業計画の立案 プランニング・ドラムビート
3. 定期的なCSAT調査を実施し、継続的な改善を目指してデータフュージョンチームにフィードバックを提供する

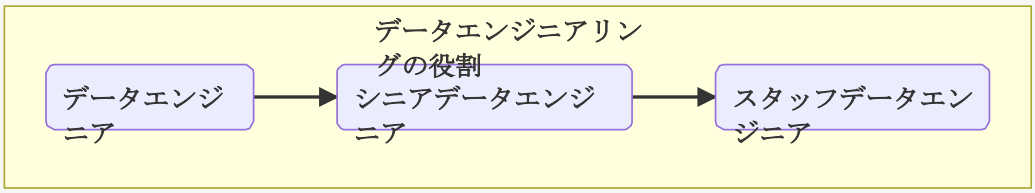
ステークホルダーの皆様には、私たちのイシューボードに沿って作業範囲を理解していただくことをお勧めします。

1. GTM

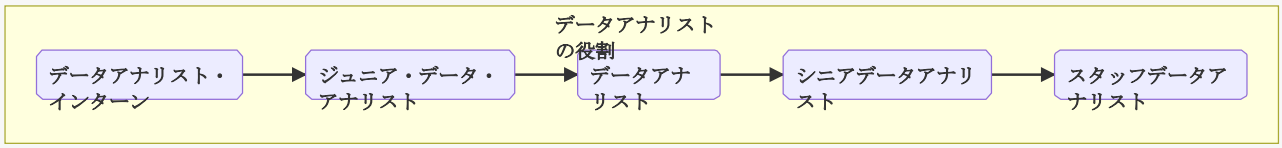
2. 研究開発
3. **G&A:** 人、エンジニアリング・アナリティクスは近日公開
4. プラットフォーム

データの役割とキャリア開発

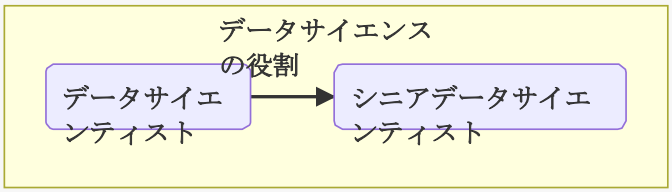
データエンジニアリング



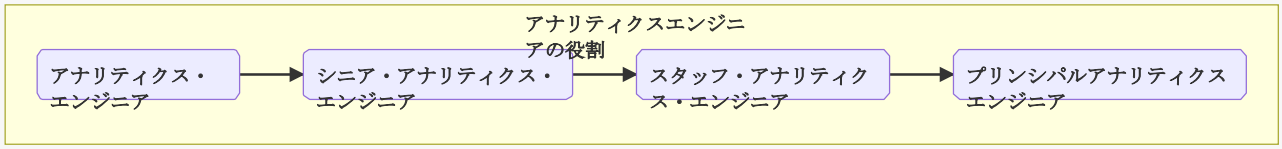
データ分析



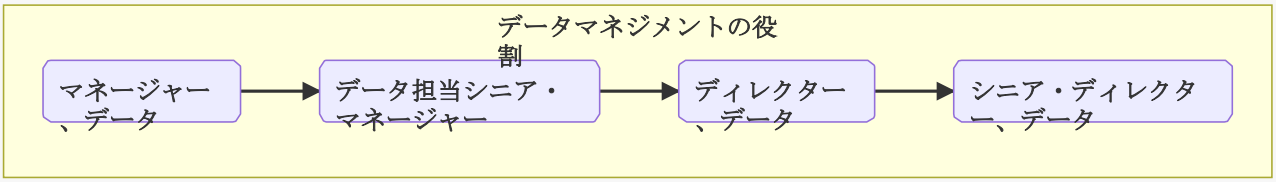
データサイエンス



アナリティクス・エンジニアリング



データ管理



layout: handbook-page-toc title:"Data Analytics Handbook" description:"GitLabデータ分析チームハンドブック"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

GitLabでのデータ分析

データ・アナリティクス・チームのミッションは、信頼性と拡張性のあるデータ・ソリューションによって、ビジネス上の意思決定と戦略のインパクトを最大化することです。

私たちは、**GitLab**バリューとデータチーム・プリンシプルによって、すべての**GitLab**チームがデータ・バリュー・ピラミッドの上に到達するのを支援します。

データ分析の責任

Data Team's Responsibilitiesのうち、**Data Analytics Team**が直接担当するのは以下の通りです。

- 会社の主要業績評価指標の定義、データベース、およびデータの可視化の管理と運営 データの可視化、データモデリング、データ品質、およびデータ統合を含むカスタマイズ可能なデータサービスの提供 データ＆アナリティクスシステムに関連する会社のガバナンス、リスク、およびコンプライアンスプログラムのサポート

さらに、データ・アナリティクス・チームは以下の責務をサポートします。

- データリーダーシップチームと
  - **GitLab**のデータ資産の価値を最大化するためのデータ戦略の策定と公開 データの成果物、進行中のイニシアチブ、今後の計画に関する定期的なアップデートの配信
- データエンジニアリングチームと
  - すべての**GitLab**チームのレポーティング、分析、ディメンショナルモデリング、データ開発をサポートするための、会社の中心となるエンタープライズデータウェアハウスの構築と維持
  - マスターデータ、リファレンスデータ、データ品質、データカタログ、データパブリッシングなどのデータマネジメント機能の開発
  - 誰もがデータと分析を活用できるよう、セルフサービスのデータ機能を提供する
  - **GitLab**データシステムのデータ品質に関するプラクティスとプログラムの定義と推進の支援

データ・アナリティクス・チームの最初のハンドブック

GitLabでは、**Handbook First**を掲げ、これを推進しています。

1. ビジネスパートナー（すべての**GitLab**チームメンバー）がハンドブックを最新かつ正確な情報で更新するのを支援する。
2. 内容を見直し、データの機密性やセキュリティを考慮した上で、ハンドブックの情報がデータを最もよく表していることを確認する。
3. データチームのハンドブックに、データアナ
   - リティクスに関するエキサイティング
     な内容が追加されました。
     - 今後のコンテンツの例データビジュアライゼーションのヒントとコツ」「データストーリーテリング
   - データチームのプロセスやコードベースの変更を反映させる（より良いプロセスやコードに興奮するのは当然のことです。

アナリティクスとは何か、そしてなぜそれを行うのか？

アナリティクスとは、データを情報、知識、そして知恵に変えていくことです。

1. 効率化とコスト削減
    1. Human Capital - Location Factor、Offer Accept Rate、Cost Per Hire
    2. ファイナンス - ARR、新規パイプラインの作成、CC障害率
    3. セールスチャネル-勝率、サービス添付率。
    4. マーケティング - 機会の創出、製品のダウンロード
2. ビジネスモデルの改善
    1. プロダクト・インテリジェンス - サインアップの増加、新機能の提供、トップ機能
    2. 顧客分析 - 企業統計、採用傾向、利用とサブスクリプションの比較
    3. お客様の購買動向 - リニューアル、アップグレード、ダウングレード
3. カスタマー・エクスペリエンスの向上
    1. データ製品 - 業界ベンチマーク、データAPI、アルゴリズム
    2. データに基づいたプロセスの改善
    3. インプロダクトインサイト

短期的な方向性

21年度から下期にかけての短期的な目標は、GitLabをデータ・キャパシティ・モデルの「レベル（1）リアクティブ」から「レベル（2）アドバンスド」に引き上げることです。

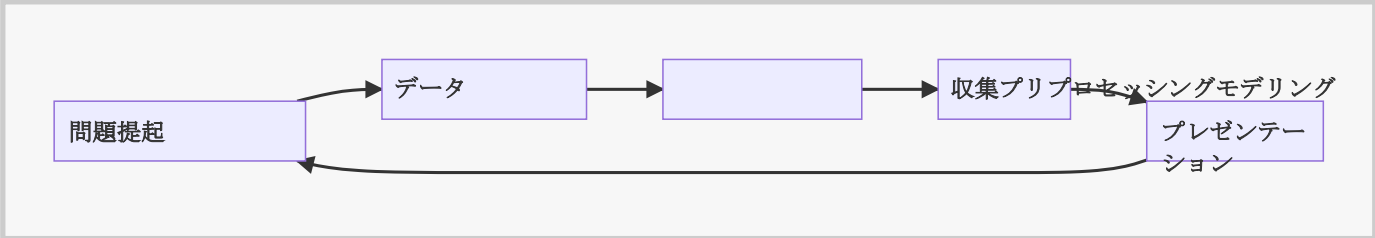
これを実現するために、データアナリティクスチームは

- データフュージョンチームの中で、同じビジネス目標に向かって、同じペースでデ
- ータ分析プロセスを強化することができる。
- GitLabのデータ・ビジュアライゼーションとデータ・ストーリーテリングのスキルを高める

データ分析プロセス

データ分析」「高度なデータ分析」「データサイエンス」は、すべて「データ分析プロセス」から始ま

ります。データ分析プロセスは5つのステップで構成されています。



データ分析では、コンテキストがすべてです。コンテキストは、データアナリストが問題やデータをどのように捉え、どのような方法でデータからの洞察を得るかを導きます。

問題提起

問題記述」のステップでは、ビジネスパートナーと一緒にビジネス上の問題を定義することが重要です。データアナリストは、ビジネスパートナーがデータの洞察を求めているビジネス上の問題を明確に定義することで、その問題が利用可能なデータで技術的に解決可能であることを保証し、その問題に対して最大限のビジネス上の洞察を提供するための成功基準を設定することができます。

データ収集

データ収集のステップでは、データアナリストは、分析の影響や一般化を制限する可能性のある「データの禁忌」を調べて理解し、「データの偏り」や「データの懸念」を評価することが重要です。各ビジネスシステムでどのようにデータが収集されているかを理解することで、正確なデータを適切な方法で報告することができます。このステップでは、信頼性の高い変換を行うために、データタイプが正確であることを確認することが重要です。

したがって、このステップはソースシステムの所有者とのフィードバックループの一部となります。データチームのメンバーが、あるデータフィールドが欠けていたり、正確に把握されていなかったりすることを発見した場合、GitLabの誰もがソースシステムのオーナーに連絡を取り、データフィールドの更新や追加を依頼する責任があります。

前処理

前処理ステップでは、データアナリストがデータを準備し、クレンジングし、データの品質を検証して、スケーラブルなレポート作成に最適なデータ構造を設計します。このステップでは、モデルをレビューし、ビジネスに適したモデルであることを確認することが重要です。

モデリング

モデリングのステップでは、GitLabチームメンバー全員のために、Enterprise Dimensional Model形式の新しいデータモデルを作成することを目指します。まず、新しいデータ構造が正確なビジネスプロセスを反映していることを確認するために、エンティティリレーションシップダイアグラム（ERD）から始めます。すべてのデータモデルは、データエンジニアリングチームによってレビューされます。

GitLabがデータ能力モデルのレベル(4)予測に達した場合、このステップでは探索的データ分析とデータフィーチャリングが行われます。

プレゼンテーション

プレゼンテーションのステップでは、データアナリストは、強力なデータ・ビジュアライゼーションによるデータ・ストーリーテリングのスキルを発揮します。データアナリストは、利害関係者と効果的にコミュニケーションをとるために、実用的なビジネスインサイトをパッケージ化して表示することが重要であると認識しています。実際、ビジネスパートナー（GitLabチームメンバー）にデータを信頼してもらうためには、洞察の背景にある詳細な理由を示すドリルダウン機能を備えたビジネスインテリジェンス（BI）ダッシュボードなどのデータソリューションに、強固なデータ品質チェックを設けることと同じくらい重要です。

データ分析の素晴らしいところは、それぞれの洞察が、解決すべき新たなビジネス上の疑問の波を伝播させ、それによってデータ分析プロセスが再び循環することです。

---

layout: handbook-page-toc title:"データサイエンスハンドブック" 説明"GitLab データサイエンスチームハンドブック"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

ギットラボのデータサイエンス

データサイエンスチームのミッションは、モデルベースのインサイトにより、ビジネス、顧客、製品をよりよく理解することです。このチームは、機械学習モデルを構築し、データを分析して、その結果を信頼できるスケーラブルな製品に変換することで、このミッションを達成します。

プロジェクト

買い方の傾向

- **開始日：2021-06-30 プロジェ**
- **クト**
- **データサイエンスの仕事の内訳 ス**
- **ラックチャンネル**

プロジェクトバックログ

- **購入（アップセル） 予測ARR**
- **ゴールデンジャーニー - グロースチーム - Hila**
- **Qu PtB.first order - グロースチーム - Hila**
  Qu

- **PtB.churn** - カスタマー・サクセス・チーム - **David Sakamoto**

- ユーザーセグメンテーション／ペルソナ活動、ユースケース - 製品チーム -
- Hila Qu コミュニティセンチメント分析／twitter、facebook - 製品チーム
- GitLab MLOps 製品開発 - 製品チーム - Taylor
- Feature $ARR uplift prediction - Product Team - Anoop Dawar

データサイエンスロードマップ - エグゼクティブサマリー

			タイムライン実現可能な価値	ステークホルダー期待される成果
FY-22 Q3 / イテレーティブ	拡大の傾向（PtE 契約傾向（PtC	営業チームがARRを増加させる機会を獲得できるようにする。	セールスチーム（Jake Bielecki	TBD
FY-22 Q4 / イテレーティブ		営業チームがARRの減少を防ぐことができる。	セールスチーム（Jake Bielecki	TBD
FY-23 Q1 / Q2 イテレーティブ	ユーザーセグメンテーション（US	お客様	のプロファイルを決定するプロダクトチーム（Hila Qu	TBD

データサイエンスの責任

- データサイエンスとは一体何なのか！？こちらのビデオやスライドで概要をご紹介しま
- す。データサイエンティストの責任については、こちらをご覧ください。
- さらに、データサイエンスチームは以下のような責務をサポートします。
    - データリーダーシップチームと
        - 分析能力を拡大するためのデータサイエンスのロードマップの策定と公開 データの成果物、進行
        - 中のイニシアチブ、今後の計画についての定期的なアップデートの配信
    - データエンジニアリングチームと
        - GitLabデータシステムのデータ品質に関するプラクティスとプログラムの定義と推進の支援

プロジェクトの構成

データサイエンティストは、CRISP-DM（Cross-IndustryStandardProcessforData Mining）に従い、6つのフェーズで構成されるプロジェクトを遂行します。プロジェクト構造の詳細な内訳とコード例はこちらをご覧ください。

ビジネス理解

最初のステップは、プロジェクトのエンドユーザーのビジネスニーズと成功基準を定義することです。このフェーズには、要件の収集、ステークホルダーへのインタビュー、ビジョンの定義、製品のユーザーストーリー、モデルのユースケースなどが含まれる。このフェーズは、GitLabのOKRや会社の価値観であるイテレーションと密接に関連しています。

データの理解

データの理解は、ビジネスの理解と密接に関連しています。この段階では、ビジネスがすでに導入しているデータソースと、そうでないデータソースを定義する必要があります。この段階では、データサイエンティストは、データエンジニアやデータアナリストと密接に連携し、不一致やリスクを定義します。外部データを導入する必要がある場合は、このフェーズでリクエストを出す必要があります。

データの準備

ビジネスニーズとデータソースを定義した後は、いよいよデータに飛び込んでいきます。この段階では、「2回目のデータ品質チェック」と「探索的データ分析」を行う必要があります。この段階では、データサイエンティストは、次の段階であるモデ

リングに役立つデータ、その種類、相関関係、分布についての理解を深めます。

モデリング

モデリングとは、機械学習モデルを構築することであり、モデルのトレーニング、テスト、バリデーションから構成されます。ビジネス理解の段階では、最初のアルゴリズムを選択し、前処理を定義する必要があります。このフェーズでは、すべてのアルゴリズムがすべてのデータタイプで動作するわけではないため、データ準備の成果が必要となります。

評価

このフェーズでは、モデルのパフォーマンスを測定し（ビジネス・アンダースタンディングで必要な場合）、その結果をステークホルダーやエンドユーザーに提示してフィードバックを得る。このフェーズは、GitLabのバリューの1つである「反復」と強く結びついている。 フィードバックの後、データサイエンティストは、ビジネスゴールが達成される限り、上記のフェーズを反復する。

展開・生産性向上

実行可能な予測モデルが作成されたら、次はそのモデルをTBDデータサイエンスのプロダクションパイプラインに展開します。このプロセスでは、新しいデータが利用可能になると自動的にモデルが更新され（「再学習」）、定期的にすべての対象レコードの予測が生成されます。

成功基準

GitLabのデータサイエンスプロジェクトは、すべて成功基準で始まり、成

功基準で終わります。成功基準は以下のように測定されます。

```
value_provided = future_state - current_state
```

私たちは、データサイエンスプロジェクトから得られるビジネス上の成果として、直接的な影響と間接的な影響の2種類を認識しています。

ダイレクトインパクト

直接的な影響を与えるプロジェクトでは、プロジェクトの成果を直接金銭的価値に置き換えることができます。これは通常、プロセスにかかる時間を短縮したり、企業の収益を増加させたりすることを目的とした最適化プロジェクトによって達成されます。以下は、ダイレクトインパクトの成功基準の例です。

節約できた時間

"私たちのプロジェクトは、プロセスXに費やす時間を10,000時間、500kドルの価値で削減することに貢献しました。"

稼いだお金

"私たちのプロジェクトは、新しい有名な顧客をもたらし、年間収益に50百万ドルを追加しました"

間接的な影響

間接的な影響を与えるプロジェクトは、直接金銭的価値には結びつきませんが、他の意思決定者に金銭的価値をもたらすプラットフォームを提供します。これらのプロジェクトは通常、重要な洞察が拡張可能な製品としてステークホルダーに提供され、意思決定者の能力向上に利用されるものである。以下は、間接的な影響を与える場合の成功基準の例です。

視認性の向上

"私たちのプロジェクトにより、ステークホルダーの意思決定能力が高まり、販売戦略が10%改善されました。"

アンロックインサイト

「私たちのプロジェクトによって、製品チームは主要な顧客が誰であるか、そして彼らがどのような行動をとるかを最終的に理解することができました。この知識によって、ARR100万ドル相当の新規顧客を獲得することができました。"

プロジェクトの洞察／成果の記述

グーグルの公式「XYZ」に慣れること ☑ を実行することで、[Y]によって測定されたように、[Z]を実行します。"

X = 何を達成したかを表す動詞に置き換える必要があります。

Y = は、達成度の尺度に置き換えてください。例えば、収益の増加やコストの削減があった場合、金額や割合を示すことができます。

Z = は、説明したい結果をどのように達成したかを表す動詞に置き換えてください。例えば、"詳細な分析を行うことで"**ABC**を達成した。

フルセンテンスの例は以下の通りです。

☑詳細な分析を行うことで、**10%**の削減を実現します。

上記の式は、プロジェクトの成果をより効率的かつ簡潔に表現するための提案であることにご留意ください。

Gitlabのデータサイエンスツール

- 設定済みの**JuypterLab**イメージ。データサイエンスチームは、一般的な**python**モジュール（**pandas**、**numpy**など）、ネイティブの**Snowflake**接続、**git**サポートなどがあらかじめ設定された**JupyterLab**を使用しています。共通のフレームワークで作業することで、モデルの作成やインサイトの導出を迅速に行うことができます。このセットアップは誰でも自由に使用することができます。詳細は**JupyterGuide**をご覧ください。
- **gitlabds** パイソンツール。一般的なデータ前処理（ダミーコーディング、外れ値検出、変数削減など）やモデリング作業（モデルの性能評価など）を自動化するための関数です。`pip install gitlabds`で直接インストールするか，上の**JupyterLab**イメージの一部として使用します.
- モデリングテンプレート **(Coming Soon!)**

データサイエンスと機械学習に関する有用なリソース

- ジェイク・バンダープラス著「***Python**データサイエンスハンドブック*」。**Python**を学び、データサイエンスに足を踏み入れようとしている初心者の方に最適です。
- **Sebastian Raschka & Vahid Mirjalili**による***Python Machine Learning***。**python**の基礎レベルを前提とした、より高度なトピックを扱っています。
- ***The Elements of Stastical Learning, Data Mining, Inference, and Prediction***」（**Trevor Hastie**、**Robert Tibshirani**、**Jerome Friedman**著）。一般的に使用されている予測技術の背景にある統計や論理を深く掘り下げて解説しています。ただし、統計学や数学の知識が必要な場合もあります。

データサイエンスの共通用語

以下は、チームで使用される一般的なデータサイエンス用語です。

データサイエンス（**DS**） - コンピュータサイエンス、統計的手法、分野の専門知識を用いて、データから洞察を引き出す学際的な分野

機械学習（**ML**） - データのパターンを決定するために、明示的にプログラムされていないアルゴリズムの使用と開発

アルゴリズム - 特定の問題を解決するために使用される、コンピュータに実装可能な命令のシーケンス

特徴 - 国、年齢など、分析に使用できるデータセット内の単一列。変数や属性と呼ばれることもある

Feature Engineering - データを選択し、組み合わせ、特徴に変換するプロセス **Weight** - 特徴の強

さを決定するために割り当てられる数値

モデル - 入力データに基づいて決定された重みを持つ、適用されたアル

ゴリズム インピュテーション - 欠損したデータを置換した値で置き換え

るプロセス トレーニング - 入力データに基づいてモデルに重みを割り

当てること

テスト - 学習したモデルの予測値と実際の値を比較する 分類 - 各データポイ

ントのカテゴリーを予測するプロセス

回帰 - 各データポイントについて連続的な数値変数を予測するプロセス クラスタリング -

データセット内のグループ化を定義するプロセス

PythonとSQLの比較

SQL（Structured Query Language）に慣れている方は、Python（ライブラリpandasを使用）で分析を行う際に句を使用したいと思うかもしれません。以下に、一般的なSQLコマンドに相当するPythonのコマンドを示します。

なお、以下の方法は、期待通りの結果を得るための唯一の方法ではなく、同じ結果を得るためにPythonには多くの方法があることをご理解ください。

SELECT * FROM

```
#データセットを'data'として読み込んだと仮定すると、Jupyterのセルdataにデータフレームの名前を入れるだけです。
```

SELECT col1, col2 FROM

```
columns = ['col1',

'col2'] data[columns].

上記は次のように1つのコードにまとめることができます: data[['col1','col2']].
```

COUNT

```
#データフレームの長さを定義する（行数に相当） len(data)

上記と同様ですが、さらに列数を表示します data.size
```

LIMIT

```
#最初の5行をリストアップ data.head()

#上位10行を表示したい場合は、head()の中に数字を入れることで実現できます function
data.head(10)
```

WHERE

```
#column1が1以上の値を持つすべての行をリストアップ condition
= data['column1'] > 1

データ[条件]
```

Pythonは複合条件を受け入れることに注意してください。PythonのANDに相当するものは&、ORに相当するものは｜です。

GROUP BY（集計機能付き

```
#列ごとの行数をリストアップ 1列目
data.groupby(['col1']).size()

#列1ごとの行の平均値を列挙する
data.groupby(['col1']).mean()

#列1ごとの行の最小値を列挙する
data.groupby(['col1']).min()

#列1ごとの行の最大値を列挙する
data.groupby(['col1']).max()
```

---

layout: handbook-page-toc title:"Data Engineering Handbook" description:"GitLab データエンジニアリングチームハンドブック"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

ギットラボのデータエンジニアリング

データエンジニアリングチームのミッションは、安全で信頼できるデータプラットフォームを構築し、誰もがアナリストになれるようにして、*唯一*の制限がデータやアナリスト自身になるようにすることです。私たちは、GitLabの価値観とデータチームの原則によってこれを実現します。

データエンジニアリングの責任

Data Team's Responsibilitiesのうち、データエンジニアリングチームは以下を直接担当しています。

- すべての**GitLab**チームのレポーティング、分析、ディメンショナルモデリング、データ開発をサポートするための、会社の中心となるエンタープライズデータウェアハウスの構築と維持
- 新しいデータソースを統合し、対象分野、活動、プロセスの分析を可能にする。
- **Single Source of Truth**を実現するための**Enterprise Dimensional Model**の構築と維持
- マスターデータ、リファレンスデータ、データ品質、データカタログ、データパブリッシングなどのデータマネジメント機能の開発
- 誰もがデータと分析を活用できるよう、セルフサービスのデータ機能を提供する
- **GitLab**データシステムのためのデータクオリティープラクティスとプログラムの定義と推進の支援

- データモデリング、データ品質、データ統合など、カスタマイズ可能なデータサービスの提供

データ・エンジニアリング・チームの最初のハンドブック

GitLabではHandbook Firstを掲げています。コードベースやプロセスに変更を加える場合は、作業を開始する前に、あるいはコードを変更するためのMRと同時に、ハンドブックのMRを作成しなければなりません。

例えば、以下のようなものがあります。データソースを追加する場合、まず、ハンドブックの抽出とロード、システム図のセクションを更新するために、マージリクエストを作成します。このMRは、変更が作成されると同時に更新され、変更と一緒にマージされます。

データエンジニアリングチームの役割

チームとして効率的に活動するために、私たちはデータエンジニアリングチームのメンバーに以下の責任を指定し、割り当てます。これらの割り当ては永続的なものではありませんが、これらの責任についてチーム内に直接責任を負う個人が誕生します。割り当てられたエンジニアは、これらの責任に対するメンテナンスと反復的な改善の両方に責任を負います。

プロダクション・オペレーション

本番運用は、データエンジニアリングチームにとって常に最優先事項であり、本番環境とそれがサポートするSLOをサポートまたは影響するすべての技術的運用を含みます。これには、モニタリング、テスト、デプロイメント、コードレビュー、および一般的なDevOpsが含まれますが、これらに限定されるものではありません。

プロジェクトマネジメントとプランニング

プロジェクト管理・計画を担当するエンジニアは、ワークフロープロセス、マイルストーン計画、および課題のトリアージを監督、指導、改善します。作業が効率的に計画・完了され、データチームと GitLab の両方の目標を適切にサポートし実現することを保証します。

ユーザーエクスペリエンス

このエンジニアは、当社のデータプラットフォームのユーザビリティを担当します。データエンジニアリングのミッションにもあるように、私たちはすべての人のためのプラットフォームを構築しています。割り当てられたエンジニアは、最も使いにくい経験を念頭に置いて、データプラットフォームに関する解決策を積極的に見つけ出します。

セキュリティ、コンプライアンス、プライバシー

この役割は、当社のプラットフォームの安全性、プライバシー、およびコンプライアンスを維持し、監査に関する質問の窓口となるだけでなく、当社のデータプラットフォームのセキュリティとプライバシーに影響を与える変更を検討する主要なエンジニアとしての役割も果たします。また、当社のプラットフォームのセキュリティ機能を繰り返し改善していく必要があります。

データウェアハウスアーキテクチャ

イタレーションを安価に抑え、ウェアハウスの使い勝手を向上させるために、ディメンションウェアハウスの導入を決定しました。このエンジニアは、当社のウェアハウス・アーキテクチャの完全性を守り、アーキテクチャの改善が必要な箇所のビジョンを導く役割を担っています。これらの変更や保護はすべて、社内外のお客様を第一に考えたものでなければなりません。

---

layout: handbook-page-toc title:"Data Team Platform" 説明"GitLab データチーム・プラットフォーム"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

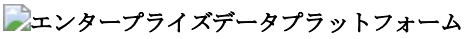
- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

クイックリンク

- データインフラ デ
- ータパイプライン
- データCIの仕事
- **dbt Guide**
- エンタープライズデータウェ
- アハウス Jupyterガイド
- パーミフロスト
- パイソンガイド
- **Sisense (Periscope)**
- **Sisense Style Guide**
- **Snowplow**
- **SQL**スタイルガイド

データスタック



アナリティクス機能の運用・管理には**GitLab**を使用しています。すべてはイシューから始まります。変更はマージリクエストによって行われ、パイプライン、抽出、ロード、変換、分析の一部などが変更されます。

	ステージツール
エクストラクトスティッチ	、フィベトラン、カスタム
LoadingStitch, Fivetran, and Custom	
	OrchestrationAirflow
データ	ウェアハウススノーフレ
ークトランスフォーメーション**dbt**と**Python**スクリ	
プトデータビジュアライゼーション	**Sisense For**
Cloud Data Teams Advanced	
	Analyticsjupyter

抽出と読み込み

現在、データソースのほとんどに**Stitch**と**Fivetran**を使用しています。これらは既製の**ELT**ツールであり、データソースから**Snowflake**データウェアハウスへのデータの移動を構築、維持、または指揮する必要がありません。**Stitch/Fivetran**のすべてのデータソースのフルリフレッシュは、セキュリティ認証を変更するのと同時に行います（約**90**日ごと）。フル・リフレッシュを実行する前に、すべてのテーブルを削除します。

Stitchと**Fivetran**は、データパイプラインの開始を自分たちで処理します。つまり、**Airflow**は**Stitch**や**Fivetran**のスケジュールのオーケストレーションには関与していません。

ソースオーナーシップについては、**TechStackApplications**シート（社内限定）をご覧ください。

データソース

次の表は、データウェアハウスにロードしているすべての**RAW**データソースのインデックスです。開発のバックログと優先順位は、**NewDataSource/PipelineProjectManagement**シートで管理しており、最新のステータスと進捗管理のために**GitLab issues**へのリンクを設けています。新しいデータソースのハンドブックのページでは、データチームが新しいデータソースのリクエストをどのように処理するかを説明しています。

キー

- パイプライン。データを複製するための技術です。

- **RF（Replication Frequency）**：新しいデータや更新されたデータをどのくらいの頻度で読み込むか。

- **Raw Schema**。データが保存されているRAWデータベースのスキーマ。
- **PREP Schema**。ソースモデルが実体化しているPREPデータベースのスキーマ。オーディエ
- ンス。データの主要ユーザー。
- **SLO：Service Level Objective**の略。私たちの**SLO**は、リアルタイムからデータビジュアライゼーションで表示される分析結果までの時間を
- ツール
  xは未定義または実行されていないことを示す

データソース	パイプライン	生スキーマ	プリップスキーマ		MNPI SLO	
アダプティブ	メルタノ	tap_adaptive		ceYes		Finan
エアフロー	ステッチ	エアフロー・ステッチ	エアフロー	データチーム	24h / 24h	No
BambooHR	エアフロー	バンブーアール	センシティブ	People	12h / 24h	No
クリアビット	x	x	x	x /		xNo
カスタマードット	pgp	tap_postgres	お客様	製品	24h / x	No
ERD デマンドベース	スノーフレータスク	デマンドベース	デマンドベース	Marketing	24h / x	No
ジッター	x	ジッター	x	x	x	いいえ
GitLab. です。	compgp		tap_postgresgitlab_dotcom	製品 エンジニアリング	6h / xNo	
GitLab Ops DB	pgp	6h /	tap_postgresgitlab_		opsEngineering	
GitLab Profiler DB					x xxxxx /	xNo
Google アナリティクス 360	ファイブトラン	google_analytics_360_fivetran	google_analytics_360	Marketing	6h / 32h	No
Google Cloud ビリング		xgcp_billing	gcp_	billingEngineering	x /	xNo
グラファイトAPI	エアフロー No	エンジニアリング_エクストラクト		xエンジニアリング	24h / 48h	
グリーンハウス		シートロード	greenhousegreenhouse	People	24h / 48h	No
ハンドブックのYAML ファイル					ハンドブック Git ログデータ	
ハンドブックMRデータ						

エアフロー 24h エアフロ			gitlab_data_yamlgitlab_data_	yamlMultiple ー ハン	ドブックハンドブックマルチプル 8h / 24 h / 24h	No No
	エアフロー	ハンドブックハンドブックマルチプル		ン 1w /	ノー1m	
ライセンスドット ERD	オートマチックプロセス 品		ライセンス_dblicense_db	製	24 h / 48 h	No
マルケト	ファイブトラン		marketoxMarketing		24h / 24h	No

データソース	PipelineRaw	SchemaPrep	SchemaAudienceRF / です。 SLO	MNPI
Netsuite	ファイブトラン	netsuite_	fivetrannetsuiteFinance 6h / 以下はその例です。 24h	はい。
	PMGxpmgpmgxx / xNo			
PTOby	RootsSnowpipeptogitlab_pto		工学的生産性人 7 / x	daysNo
クオルトリックス	エアフロー QUALITRICSMARKETING	12H /	QUALITRICS NO 48h	
SaaSの利用	エアフロー saas_usage_pingsaas_usage_ping	製品Ping	1 / x	weekNo
セールスフォース	スティッチ stitchsfdcSales 6h / 1枚		salesforce_ 24h	はい。
シートロードSheetLoad	sheetloadsheetloadMultiple	24h /	48h	はい。 いいえ
スノープラウ	スノープラウ 15m	スノープラウスノープラウ	製品 / 24h	
サノスレーク	スノーフ タスク	グ prometheusprometheus / xNo	24 h エンジニアリン	
バージョンDB	オートマチック プロセス品	バージョン_dbversion_db 製	24 h / 48 h	No
Xactly	メルタノ	tap_	xactlyN/ASales 24h / N/A	はい。 いいえ
Zendesk	スティッチ stitchzendeskSupport	zendesk_ 6h /	24h	いいえ
ズーム	メルタノ	tap_	zoomN/APeople 24h / N/A	はい。
ズーラ	スティッチ	zuora_	stitchzuoraFinance 6h / 24h	48h
Zuoraの収益	エアフロー	24h / zuora_revenuezuora_	revenueFinance	アナ

リティクス&					
Zoominfo	はい。Snowflakeデータシェア	N/o	Azoominf	インサイト、プロダクト、セールス、マーケティング	24時間 / N/A

新しいデータフィールドの追加

BambooHRの抽出物に新しいフィールドを追加する。

- **AnalyticsプロジェクトでBambooHRテンプレートを使用して新しい課題を作成します。**
- **Data Team ManagerおよびCompensation and Benefits Managerの承認を得る。**
- **承認されたら、Compensation and Benefits Managerと、抽出物を検証するData Engineerに割り当てます。**

データチームによるデータソースへのアクセス

新しいデータソースをデータウェアハウスに統合するために、データチームの特定のメンバーは、**UI**と**API**の両方で、データソースへの管理者レベルのアクセスが必要になります。適切な分析を構築するために必要なすべてのデータを引き出すためには、**API**を通じてこの管理者レベルのアクセスが必要であり、準備された分析の結果を**UI**と比較するためには、**UI**を通じてのアクセスが必要となります。

機密性の高いデータソースは、必要なレポートを作成するためのアクセス権を持つデータエンジニアとデータアナリストを**1**名以上に限定することができます。場合によっては、**2**人のデータエンジニアに限定することもあります。自動抽出プロセスのために、追加のアカウントを要求することもあります。

機密データは以下のセキュリティパラダイムによってロックダウンされます。**Sisense**はデフォルトではいかなるデータにもアクセスできないため、機密データにアクセスすることはありません。**Sisense**のアクセスは常に明示的に許可されます。

DataSiren

データチームがデータウェアハウスのどこに機密データがあるかを完全に把握し、**Sisense**が機密データにアクセスできないようにするために、**dbt**と社内で開発されたツールのライブラリ`datasiren`を使ってデータウェアハウスの定期的なスキャンを行っています。このスキャンは現在、週**1**回のペースで実行されています。細かい結果は、**Snowflake**の`PREP.DATASIREN`スキーマに格納され、機密性の理由から**Periscope**では利用できません。高レベルの結果は**Periscope**で公開されており、ここにあるシンプルなダッシュボードもその一つです。

Qualtrics Mailing List Data Pump / Qualtrics SheetLoad

Qualtricsのメーリングリストデータポンププロセスは、コード上では`Qualtrics SheetLoad`とも呼ばれ、データウェアハウスから**Qualtrics**にメールをアップロードする際に、チームメンバーのマシンにダウンロードする必要がありません。このプロセスの名前が**SheetLoad**と同じなのは、**Google Sheets**を使って`qualtrics_mailing_list`で始まる名前のファイルを探すためです。見つかったファイルのうち、**1**列目に`id`列があるものについては、そのファイルを**Snowflake**にアップロードします。そしてできあがったテーブルを**GitLab**のユーザーテーブルと結合し、メールアドレスを取得します。その結果を新しいメーリングリストとして**Qualtrics**にアップロードします。

プロセス中、**Google** シートはプロセスの状態を反映して更新されます。最初の列の名前は、プロセスの開始時に`処理中`に設定され、その後、メーリングリストと連絡先が**Qualtrics**にアップロードされたときに`処理中`に設定されます。カラム名を変更することで、依頼者にプロセスの状態を知らせ、デバッグを支援し、各スプレッドシートでメーリングリストが一度しか作成されないようにします。

エンドユーザーエクスペリエンスについては、**UX Qualtrics**の

ページに記載されています。デバッグ

スプレッドシートにエラーが発生し、リクエストファイルに明らかな問題がない場合は、通常、スプレッドシートの再処理を試みることが最初の行動となります。過去には、新しい **GitLab** プラン名が `gitlab_api_formatted_contacts` **dbt** モデルに追加されたときや、**Airflow** タスクがファイルの処理中にハングアップしたときに、再処理が必要になりました。この処理は、スプレッドシートの所有者が、この処理によって作成された部分的なメーリングリストを使用していないことを確認し、スプレッドシートに追加の変更を加えないようにするために、協調して、または要求に応じてのみ実行する必要があります。

Qualtrics Mailing Listリクエストファイルを再処理するには、以下の手順で行います。**1.Airflow**で**Qualtrics Sheetload DAG**を無効にします。**1.**エラーとなったスプレッドシートから作成された**Qualtrics**のメーリングリストをすべて削除します。`Qualtrics - API`のユーザー認証情報を使用して**Qualtrics**にログインし、メーリングリストを削除することができるはずです。メーリングリストの名前は、`qualtrics_mailing_list`の後のスプレッドシートファイルの名前に対応します。この名前は、スプレッドシートファイルのタブの名前と同じである必要があります。**1.Airflow** で **Qualtrics Sheetload DAG** を再度有効にし、**Airflow** のタスクログを注意深く監視しながら実行します。

スノープラウ・インフラ

セットアップの詳細については、「スノープラウ・インフラストラクチャー」のページを参照してください。

データソースの概要

- カスタマーサクセスダッシュボード
- Netsuite
  - Netsuiteとキャンペーンデー
- タのバージョン(ping)
  - なお、2019年10月までは、データチームはバージョンデータソース全体を「ping」と呼んでいました。しかし、Usage pingはバージョンデータソースの1つのサブセットに過ぎないため、現在では、version.gitlab.comデータソースを参照するために「version」または「version app」を使用し、バージョンデータソースの特定の使用データ機能を参照するために「Usagedata」または「Usagepings」または「Pings」を使用しています。データの抽出において、`Serviceping`データの取り込みに関しては、Service pingページまたはService pingのReadme.mdページに詳細が記載されています。
- Salesforce
- Zendesk

オーケストレーション

オーケストレーションにはAirflow on Kubernetesを使用しています。私たちの具体的なセットアップ/実装方法はこちらをご覧ください。また、データ基盤のページにも詳細が記載されています。

データウェアハウス

現在、データウェアハウスとしてSnowflakeを使用しています。エンタープライズデータウェアハウス（EDW）は、GitLabの企業データ、パフォーマンス分析、主要業績評価指標などの企業全体のデータのための唯一の情報源です。EDWは、レポーティング、ダッシュボード、分析のための共通のプラットフォームとフレームワークをすべてのチームに提供することで、GitLabのデータドリブンな取り組みをサポートします。ポイントツーポイントのアプリケーション統合を除いて、現在および将来のデータプロジェクトはすべてEDWから推進されます。EDWは、GitLabの様々なソースシステムからのデータを受信することで、データ品質のベストプラクティス、対策、改善策を伝え、推進し、可能な限り最高のデータを使ってすべての意思決定がなされるようにします。

Snowflakeサポートポータルへのアクセス

Snowflakeサポートポータルにアクセスするには、以下の手順を実行してください。

- gitlabのメールアドレスをコミュニティポータルに登録
- この登録により、gitlab のメールに `Welcome to the Snowflake Community` という件名の歓迎メールが送信されます。このメールでは、登録を完了するように求められ、その一環としてコミュニティポータルのパスワードを設定するように求められます。
- 終わったら、snowflakeのコミュニティアカウントに再度ログインし、ホームページで「ケースの送信」をクリックします。初めて、アクセス権を持たないユーザーがsnowflakeでケースを送信することができます。アクセスのためのフォームを入力するように求められます。
- フォームでは、すでにスノーフレークのお客様のアクセスを選択します。次のページでは、`Account Name（アカウント名）`、`Cloud Name（クラウド名）`、`Region Name（地域名）`の情報が求められます。以下は、snowflakeコンソールからこの情報を引き出す1つの方法です。
  - `アカウント名` - CURRENT_ACCOUNT()を選択します。
  - `地域名`- CURRENT_REGION()を選択します。
  - `クラウド名` - リージョン名の値に基づいて、クラウド名を特定できます。
- 申し込みが完了すると、「`Request received] Case#` instant」という件名の確認メールが届きます。万が一、メールが届かない場合は、フォームを再送信してください。
- 24時間以内に、件名が`「Case# -Self Register - Enable Case access」`のリクエスト確認メールが届くことを掲示する。

倉庫へのアクセス

データウェアハウスにアクセスするために

- アクセス要求プロジェクトで課題を作成し、必要なアクセスレベルを文書化します。共
- 有アカウントをリクエストしてはいけません。各アカウントはユーザーに関連付けられている必要があります。

- 私たちは、このブログ記事で説明されているユーザーの許可に関するパラダイムにゆるやかに従っています。
- 既存のアカウントのミラーリングを依頼する際には、制限されたSAFEデータへのアクセスはプロビジョニング／ミラーリングされないことに注意してください（現在はrestricted_safeロールを通じて提供されています）。

SAFEデータへのアクセス

すべてのSAFEデータは、Snowflakeの別々のデータベーススキーマ内のテーブルに保存されます。1つのテーブルにアクセスすると、すべてのSAFEテーブルにアクセスできます。SAFEデータへのアクセスには

1. あなたの直属の上司の承認
2. 部門のVP（またはそれに相当する）の承認を得ていること。
3. GitLab Dashboard Indexに定義されているSAFE Dashboard Space Ownerの承認。

SAFEデータにアクセスするには

1. アクセスリクエストを作成し、お客様のニーズと意図をお伝えください。
2. 直属の上司、部門担当副社長（またはそれに相当する人物）、およびGitLabダッシュボードのインデックスヘッダーに定義されているSAFEスペースオーナーに承認を求める。SisenseでSAFEデータへのアクセスが60日以内に承認されていれば、承認は必要ない。このステップをスキップして、特定のARにリンクする。
3. リクエストが承認されると、Snowflakeのプロビジョナーにタグを付け、プロビジョナーがリクエストを処理します。
4. 処理が完了すると、SnowflakeでSAFEデータ（スキーマ）にアクセスできるようになります。

スノーフレーク パーミッション パラダイム

Snowflakeのパーミッション管理にはPermifrostを使用しています。私たちのSnowflakeインスタンスの設定ファイルは、このroles.ymlファイルに格納されています。また、Permifrostに関するハンドブックのページもあります。

私たちは、このような一般的な戦略に基づいて役割管理を行っています。

- すべてのユーザーは、関連するユーザーロールを持っています。
- 機能的な役割は、共通の権限セットを表すために存在します（analyst_finance、data_manager、product_manager
- ） データの論理的なグループには、それぞれのオブジェクトの役割があります。
- オブジェクトの役割は主に機能的な役割に割り当てられる
- より高い権限を持つロール（accountadmin、securityadmin、useradmin、sysadmin）がユーザーに直接割り当てら
- れる サービスアカウントには同じ名前のロールが割り当てられる
- 追加の役割は、用途やニーズに応じて、サービスアカウントの役割またはサービスアカウント自体に割り当てることができます。
- テーブル＆ビューの粒度で個別の権限を付与可能
- Warehouseの使用は、必要に応じてどのような役割にも付与することができますが、機能的な役割に付与

することをお勧めします。 ユーザーの役割

すべてのユーザーは、ユーザー名と一致する独自のユーザーロールを持ちます。オブジェクトレベルのパーミッション（データベース、スキーマ。
テーブル）は、Snowflakeではロールにのみ付与することができます。ロールはユーザーまたは他のロールに付与することができます。私たちは、ユーザーがデータベースと対話するために1つのロールを使用する必要があるように、すべての権限がユーザーロールを介して流れるように努めています。例外として、 accountadmin、securityadmin、useradmin、sysadmin などの特権ロールがあります。これらのロールは、より高いアクセス権を与えるため、使用する際には意図的に選択する必要があります。

機能的役割

機能的な役割とは、一般的にジョブファミリーにマッピングされる権限と役割付与のグループを表します。主な例外は、アナリストロールです。アナリストロールには、組織のさまざまな領域に対応するいくつかのバリエーションがあります。
analyst_core、analyst_finance、analyst_peopleなどがあります。アナリストは関連するロールに割り当てられ、必要なスキーマへのアクセスが明示的に許可されます。

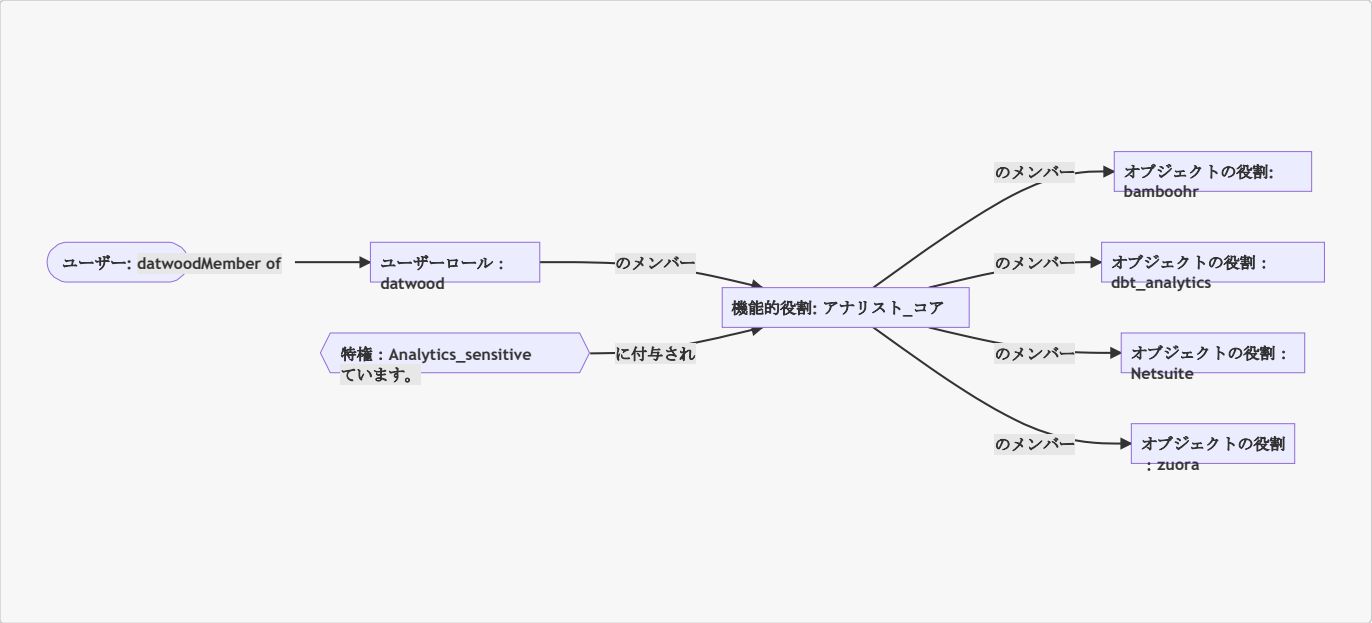
機能的な役割はいつでも作成することができます。これは、非常に似通ったジョブファミリーと権限を持つ複数の人がいる場合に最も意味のあるものです。

オブジェクトの役割

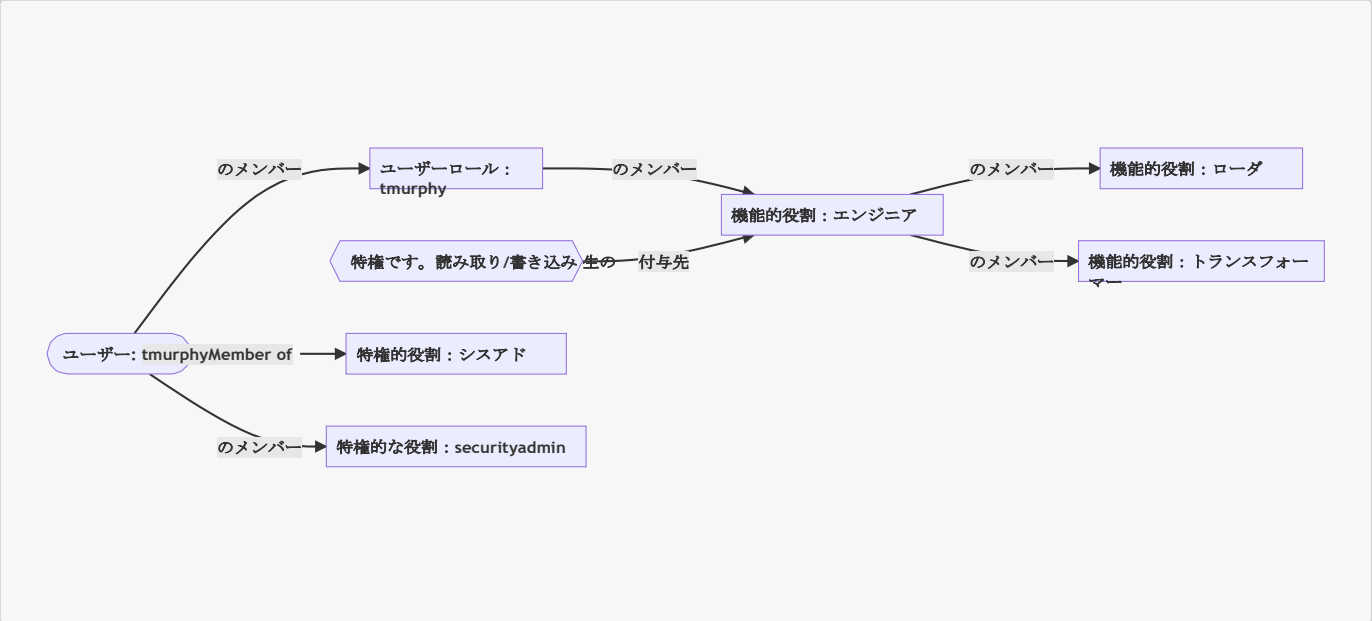
オブジェクトロールは、一連のデータへのアクセスを管理するためのものです。一般的には、特定のソースのすべてのデータを表します。`zuora`オブジェクトロールはその一例です。このロールは、**Stitch**から送られてくる生の**Zuora**データと、`prep.zuora`スキーマのソースモデルへのアクセスを許可します。ユーザーが**Zuora**データへのアクセスを必要とする場合、そのユーザーのユーザー・ロールに`zuora`ロールを付与するのが最も簡単な解決策です。何らかの理由でオブジェクト・ロールへのアクセスが意味をなさない場合は、テーブルの粒度で個別の権限を付与することができます。

例

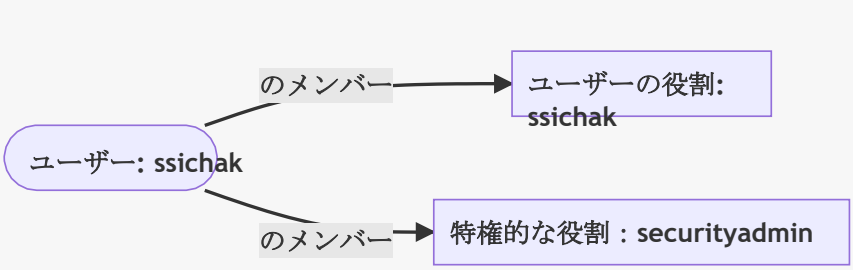
これは、**Data Analyst, Core**の役割階層の例です。



これは、データエンジニアとアカウントアドミニストレーターの役割階層の例です。



これは、セキュリティオペレーションエンジニアの役割階層の例です。



Snowflakeのロールを管理する {: .panel-heading}。

ここでは、新しいユーザーとユーザーロールをプロビジョニングするための適切な手順を説明します。

- プロビジョニングラベルが適用されたオリジナルのリクエストをリンクするGitLab Data Teamプロジェクトの課題を確認します。
- Snowflakeにログインし、securityadminロールに切り替えます。
  - すべてのロールがsecurityadminの所有下にあること
- user_provision.sqlスクリプトをコピーし、初期ブロックのemail、firstname、lastnameの値を置き換えます。 パ
- スワードが必要な場合は、PasswordGeneratorを使って作成します。
  - ユーザー名とパスワードの認証情報をワンタイムシークレットまたはSlack経由でユーザーに送信
- Snowflake roles.yml permifrost設定ファイル内のドキュメント（このファイルは毎日12:00a.m.（UTC）に自動的に読
  - み込まれます） 作成したユーザーとユーザーロールを追加する
  - 新規ユーザーへのユーザーロール
  - の割り当て ユーザーへの追加ロー
    ルの割り当て
- ユーザーがOktaでアプリケーションを割り当てられていることを確認する
- ユーザーがokta-snowflake-usersのGoogleグループに割り当てられていこ

とを確認する 既存のユーザーをデプロビジョニングするための適切な手順を説明

します。

- Snowflakeのデプロビジョンは、オフボーディングの問題またはアクセスリクエストの問題を介して行う必要があります。
- GitLab Data Team プロジェクトで、オリジナルのソースリクエストにリンクしているイシューがあることを確認します。デプロビジョニングラベルが適用されています。
- Snowflakeにログインし、securityadminロールに切り替えます。
  - すべてのロールはsecurityadminの所有下にある必要があります。
- user_deprovision.sqlスクリプトをコピーして、USER_NAMEを置き換えます。ユーザーを削除せずにsnowflakeに残し、disabled = TRUEを設定する理由は、ユーザーがいつアクセスできなくなったかを記録するためです。
- okta-snowflake-usersのGoogleグループからユーザーを削除します。
- Snowflake roles.yml permifrost設定ファイルのユーザーレコードを削除する（このファイルは毎日12:00a.m. UTCに自動的に読み込まれます

詳しくは、このペアリングセッションの録画をご覧ください（GitLab Unfilteredとして視聴する必要があります）。

コンピューティングリソース

Snowflakeのコンピュートリソースは「ウェアハウス」と呼ばれています。クレジット消費量をよりよく追跡・監視するために、倉庫にアクセスする人に応じていくつかの倉庫を作成しました。倉庫の名前には、そのサイズが付加されています（analyst_xsはextra small）。

ウェアハウス・パーポゼッション・マックスクエリ

(分）となります。)

admin	パーミッション・ボットやその他の管理作業のためのものです。	10
airflow_testing_l	局所的な気流のテスト用	30
アナリスト_*さん	これらは、データアナリストがデータベースへの問い合わせやモデリングを行う際に使用します。データ	30
エンジニア_*さん	これらは、データエンジニアやマネージャーが、データを照会する際に使用します。データベースやモデリングデータ	30
fivetran_warehouse	これはFivetranが使用するための専用のものです。	30
gitlab_postgres	これは、GitLab内部のPostgresデータベースから抽出するジョブのためのものです。	10
グラファナ	これは、Grafanaが使用するための専用	60
ローディング	これは、当社のExtract and Loadジョブのためのものです。	60
merge_request_*	これらは、マージリクエスト内のdbtジョブのGitLab CIにスコープされていま	60

ウェアハウス・パーポゼッション・マックスクエリ

(分）となります。)

報告	これはBIツールでのクエリ用です。なお、Sisenseは4分を強制的にのタイムアウトが発生します。	30
ステッチ	これはスティッチが使用するための専用のものです	30
ターゲット_スノーフレーク	これは、MeltanoチームがSnowflakeローダーをテストするためのものです。	5
transforming_*。	これらはプロダクションDBTジョブ用です。	60

クエリの時間が制限されている場合は、より大きな倉庫の使用を検討してください。

データストレージ

私たちは、raw、prep、prodという3つの主要なデータベースを使用しています。rawデータベースは、Snowflakeに最初にデータが読み込まれる場所で、その他のデータベースは、分析の準備が整った（または整いつつある）データ用です。

スノーフレークデータベースというものがあり、そこにはGitLabインスタンス全体の情報が含まれています。これには、すべてのテーブル、ビュー、クエリ、ユーザーなどが含まれます。

covid19データベースがあり、これはSnowflake Data Exchangeで管理されている共有データベースです。

testing_dbデータベースがあり、Permifrostのテストに使用されています。

roles.yml Permifrostファイルで定義されていないすべてのデータベースは、週単位で削除されます。

Sisense	rawNo.の
DatabaseViewable	
	PREPNO
	prodYes

生

このデータベースは、Sisenseではクエリできません。このデータに対するdbtモデルは存在しません。

- 生データには機密データが含まれている可能性があるため、権限を慎重に管理す
- る必要がある データはソースに応じて異なるスキーマに保存される
- スキーマやテーブルごとにユーザーのアクセスを制御できる

プリップ

このデータベースは、Sisenseでは照会できません。

- ソースモデルは、データソースに対応する論理スキーマで構築されます（例：sfdc、zuora）
- PREPARATION - これは、dbtモデルが構築されるデフォルトのスキーマです。
- SENSITIVE

Prod

このデータベースと、その中のすべてのスキーマとテーブルは、Sisenseが問い合わせ可能です。

public と boneyard を除いて、すべてのスキーマは dbt で制御されています。詳細はdbtガイドを参照してください。分

析プロジェクトのフォルダ構造

以下の表は、アナリティクス・プロジェクトのmodels/ディレクトリ内のフォルダに格納されたモデルが、データウェアハウスでどのように実体化されるかを示すマッピングです。

これは、dbt_project.ymlという設定ファイルに記載されています。

スノーフレークのフォルダー dbt/models/		Sisenseの	db.schemaDetailsQueryable
d.common	common/pro	ファクトのトップレベルフォルダとの寸法を表示します。ここにはモデルを入れないでください。	はい。
	common/prep/prep.prep.	prepモデルを使用して、No facts/dimsを作成する。	
	common/sensitive/prep.	sensitiveFacts/dimsには、機密データが含まれていません。	ま
レート/prod.curateYes			コモン/キュ
ト/	ディムテーブル。はい。	common/prod/prod.commonProductionのファク	
	common_mapping/prod.common_mapping	マッピング、ブリッジ、またはルックアップテーブルを含みます。	はい。
common_mapping/prep/	prod.common_mapping	の準備テーブル。マッピング、ブリッジ、ルックアップテーブル	はい。
みます。	marts/prod. marts はい。	マートレベルの	データを含
	prep/prep.準備	生産のための一般的な準備モデル。	いいえ。
	legacy/prod.legacy	非次元的に構築されたモデルを含みます。	はい。
	source/prep.source	ソースモデルを含んでいます。スキーマはデータソースに基づく	いいえ
ワークスペース/	prod.workspace_workspace	SQLの対象とならないワークスペースモデルやdbt規格。	はい。
共通／制限付き	prod.restricted_domain_common	制限されたファクトとディメンションのトップレベルフォルダ。レギュラーの同等品共通のスキーマではなく、限定されたデータを対象としています。	はい。
common_mapping/resticted	prod.restricted_domain_common_mapping	制限付きのマッピング、ブリッジ、またはルックアップテーブルを含みます。レギュラーのエクイップメント共通のマッピングスキーマを使用していますが、制限されたデータを対象としています。	はい。
マート／制限付き	prod.restricted_domain commonmarts	はい。	
レガシー／リストリクテッド	prod.restricted_domain_legacy	無次元的に構築された制限付きモデルを含む。通常のレガシーに相当するスキーマではなく、制限	はい。

dbtで自動的に更新することなく、ユーザーのためにデータを保存する必要があるデータウェアハウスのユースケースでは、`STATIC`データベースを使用します。これにより、アナリストや他のユーザーが独自のデータ・リソース（テーブル、ビュー、その他）を作成することもできます。

一時的なテーブル）を使用しています。)静的データベースの中には、機密データ用のスキーマがあります。静的データベースのユースケースで、機密データの使用または保存が必要な場合は、データエンジニアのために課題を作成してください。

タイムゾーン

ウェアハウス内のすべてのタイムスタンプデータは、UTCで保存する必要があります。SnowflakeセッションのデフォルトのタイムゾーンはPTですが、これをオーバーライドしてUTCがデフォルトとなっています。これは、`current_timestamp()`がクエリされたときに、結果がUTCで返されることを意味します。

Stitchはタイムスタンプ明示的にUTCに変換します。Fivetranも同様に行っています（サポートチャットで確認済み）。

このルールの唯一の例外は、ファクトテーブルの date_id の作成に太平洋時間を使用することです。これは常に `get_date_pt_id` dbt マクロで作成し、`_pt_id` サフィックスを付ける必要があります。

スナップショット

{:#snapshots-definition}になります。

データチームハンドブックの複数の箇所でスナップショットという言葉を使用していますが、この言葉は文脈によっては混乱を招く恐れがあります。辞書で定義されているスナップショットとは、「ある時点での記憶場所やデータファイルの内容を記録したもの」のことです。私たちは、この言葉を使うときは常にこの定義を使うように努めています。

dbt

最も一般的な使い方は、dbtsnapshotsに関するものです。dbt snapshotsが実行されると、ユーザーが指定したクエリに基づいてデータの状態を取得し、データの状態の全履歴を含むテーブルを更新します。このスナップショットには、特定のスナップショットが有効な期間を示す `valid_to` と `valid_from` フィールドがあります。より詳しい技術情報については、dbtガイドの「dbt snapshots」セクションをご覧ください。

dbt snapshotsで生成・管理されるテーブルは、生のヒストリカル・スナップショット・テーブルです。これらの生のヒストリカル・スナップショットの上にダウンストリーム・モデルを構築し、さらにクエリを行うことになります。snapshotsフォルダには、dbtのモデルが格納されています。これは、24時間の間に複数のスナップショットが取得されている場合に便利です。また、生の履歴テーブルから最新のスナップショットを返すモデルも作成します。

その他の用途

グリーンハウスのデータはスナップショットのようなものだと考えられています。グリーンハウスから毎日データベースダンプが送られてきて、それをSnowflakeに読み込みます。もしこれらのテーブルのdbtスナップショットを取り始めたら、グリーンハウスデータのヒストリカル・スナップショットを作成することになります。

一部の yamlファイルや BambooHR で行っている抽出は、スナップショットと考えることもできます。この抽出では、ファイルやテーブルの全体を取得し、それをタイムスタンプ付きの行としてウェアハウスに保存します。つまり、これらのファイルやテーブルの過去のスナップショットがあるということですが、これらはdbtと同じ種類のスナップショットではありません。同じ`valid_to`と`valid_from`の動作を得るためには、追加の変換を行わなければなりません。

言語

- スナップショット - 特定の時点でのデータの状態
- Take a snapshot - 現在のデータの状態を取得して保存するジョブを実行します。dbtのコンテキストで使用できます。yamlの抽出ジョブを参照することはお勧めしません - これらは "run the extract "となります。
- ヒストリカル・スナップショット - 特定のソース・テーブルの複数の時点でのデータを含むテーブル。dbtが生成したスナップショットテーブルを参照するために最もよく使用されます。また、yaml抽出テーブルの参照にも使用できます。
- 最新のスナップショット - 保存されているデータの最新の状態です。dbtのスナップショットでは、`valid_to`がnullのレコードが該当します。BambooHR と yaml の抽出では、抽出ジョブが最後に実行されたときの状態です。Greenhouse rawの場合、これはウェアハウスにあるデータを表しています。Greenhouse データのスナップショットを開始する場合、発言者は raw テーブルを意味しているのか、Historical snapshots テーブルの最新レコードを意味しているのかを明確にする必要があります。

バックアップ

また、堅牢性を高めるために、倉庫のデータをGCS（Google Cloud Storage）にバックアップしています。ジョブの実行にはdbtのrun-operation機能を使用しています。現在、すべてのスナップショットを毎日バックアップし、1ヶ月間保存しています。Calogica社のブログで紹介されている基本的な手順を実行しました。

アドミン

Snowflakeを稼働させるために、管理作業を行います。保存場所の作成

データウェアハウスにデータをロードするためには、通常、データはストレージのバケットから読み出されます。バケットから読み込むためには、そのバケットをSnowflakeの許可リストの一部として追加し、ステージを作成する必要があります。

まず、現在のストレージロケーションをすべて選択します。property_valueでproperty=STORAGE_ALLOWED_LOCATIONSとなっている値をコピーします。

```
desc integration gcs_integration;
```

<<<<_paste_here_>>>> + 新しいバケットの位置の値を上にして、以下のクエリに値を貼り付けます。値は , で区切る必要があります。

```
統合を変更する gcs_integration
SET STORAGE_ALLOWED_LOCATIONS = ('<<<<_paste_here_>>>>')
```

そして、新たなステージでは、収納場所を追加して作成することができます。

```
ステージ "raw "を作成します。"PTO".pto_load
STORAGE_INTEGRATION = GCS_INTEGRATION URL = 'bucket location';
```

トランスフォーメーション

当社では、すべての変換にdbtを使用しています。このツールを使う理由と方法については、dbtガイドをご覧ください。

トラステッドデータフレームワーク

{:#tdf}になります。

データ顧客は、重要な意思決定を行うために、信頼できるデータをデータチームが提供することを期待しています。そしてデータチームは、提供するデータの品質に自信を持つ必要があります。しかし、これは解決が難しい問題です。エンタープライズデータプラットフォームは複雑で、複数の段階でデータの処理や変換が行われ、数十人から数百人の開発者やエンドユーザーが24時間体制でデータの変更や問い合わせを行っています。Trusted Data Framework（TDF）は、技術チームやビジネスチームがアクセス可能な、データ処理段階におけるデータのテストとモニタリングのための標準フレームワークを定義することで、これらの品質と信頼性のニーズをサポートします。TDFは、既存のデータ処理技術から独立したスタンドアローンのモジュールとして実装されており、独立したデータ監視ソリューションのニーズを満たしています。

- アナリストやエンジニアだけでなく、誰もが信頼できるデータに貢献できるようにな
- る データ処理の上から下まで、すべての段階でデータを検証できるようになる ソー
- スシステムのデータパイプラインからデータを検証できるようになる
- 次元モデルへのデータ変換を検証する 重要な企業デー
- タを検証する
- 中央のデータ処理技術から独立して展開可能

主な用語

- アサーションまたはテストケース - 個々のテストであり、実行可能なテストの最小単位である。TDFでは、テストケースはSQLステートメントとして、またはSQLコンパイルツールdbt内のYAMLコンフィギュレーションを介して表現されます。
- データスキーマ - データ対象領域を構成するテーブル、カラム、ビュー、その他の構造要素で、SQLデータ定義言語（DDL）を用いて作成される。
- ゴールデンデータ - ゴールデンデータとは、ビジネスにとって重要な1つのフィールドまたは複数のフィールド
- のグループから得られるデータの定数です。モニタリング - テストケースの結果を追跡することで、データが使用可能な状態であることを確認します。

信頼できるデータコンポーネント

TDFの主な要素は以下の通りです。

1. 新しいデータソリューションから問題解決に至るまで、日々のデータ開発の中で品質を当たり前のように組み込む好循環テストサイクル。
2. テストケースをSQLやYAMLで表現し、誰でも開発できるようにしました。
3. Trusted Data Schemaは、ビジネスプロセスやデータプラットフォームのパフォーマンスに関する知恵を開発するために、テスト結果をモニタリングやアラート、長期的な分析のために保存します。
4. Schema-to-Golden Record Coverage：スキーマから重要な "Golden "データまで、データウェアハウス領域を幅広くカバーします。
5. Trusted Data Dashboard」は、*ビジネスに役立つ*ダッシュボードで、全体のテストカバレッジ、成功、失敗を可視化します。
6. テスト実行とは、テストケースを実行することです。
7. ソースシステムとSnowflakeのVirtuous Test Cycle間の行数を再計算するRow Countテスト

TDFは、ビジネスユーザーを信頼できるデータを確立するための*最も重要な参加者*として受け入れ、シンプルでアクセス可能なテストモデルを使用しています。テストエージェントとしてSQLとYAMLを使用することで、幅広いグループの人々がテストケースを提供することができます。テストフォーマットはシンプルなPASS/FAILの結果と4つのテストケースタイプで構成されています。TDFが価値を示すことで、採用は急速に拡大します。

- データ・カスタマーとビジネス・ユーザーがテスト・フレームワークを学び、自らテストを作成
- する チームは、テストを最後の手段としてではなく、*常に*組み込むべき貴重な活動として受け入れる
- データチームは、プロダクションダウン・レトロスペクティブの一環として新しいテストを追加し、大きな問題になる前に問題をより迅速に特定できるようになりました。
- チームは、継続的に新しいテストを開発し、テストカバレッジを拡大するための運用リズムを開発します。

時間が経てば、何百ものテストケースを開発して毎日のように実行し、データの品質を継続的に検証することも珍しくありません。

SQLとYAMLで表現されるテストケース

SQLはデータベースの世界共通言語であり、データを扱うほとんどの人が、ある程度のSQLの能力を持っています。しかし、すべての人がSQLに精通しているわけではなく、それによって貢献できる人が制限されてしまうことは避けたいと考えています。私たちはdbtを使ってTDFをサポートしており、SQLとYAMLを使ってテストを定義することができます。

信頼性の高いデータスキーマ

すべてのテストはdbtで実行されるので、テスト結果の保存は簡単です。すべてのテストの実行結果をデータウェアハウスに保存します。テスト結果を保存することで、以下のような様々な価値ある機能が可能になります。

- データの可視化とパターン分析 テスト結果（日付ごとの総テスト数、分野ごとの PASS/FAIL 率など） データの対象や
- スキーマに対するテストカバレッジの測定（分野ごとのテ スト数）。
- 時間経過によるシステム品質の向上（PASS率の向上）の測定 テスト結果に基づくアラ
- ートシステムの開発

これらのテスト結果は解析され、Sisenseでの問い合わせに利用できます。

すべてのテスト結果を保存するスキーマは
`workspace_data`」です。注：このスキーマには、ビュ

ーしか含まれていません。

スキーマからゴールデンレコードカバレッジ

データウェアハウスの環境は急速に変化する可能性がありますが、**TDF**はデータウェアハウス内で最も変化する可能性の高い領域をテストでカバーすることで、予測可能性、安定性、品質をサポートします。

1. スキーマの整合性を検証するスキーマテスト
2. 列のデータ値が事前に定義されたしきい値やリテラルに一致するかどうかを判断する列値テスト
3. **Rowcount** は、事前に定義された期間におけるテーブルの行数が、事前に定義されたしきい値またはリテラルに一致するかどうかを判定するテストです。
4. **Golden Data**テストでは、あらかじめ定義された価値の高いデータがテーブル

に存在するかどうかを判断します。 これらのテストの実装の詳細は、**dbt**ガイドに記

載されています。

信頼できるデータダッシュボード

Sisenseの**Trusted Data Dashboard**はこちらをご覧くださ

い。 **Test Run**

まだまだ続きま
す。

行数テスト

行数テストでは、ソース データベースとターゲット データベース間の行数を調整するために、以下のデータを抽出します。ソースの**DB**テーブルを**Snowflake**テーブルにロードし、**Snowflake**から類似の統計情報を抽出してソースとターゲットの比較を行います。ソースとターゲットを完全に一致させるのは難しいです。

- タイミングの違いがあります。
- データウェアハウスでは履歴が残ることが
- あります。削除はソースデータベース上で
  行われます。

シナリオによっては、行数を最高（テーブル）レベルではなく、より低い粒度のレベルでチェックすることをお勧めします。これは、論理的な分布を持つ**1**つまたは複数のフィールドである可能性がありますが、依然として集約されたレベルであると言えます。例えば、テーブルの挿入日や更新日などが挙げられます。

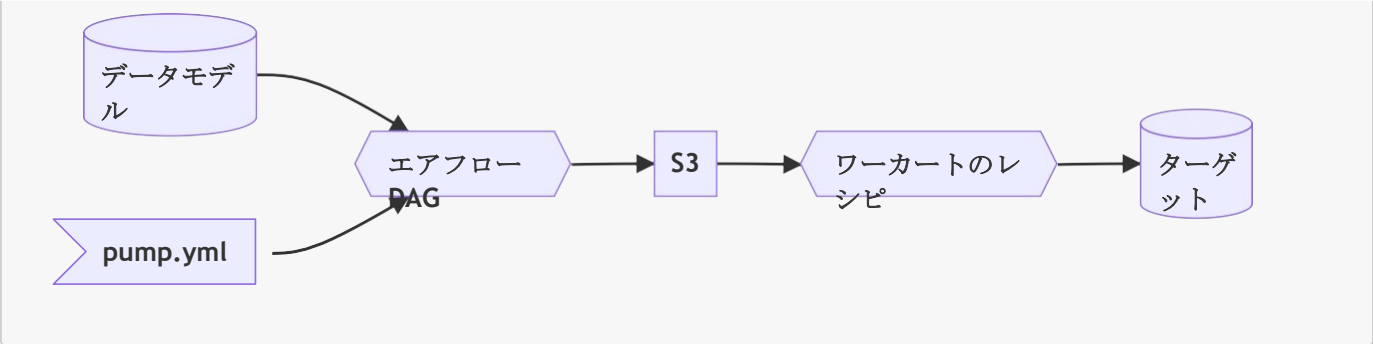
ソースの行数とターゲット（**Snowflake**データウェアハウス）の行数に基づいて、すべての行がデータウェアハウスにロードされているかどうかを判断するために、リコンシリエーションを行うことができます。

行数テスト **PGP**

このフレームワークは、テストを実行するためのあらゆる種類のクエリの実行を扱うように設計されています。現在のアーキテクチャでは、すべてのクエリが**1**つの**Kubernetes**ポッドを作成するため、**1**つのクエリにグループ化することで、**Kubernetes**ポッドの作成数を減らすことができます。**postgres DB**と**snowflake**の間のレコード数とデータの実際のテストでは、ボリュームの少ないソーステーブルをグループ化し、ボリュームの大きいソーステーブルは個別のタスクとして実行するアプローチをとった。

すべてのタイプの調整を行うことを想定した新しい **yaml** ファイルが作成されます（そのため、既存の **yaml** 抽出マニフェストには組み込まれていません）。マニフェストファイルは、ボリュームの少ないテーブルのグループをまとめ、ボリュームの大きいテーブルを個々のタスクとして結合します。**Postgres** と **snowflake** からの行数テストの比較は、**"PROD". "WORKSPACE_DATA". "PGP_SNOWFLAKE_COUNTS"** という名前の **snowflake** テーブルに保存されます。

データポンプ



誰でも簡単にSnowflakeからGitLabテックスタックの他のアプリケーションにデータを送れるようにするために、私たちはEnterprise Applications Integration Engineeringチームと提携して、Data Pumpと呼ぶデータ統合フレームワークを作りました。

これらはすべて、ポンプを実行するData Pump AirflowDAGでオーケストレーションされ、毎日05:00 UTCに1回実行されるように設定されています。

データポンプの追加

ステップ1dbtを使用して、`/marts/pumps`（モデルにREDまたはORANGEデータが含まれる場合は`/marts/pumps_sensitive`）にデータモデルを作成しdbtモデル変更テンプレートを使用してMRを作成しこれがマージされ、Snowflakeで`PROD.PUMPS`または`PROD.PUMPS_SENSITIVE`に表示されると、ステップ2と3の準備が整います。

Step 2: 「Pump Changes」MRテンプレートを使って、`pumps.yml`に以下の属性を持つモデルを追加します。

- model - dbtおよびsnowflakeにおけるモデルの名前です。
- timestamp_column - データのバッチ処理に使用される列の名前（列がなく、テーブルが小さい場合は`null`）。
- sensitive - このモデルが機密データを含み、pumps_sensitiveディレクトリとスキーマオーナーにある場合は
- `true`、あなた（またはビジネスDRI）のgitlabハンドルです。

ステップ3： インテグレーションチームがデータをターゲットアプリケーションにマッピングして統合できるように、「New Data Pump」課題テンプレートを使用して、インテグレーションプロジェクトに統合課題を作成します。

データスピゴット

データスピゴットは、外部システムに制御された方法でSnowflakeのデータへのアクセスを与えるための概念／方法論です。外部システムがSnowflakeにアクセスできるようにするには、以下のようなコントロールが必要です。

- サービス専用のアカウントです。
- 必要最小限のデータのみを表示する専用のビュー（複数可）。個人を特定できる情報（PII）は公開されません。
- 指定されたテーブル/ビューのみにアクセス可能な専用ロール（または同等のも
- の）。コストを制限・監視するための専用のXS倉庫。

新しいData Spigotを設定する手順は以下の通りです。

1. 上記のように、実施されているコントロールに従うこと。
2. 下の表に新しいData Spigotsを追加します。

接続システム	データの範囲	データベースビュー
Grafana	除雪機のロード時間	`prod.legacy.snowplow_page_views_all_grafana_spigot`

ビジュアライゼーション

データの可視化とビジネス・インテリジェンス・ツールとしてSisenseを使用しています。アクセスをご希望の方は、アクセスリクエストを送信してください。

データチームのためのメタアナリシス

- Sisenseの使い方!
- Sisense アカウント最適化
- Sisense アカウントメンテナンス
- dbtイベントロギング
- スノーフレークスペンド❄の

セキュリティ

パスワード

GitLabのパスワードポリシーに基づき、パスワードのみで認証されるサービスアカウントを90日ごとにローテーションしています。パスワードが変更されたシステムと更新された場所の記録は、このGoogleシートに保存されています。

また、Snowflakeユーザーのパスワードは、毎年第3月の第1日曜日（1月、4月、7月、10月）に「SnowflakeパスワードリセットDAG」でローテーションしています。

ソフトウェアのユーザープロビジョニング

データチームは、データチームが管理するツール内のユーザーをプロビジョニングする責任があります。これには、Fivetran、Stitch、Snowflakeなどのツールが含まれます。

Snowflakeについては、このページの「Snowflake Permissions Paradigm」のセクションで説明されている強固なプ

ロセスがあります。他のツールについては、UIを介してユーザーを追加し、適切なGoogleグループが存在する場合

はそれに参加します。

layout: handbook-page-toc title:"Data Team CI Jobs" description:"GitLabデータチームのCIジョブズ"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

このページでは、DataTestsとAnalyticsの両プロジェクトのMerge Requestsでデータチームが使用するCIジョブについて説明します。

パイプラインが故障した場合の対応

- **週末が経過した場合は、以前に実行されたCLONEステップを再実行し、毎週土曜日に古いパイプラインデータベースがSnowFlakeから削除されます。**
- **masterブランチをマージします。dbtがパッケージをどのように処理するかに起因して、パッケージの不具合によりパイプラインが失敗する可能性がありますので、常に最新のブランチで処理する必要があります。**
- **モデル選択の構文を確認します。一般的には、変更するモデルのファイル名をそのまま使うのが最も簡単です。それでも**
- **不明な場合は、@gitlab-data/data-engineers（緊急にサポートが必要な場合はエンジニア）にタグを付けて、問題解決のサポートを依頼してください。**

パイプライン変数がジョブに渡されない

- **現在、GitLab CI パイプラインに問題があります。変数を渡す必要があるパイプラインでは、一度しか変数を渡すことができません。**
- **つまり、dbtモデルのパイプラインを使って2回目のプロセスを実行するテストをしたい場合は、新しいパイプラインを作成する必要があります。**
- **最も簡単な方法は、以下のような青い run pipeline ボタンをクリックすることです。run_pipeline_button.png**

アナリティクス パイプライン

ステージ

CIの仕事はステージごとに分類されています。

❄スノーフレーク

これらのジョブは `.gitlab-ci.yml` で定義され

ています。 **clone_prep_specific_schema**

準備中のデータベースで利用可能な任意のスキーマのクローンが必要な場合、これを実行します。クローンを作成するスキーマを`SCHEMA_NAME`変数を使用します。すでにクローンが存在する場合は、何もしません。

クローン_開発者固有のスキーマ

prodデータベースで利用可能な任意のスキーマのクローンが必要な場合に実行します。クローンを作成するスキーマを`SCHEMA_NAME`変数を使用します。クローンがすでに存在する場合は、何もし

ません。

MRの起動時に自動的に実行され、あらゆる**dbt**ジョブを実行できるようになります。クローンが存在するかどうかを確認するだけなので、このジョブの後続の実行は高速です。これは`prod`と`prep`のデータベースの空のクローンです。

クローン・プロード・リアル

`prod`データベースと`prep`データベースの実際のクローンを作成する必要がある場合に実行します。両方のデータ

ベースの完全なクローンを作成します。 **clone_raw_full**

エクストラクト、フレッシュネス、スナップショットの各ジョブを実行する必要がある場合に実行します。このジョブの後続の実行は、以下の場合にのみ検証されるため、高速になります。
クローンが存在しています。

クローン_ロースト_postgres_pipeline

postgresパイプラインやマニフェストファイルの変更をテストするために、生の`tap_postgres`スキーマのクローンが必要な場合に実行してください。生のクローンが既に存在する場合、これは何もしません。

clone_raw_sheetload

sheetloadへの変更や追加をテストするために、生の`sheetload`スキーマのクローンが必要な場合に実行します。生のクローンがすでに存在する場合は、何もしません。

clone_raw_specific_schema

変更や追加をテストするために、他の **raw** スキーマのクローンが必要な場合に実行します。クローンを作成する **raw** スキーマを `SCHEMA_NAME` 変数で指定します。**raw** クローンがすでに存在する場合は、何もしません。

フォースクローン・ボース

raw、**prod**、**prep**を強制的にリフレッシュしたい場合に実行します。これは、**raw**の完全なクローンを作成しますが、`pref`と`prod`の浅いクローンを作成します。

🏭エキス

これらのジョブは`extract-ci.yml`で定義されています。

ボーンヤード・シートロード

新しい**Boneyard sheetload**のロードをテストしたい場合に実行します。これには本物の`prod`クローンと`prep`クローンが利用できることが必要です。

新しいシートロードをテストしたい場合に実行します。このジョブは、RAWのクローンに対して実行されます。clone_rawジョブが実行されている必要があります。

pgp_test

gitlab-ops DB を除いて、**postgres pipeline** のマニフェストを追加または更新する場合に実行します。**RAW**のクローンと`MANIFEST_NAME`変数、**SCD**テーブルの場合は`TASK_INSTANCE`変数が必要です。

`MANIFEST_NAME` は、マニフェストのファイル名とは異なります。**postfix** `_db_manifest.yaml` を除いたファイル名です

。 **gitlab_ops_pgp_test**

gitlab-opsのマニフェストファイルを更新する場合に実行します。**RAW**のクローンが必要です。変数は一切必要ありません。**SCD**テーブルの場合は、`TASK_INSTANCE`変数を指定することもできます。これは、データベースに接続するために**CloudSQL Proxy**が動作している必要があるため、`pgp_test`ジョブとは別です。

⚙ dbt Run

これらのジョブは`snowflake-dbt-ci.yml`で定義されています。

> **DBT**モデル変更の**MR**の一部として、変更が本番環境で何も壊さないことをテストするためのパイプライン・ジョブをトリガーする必要があります。これらのジョブをトリガーするには、この**MR**の下部にある **"Pipelines "**タブに行き、適切なステージ（**dbt_run**または**dbt_misc**）をクリックします。

これらのジョブは `ci` ターゲットにスコープされています。このターゲットは、**snowplow**および**version**データセットの

データのサブセットを選択します。**dbt**のすべての実行ジョブについて、ジョブの成果物が得られます。これには、コ

ンパイルされたコードと実行結果が含まれます。

これらのジョブは、プライマリ`RAW`データベースに対して実行されます。

ほとんどの**dbt run**ジョブは、テストを必要とする**dbt**モデルを指定する変数でパラメータ化することができます。このイントロでは、変数の設定方法の例をご覧いただけます。

変数 `DBT_MODELS` は、モデル選択の構文に関する**dbt**ドキュメントの任意の例の代用となります。

`data-tests` プロジェクトのテストに対する変更をテストする場合は、手動ジョブにブランチ名とともに `DATA_TEST_BRANCH` を渡すことができます。これにより、`packages.yml` の **data-tests** パッケージのブランチが更新されます。これは`dbt test`を実行しているすべてのジョブで有効です。

また、モデル選択の最後に`--fail-fast`を追加することで、最初の失敗で素早く**dbt**の呼び出しを終了させることができます。詳しくは**dbt**のドキュメントをご覧ください。

➕⤴⤴⤴⤴モデル

どのモデルを実行するかは、変数 `DBT_MODELS` で指定します。

➕🦕スペック_**L**モデル

変数 `DBT_MODELS` で **L** ウェアハウスを使って実行するモデルを指定

➕🐘**specify_xl_model**

XL ウェアハウスを使用して実行するモデルを変数 `DBT_MODELS` で指定する。

変数 `DBT_MODELS` でどのモデルを除外するかを指定します。

変数DBT_MODELSでL倉庫を使って除外するモデルを指定する

ーー

XL ウェアハウスで除外するモデルを変数 `DBT_MODELS` で指定します。

➕ sｐecify_raw_model

RAWデータベースのクローンに対して**dbt**モデルを指定します。このジョブは、`RAW`のクローンに対して実行されます。`clone_raw`が必要です。
ジョブが実行されたことを確認します。この機能は以下のような場合に有効です。

- あなたは新しいシートロードファイルをアップロードしています。これを使って、シートを追加するのと同じ**MR**で、**sheetload dbt**モデルをテストすることができます。
- 新しい**gitlab.com**やその他の**pgp**テーブルを追加しています。これを使って、テーブルを追加するのと同じ**MR**で**dbt**モデルをテストすることができます。
- **dbt**のスナップショットを追加して、そのスナップショットの上に構築されたモデルをテストしたい。

➕ Specify_csv_seed

このジョブでは、特定のシードファイルをテストします。

変数 `DBT_MODELS` で種ファイルを指定します。

➕ sｐecify_snapshot

実行するスナップショットを変数 `DBT_MODELS` で指定します。このジョブは、`RAW` のクローンに対して実行されます。`clone_raw`が必要です。
ジョブを実行したことになります。

➕ specify_l_snapshot

変数 `DBT_MODELS` で実行するスナップショットを指定します。このジョブは、大規模な**SnowFlake**倉庫を使用して、`RAW`のクローンに対して実行します。`clone_raw`ジョブが実行されている必要があります。

dbt Misc

これらのジョブは`snowflake-dbt-ci.yml`で定義されています。

すべてのテスト

すべてのテストを実行する

- 注：テストが含まれているので、**dbt_run**ステージジョブを実行している場合は、このジョブを実行する必要はありません。

データテスト

データテストのみの実行

鮮度

ソースフレッシュネステストを実行します。このジョブは、`RAW`のクローンに対して実行されます。`clone_raw`ジョブが実行されている必要があります。

ペリスコープ_クエリ

このジョブは自動的に実行され、`.sql`ファイルが変更されたときにのみ表示されます。最も単純な形式では、このジョブは、現在変更されているモデルのいずれかが**Periscope**で照会されているかどうかを確認します。照会されている場合、ジョブは関連するダッシュボードを確認するように通知して失敗します。照会されていない場合は、ジョブは成功します。

現在、この仕事の注意点は

- どのダッシュボードをチェックすべきかは教えてくれません。
- 文字列補間構文（例：`retention_[some_variable]`）でクエリされたテーブルは検証できません。 テーブルが**dbt**でエ
-

イリアスされている場合は検証できません。

説明

ここでは、**periscope**のクエリの仕組みを説明します。

```
git clone -b periscope/master -single-branch https://gitlab.com/gitlab-data/periscope.git --depth 1
```

これは、**periscope**プロジェクトのクローンです。

```
git diff origin/$CI_MERGE_REQUEST_TARGET_BRANCH_NAME...HEAD --name-only | grep -iEo "(.*)\.sql"| sed -
E 's/\.sql//' | awk -F '/' '{print tolower($NF)}' | sort | uniq > diff.txt
```

これは、**master**ブランチ(つまり**target**ブランチ)から現在のコミット(**HEAD**)までに変更されたファイルの一覧を取得します。次に、**sql** ファイルだけを見つけて (**grep**) 、`.sql` を空の文字列に置き換えます (**sed**) 。`awk`を使用して、ファイルの各行の最後のカラムの小文字を表示します（**$NF**（フィールド数）で表されます）。フィールドの区切りにはスラッシュ（**/**）を使用します。出力は**directory/directory/filename**であり、ほとんどの**dbt**モデルはファイル名にちなんだテーブルに書き込むと仮定しているので、これは期待通りに動作します。その後、結果をソートし、ユニークなセットを取得して、**diff.txt**というファイルに書き込みます。

```
periscope_check.py
```

periscope リポジトリ全体を再帰的に検索し、現在問い合わせ可能な **3** つのスキーマのうち、 `from|join` 文にマッチする文字列を探します。一致したファイルのクリーニングを行い、参照されているすべてのファイルに対応するテーブル名の辞書を作成します。その後、`diff.txt`を読み込んでルックアップを行い、**comparison.txt**に書き込み、モデル名に基づいてマッチングを行います。

```
if (( $(cat comparison.txt | wc -l | tr -d ' ') > 0 )); then echo "Check these!" && cat comparison.txt
&& exit 1; else echo "All good" && exit 0; fi;
```

ワードカウント（**wc**）を使用して、比較ファイルに何行あるかを確認します。**0**行以上ある場合は、その行を表示し、失敗して終了します。行がない場合は成功して終了します。

safe_model_script

すべての**SAFE**データが適切なスキーマに格納れことを保証するために、**MNPI**データを持つソースモデルの下流にあるすべてのモデル`PROD`で例外タグを持つか、制限付きスキーマになっていなければならない。この**CI**ジョブは、この状態への準拠をチェックします。**MR**がこのジョブに失敗した場合は、監査を受けて**MNPI**データが変更されていないことを確認し、適切な例外タグを追加するか、モデルを適切な制限付きスキーマに移行する必要があります。

スキーマ・テスト

スキーマテストのスナ

ップショットのみを実

行する

スナップショットを実行します。このジョブは、`RAW`のクローンに対して実行します。`clone_raw_full`ジョブが実

行されている必要があります。 **specify_tests**

変数 `DBT_MODELS` で指定されたモデルテストを実行します。

パイソン

これらのジョブは `.gitlab-ci.yml` で定義されています。

`.py`ファイルが変更されたときにのみ表示されるジョブがいくつかあります。すべてのジョブは、`.py`ファイルが存在する新しいコミットごとに自動的に実行されます。それ以外のジョブは実行できません。他のジョブは

パーミフロストの取扱説明書

Permifrostのドライランを行うマニュアル作業。

YAML_VALIDATION

`permissions/snowflake/roles.yml`に変更があったときにトリガーされます。**YAML** が正しくフォーマットされているかどうかを検証します。

🛑 スノーフレークストップ

これらのジョブは `.gitlab-ci.yml` で定義され

ています。 **clone_stop**

MRがマージまたはクローズされたときに自動的に実行されます。手動では実行しないでください。

データテストパイプライン

以下はすべて、レポで提供されている変更点を使って、**Prod DB**に対して実行します。以下の実行にはクローニングは必要ありません。

all_tests_prod

analytics & data tests repoにあるすべてのテストを実行します。

data_tests_prod

アナリティクス&データテストのレポにあるすべてのデータテストを

実行します。

analytics & data tests repoにあるすべてのスキーマテストを実行します。

テストプログラムの指定

変数 `DBT_MODELS` で指定されたモデルテストを実行します。

---

layout: handbook-page-toc title:"dbt Guide" 説明"data build tool (dbt) Guide" このページでは

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

クイックリンク

PrimaryProject{:.btn .btn-purple-inv} dbtdocs{:.btn .btn-purple-inv}。

何を、なぜ

dbtは、**DataBuildTool**の略で、データウェアハウス内のデータ変換を管理するための**オープンソースプロジェクト**です。**dbt**は、データウェアハウス内のデータ変換を管理するためのオープンソースプロジェクトです。データがウェアハウスにロードされると、チームは分析に必要なすべてのデータ変換を管理することができます。また、**dbt**にはテストとドキュメントが組み込まれているので、生成・分析するテーブルに高い信頼性を持たせることができます。

以下のリンクは、**dbt**とは何かについての優れた概要を示しています。

- **とは何でしょか？-** これは、ツールを理解するための技術的ではない概要です。
- **dbtとは**より技術的な内容で、ドキュメントからの抜粋です。

しかし、なぜ私たちは**dbt**を使うのでしょうか？理由はいくつかあります。

第一に、オープンソースのツールであり、活発なコミュニティが存在することです。オープンソースのツールを選択することで、より大きなデータコミュニティとのコラボレーションが可能になり、独自のソリューションを採用した場合よりも早く問題を解決することができます。

2つ目は、バージョン管理を念頭に置いて作られていること。GitLabの場合、会社の構築や運営に製品を使用しているので、これは必須です。

第三に、アナリストの言語であるSQLを話すことができます。SQLは多くの人の仕事に欠かせないものになってきているので、貢献できる人の数も増えています。

最後に、テストとドキュメントを最初から統合することで、チームがより速く動けるようになります

。dbtの基本についての詳細は、データアナリストのオンボーディング問題テンプレートをご覧くだ

さい。

時には、データ変換のためにdbtのパッケージを利用することがあります。パッケージ管理はdbtに組み込まれています。利用可能なパッケージの全リストはdbtHubサイトにあります。

dbtの実行

dbtに興味をお持ちの方は、dbtのドキュメントJaffle Shopという架空の企業のデータを扱うためのセットアップに関する素晴らしいチュートリアルがあります。

dbtを使用してデータチームプロジェクトに貢献したい場合は、Snowflakeインスタンスへのアクセスを取得する必要があります。

構成

- Snowflakeインスタンスへのアクセスを確保します。
- Makeがインストールされていることを確認する（新しいMacやXCodeではインスト
- ールされているはず） ホームディレクトリに`.dbt`というフォルダを作成する
- `~/.dbt/`フォルダの中には、以下のサンプルプロファイルのような`profiles.`ymlfileがあるはずです。
- 可能な限り小さい倉庫を環境変数として保存する必要があります。当社のdbtジョブでは、倉庫を識別するための変数名として`SNOWFLAKE_TRANSFORM_WAREHOUSE`を使用しています。この環境変数は、`.bashrc`または`.zshrc`ファイルで以下のように設定できます。
  - `export SNOWFLAKE_TRANSFORM_WAREHOUSE="ANALYST_XS"`
  - より多くの計算が必要な場合は、dbtコマンドに`--vars '{warehouse_name: analyst_xl}'`を追加することで、変数を上書きすることができます。
- アナリティクス・プロジェ
- クトのクローンを作成しま
  す。
  - Dockerがインストールされていることを確認する

なお、これらのステップの多くは、新しいアナリストが実行することを推奨するオンボーディング・スクリプトで行われます。

Venvワークフロー

{:#Venv-workflow}

Macをお使いの方にお勧めのワークフローです。dbtの使い

方

- `DBT_PROFILE_PATH`環境変数が設定されていることを確認します。これは、onboarding_script.zshを使用している場合は設定されているはずですが（最新で定期的に更新されているので、これを使用することをお勧めします）、そうでない場合は、あなたの
  `.bashrc`または`.zshrc`に`export DBT_PROFILE_PATH="/<your_root_dir/.dbt/"`を追加するか、ローカルのターミナルセッションで同じコマンドを実行してください。
- SSH の設定が GitLab の指示通りになっていることを確認します。鍵は `~/.ssh/` にあり、鍵はパスワードなしで生成されているはずです。

    - また、メインプロジェクトの`dbt deps`を実行するには、このプロジェクトへのアクセスが必要になります。
- 注意：デフォルトのブラウザがクロームに設定されていることを確認してください。内蔵の**SSO**ログインはクロームでのみ動作します。
- 注意：初めて**dbt**を実行する前に、`make-prepare-dbt`を実行してください。これにより、**venv**がインスト
- ールされていることが確認できます。`dbt`コンテナを起動し、その中でシェルからコマンドを実行するには、`make run-dbt`を使用します。

- これにより、ローカルの`profiles.yml`や**repo**ファイルを含む、`dbt`の実行に必要なすべてのものが自動的にインポートされます。
- 現在のブランチのドキュメントを見るには、`make run-dbt-docs` を実行してから、ウェブブラウザーで `localhost:8081` にアクセスしてください。このためには、`profiles.yml` で `docs` プロファイルが設定されている必要があることに注意してください。

Dockerのワークフロー

{:#docker-workflow}になります。

venvのワークフローは前提条件が少なく、かなり高速であるため、主に**Linux**を使用しているユーザーには以下のようなワークフローを推奨します。

`dbt`とその依存関係をローカルで処理する際の複雑さを排除するために、メインの分析プロジェクトでは、`Docker`コンテナ内での`dbt`の使用をサポートしています。このコンテナは`data-image`プロジェクトから構築します。`Makefile`の中には、この作業を容易にするためのコマンドが用意されています。様々な`make`コマンドとその役割について質問がある場合は、`make help`を使ってコマンドのリストとそれぞれの役割を確認してください。

初回実行時（およびコンテナが更新されたとき）には、以下のコマンドを実行してください。

1. `make update-containers`
2. メイク・クリーンアップ

これらのコマンドは、最新バージョンのコンテナを確実に取得し、ローカルの`Docker`をクリーンアップします。
の環境を整え
ます。

dbtの使用

- `DBT_PROFILE_PATH`環境変数が設定されていることを確認します。これは、**onboarding_script.zsh**（定期的に更新される最新のものを使用することをお勧めします）または**onboarding_script.sh**を使用している場合は設定されているはずですが、そうでない場合は、`.bashrc`または`.zshrc`に`export DBT_PROFILE_PATH="/<your_root_dir/.dbt/"`を追加するか、ローカルのターミナルセッションで同じコマンドを実行することで設定できます。
- **SSH** の設定が **GitLab** の指示通りになっていることを確認します。鍵は `~/.ssh/` にあり、鍵はパスワードなしで生成されているはずです。

また、メインプロジェクトの`dbt deps`を実行するには、このプロジェクトへのアクセスが必要になります。
`db`コンテナを起動して、その中でシェルからコマンドを実行するには、`make dbt-image`を使用します。

- これにより、ローカルの`profiles.yml`や**repo**ファイルなど、`dbt`の実行に必要なものがすべて自動的にインポートさ
  - れます。 見つからない変数（`GIT_BRANCH`、`KUBECONFIG`、`GOOGLE_APPLICATION_CREDENTIALS`など）に関する警告が表示されることがあります。**Airflow**での開発でない限り、これは問題なく、予想されることです。
- 現在のブランチのドキュメントを見るには、`make dbt-docs` を実行してから、ウェブブラウザーで `localhost:8081` にアクセスしてください。このためには、`profiles.yml` で `docs` プロファイルが設定されている必要があることに注意してください。
- `dbt`コンテナに入ったら、通常通りに`dbt`コマンドを実行します。
- レポ内の任意のファイルに加えられた変更は、コンテナ内で自動的に更新されます。エディタでファイルを変更しても、コンテナを再起動する必要はありません。

コマンドライン・チートシート

これは、プライマリコマンドリファレンスの簡易版で、**dbt**固有

のものです。

- `dbt clean - /dbt_modules`（**deps**の実行時に生成される）と`/target`フォルダ（モデルの実行時に生成される）を削除します。

- `dbt run` **- regular run**

- モデル選択の構文（出典）。モデルを指定することで、関連性があると思われるモデルのみを実行/テストすることができ、時間を大幅に節約することができます。しかし、上流の重要な依存関係を指定し忘れるリスクがあるので、構文をし

っかり理解しておくと良いでしょう。

- `dbt run --models modelname` - **modelname**のみを実行します。
- `dbt run --models +modelname` - **modelname**とすべての親を実行します。

    - `dbt run --models modelname+` - **modelname**とすべての**children**を実行します。
    - `dbt run --models +modelname+` - モデル名と、その親と子のすべてを実行します。
    - `dbt run --models @modelname` - **modelname**、すべての親、すべての子、およびすべての子のすべての親を実行します。
    - `dbt run --exclude modelname` - **modelname**以外のすべてのモデルを実行
    - します。 なお、これらはすべてフォルダ選択の構文でも動作します。
        - `dbt run --models folder` - フォルダ内のすべてのモデルを実行します。
        - `dbt run --models folder.subfolder` - サブフォルダ内のすべてのモデルを実行します。
        - `dbt run --models +folder.subfolder` - サブフォルダ内のすべてのモデルと、すべての親モデルを実行します。

- `dbt run --full-refresh` - インクリメンタルモデルを更新します。
- `dbt test` - カスタムデータテストおよびスキーマテストを実行します。
- `dbt seed` - `data-paths`ディレクトリで指定された**csv**ファイルをデータウェアハウスにロードします。このガイドの**seed**のセクションも参照してください
- `dbt compile` - すべてのモデルをコンパイルします。このコマンドは定期的に実行する必要はありません。モデルを実行すると、**dbt**はモデルをコンパイルします。

  オンボーディング・スクリプトを実行した場合のみ動作します。

- `dbt_run_changed` - 変更されたモデルのみを実行するためにあなたのコンピュータに追加された関数です（これは**docker**コンテナ内からアクセスできます）。
- `cycle_logs` - **dbt**のログを消去するためにあなたのコンピュータに追加した機能です（**docker**コンテナ内からはアクセスできません）。
- `make dbt-docs` - `dbt`のドキュメントを**Web**ブラウザで表示するためのローカルコンテナを起動するコマンドです。
  ローカルホスト:8081

dbtプロジェクトへの貢献のための構成

dbtへの貢献に興味をお持ちの方のために、お勧めのローカル環境の設定方法をご紹介します。

- **GitHub**の**UI**を使って**dbt**プロジェクトを個人の名前空間にフォーク
- する プロジェクトをローカルにクローンする
- 以下のコマンドで**dbt**の仮想環境（**venv**）を作成します。

```
cd〜。
mkdir .venv # これはあなたのルート"~"ディレクトリにあるは
ずです python -m venv .venv/dbt
ソース ~/.venv/dbt/bin/activate
pip install dbt
```

- 仮想環境の起動を容易にするために、`.bashrc`または`.zshrc`に`dbt!="source ~/.venv/dbt/bin/activate "`というエイリアスを追加することを検討してください。
- 同じターミナルウィンドウで**dbt**プロジェクトに移動します。コマンドプロンプトの最初に`(dbt)`と表示されているはずです。
- `pip install -r editable_requirements.txt`を実行します。これにより、**venv**でローカルに**dbt**を実行したときに、あなたのマシン上のコードを使用していることが保証されます。
- どの`dbt`を実行して**venv**を指しているか確認する
- ローカルでコードを開発し、通常通りに変更をコミットし、**GitHub**の自分の名前空間にプッシュする

PRのためにコードを提出する準備ができたら、相手の**CLA**に署名していることを確認しましょう。

スタイル&ユースガイド

モデル構造

よりKimballスタイルのウェアハウスへの移行に伴い、ウェアハウスやプロジェクト構造でのモデルの整理方法を改善しています。以下のセクションはすべて、dbtのデフォルトであるmodelsディレクトリの下のトップレベルのディレクトリになります。この構造は、Fishtown Analyticsのプロジェクト構造を参考にしています。

Legacy Structure {: .panel-heading} {: .panel-heading}。

Kimballの次元モデリングに注目する前は、CorrとStagnittoの「Agile Data Warehouse Design」で紹介されているBEAM*アプローチのモデリングからインスピレーションを得ていました。既存のモデルの多くは、今でもそのパターンを踏襲しています。このセクションの情報は、ハンドブックの過去のイテレーションからのものです。

- 最終的な）_xf dbtモデルのゴールは、BEAM*テーブルでなければなりません。つまり、ビジネスイベントの分析とモデルの構造に沿って、ビジネスを測定するための、誰が、何を、どこで、いつ、何人で、なぜ、どのように、という質問の組み合わせに答えるものです。
- ベースモデル- ソーステーブルを参照する唯一のdbtモデル。ベースモデルは最小限の変換ロジックを持つ（通常、データインテグリティに問題のある行や、分析対象外のフラグが立っている行をフィルタリングしたり、分析しやすいようにカラム名を変更する程度）。
- エンドユーザーモデル - 解析に使用されるdbtモデル。モデルの最終バージョンは、BEAM*テーブルになることを目的としている場合、_xfという接尾辞が付けられます。このモデルは、ビジネスイベント分析とモデルの構造に従い、ビジネスを測定するために、誰が、何を、どこで、いつ、いくつ、なぜ、どのように、という質問の組み合わせに答えなければなりません。エンドユーザーモデルは、レガシースキーマにあります。

FY21-Q4モデルの移行 {: .panel-heading}。

FY21-Q4では、アナリティクス・データベースを置き換えるために、prodおよびprepデータベースを導入しました。この2つの新しいデータベースは、アナリティクス・データベースを完全に置き換えるものです。

ローカル開発も、カスタムスキーマからカスタムデータベースに変更。ソース

すべての生データは、SnowflakeのRAWデータベースに残っています。これらのRAWテーブルは、ソーステーブルまたはRAWテーブルと呼ばれています。これらは通常、元のデータソースを示すスキーマに保存されます。

ソースはdbtのsources.ymlファイルで定義します。

- dbtソースでデータベースを参照するために変数を使用し、Snowflakeクローンでの変更をテストする場合、プログラムで参照を設定できるようにしています。
- 通常の慣習に合わない名前や意味が不明瞭な名前のソーステーブルを扱う場合、元の名前が混乱していたり混乱している場合は、識別子を使ってソーステーブルの名前を上書きします。(識別子の使用に関するドキュメント)

```
# 良い
テーブ
ル
  - 名前: bizible_attribution_touchpoint
    識別子: bizible2 bizible_attribution_touchpoint c

# 悪い
テーブ
ル
  - name: bizible2 bizible_attribution_touchpoint c
```

ソースモデル

すべての生データの上に、非常に薄いソースレイヤーを強制的に配置しています。このディレクトリには、ソース固有の変換の大半が保存されます。これらは、生データから直接取り出し、ファクトやディメンションを作成するために必要な準備作業を行う「ベース」モデルであり、以下のこと*だけ*を行う必要があります。

- フィールド名をユーザーフレンドリーな名
- 前に変更 カラムを適切なタイプにキャスト
- 当面の間、**100%**使えることが保証されている最小限の変換です。例えば、データが乱れていることが知られているフィールドから**Salesforce ID**を解析することがその例です。
- 論理的に命名されたスキーマへの配置

基礎となる生データが完璧にキャストされ、名前が付けられている場合でも、フォーマットを強制するソースモデルが存在すべきです。これは、エンドユーザーの利便性を高めるためで、見る場所が**1**つで済みますし、この完璧なデータが機密性の高いものである場合には、パーミッションがより明確になります。

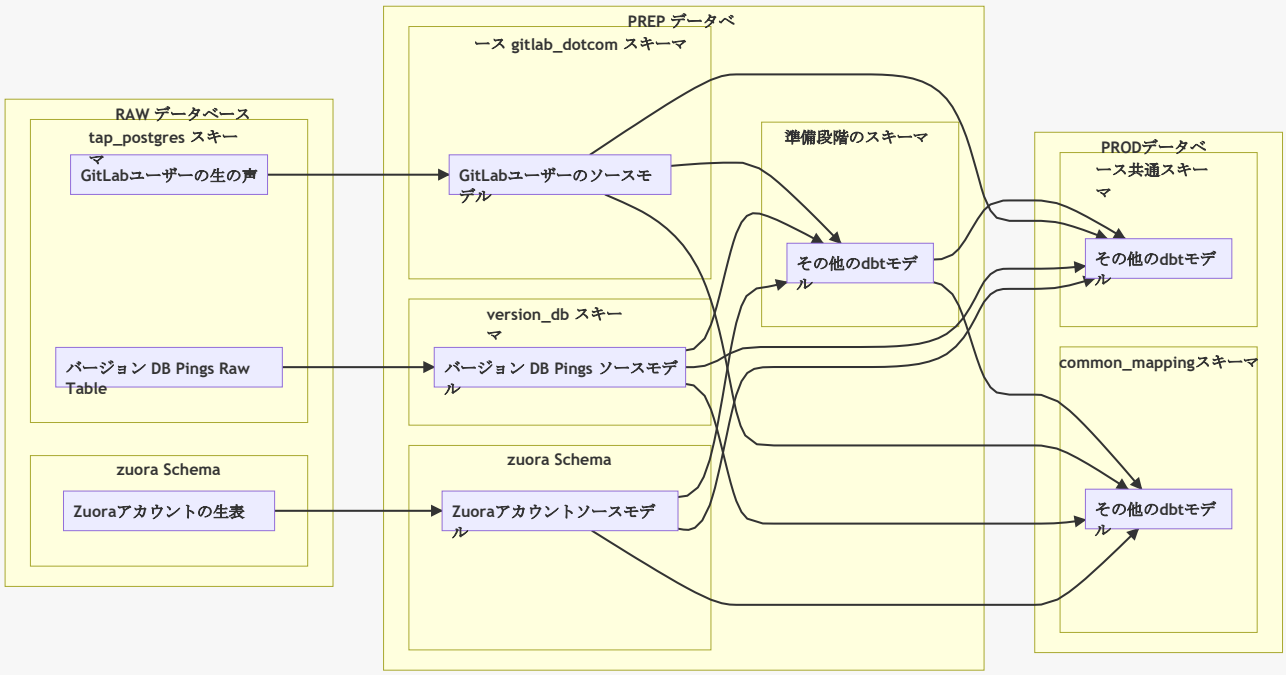
ソースモデルでは以下のようなことをしてはいけません。

- データの削除 他のテー
- ブルへの結合
- カラムの意味を根本的に変える変換

大多数のユーザーにとって、ソースモデルは「生」のデータであると考えられます。覚えておくべき重要なポイント

- これらのモデルは、データソースの種類に基づいて、準備データベースの論理的に名付けられたスキーマに書き込まれます。例えば、以下のようになります。
  - `raw.zuora` に保存されている **Zuora** データは、`prep.zuora` にあるソースモデルを持ちます。
  - `raw.tap_postgres.gitlab_db_*`に格納されたテーブルを持つ**GitLab.com**のデータは、ソースモデルが `prep.gitlab_dotcom`
  - テーブルが `raw.tap_postgres.customers_db_*` に格納されている **Customers.gitlab.com** データは、ソースモデルが `prep.customers_db` に格納されています。
- これらのモデルは、ソースごとに整理されるべきです。これは通常、生のデータベースのスキーマに
- 対応します。
- ソースモデルのみ、ソース/ローテーブルから選択する必要がある
- ソースモデルは、生のデータベースから直接選択すべきではありません。代わりに、以下のように**jinja**でソースを参照する必要があります。
  `FROM {{ source('bamboohr', 'job_info') }}`。
- ソースモデルは、`/models/sources/<data_source/>`ディレクトリに配置してください。
- ソースモデルは、必要なすべてのデータタイプのキャストを行い、キャストする際には `::` 構文を使用します（より少ない文字数で同じことができ、よりすっきりとした印象を与えます）。
  - 理想的には、ソースモデルはすべてのカラムをキャストする必要があります。暗黙的ではなく明示的な方が良い。仮定のテスト
- ソースモデルでは、すべてのフィールドの命名を行い、フィールド名を標準的なフィールド命名規則に適合させる必要
- があります。 予約語を使用しているソースフィールドは、ソースモデルで名前を変更する必要があります。
- 特に大きなデータのソースモデルは、常に論理フィールド（通常は関連するタイムスタンプ）の**ORDER BY**ステートメントで終わるべきです。これは基本的にウェアハウスのクラスタキーを定義し、**Snowflake**のマイクロパーティショニングを活用するのに役立ちます。

ソースモデルが生のテーブルとどのように関連しているか、また、ソースモデルが下流のすべてのモデリングのためのクリーンなレイヤーとしてどのように機能するかを視覚的に示すために、次の図を参照してください。



機密データ

場合によっては、生の値を公開してはいけない列があります。これには、お客様のEメールやお名前などが含まれます。しかし、このようなデータを必要とする正当な理由があり、以下のような方法でデータを保護しながらも、正当に知る必要のある人には利用できるようにしています。

与えられたモデルでは、ソースのフォーマットは上記の通りです。ソースモデルにはカラムのハッシュ化はありません。これは、セキュリティやアクセスの観点から、生データと同じように扱う必要があります。

機密性の高いカラムは、`schema.yml`ファイルでメタキーを使用し、`sensitive`を`true`に設定することで文書化されます。例は以下の通りです。

```
- 名前: sfdc_contact_source
  の説明を参照してください。SFDC Contactsカラ
  ムのソースモデル。
        - name: contact_id
          tests:
              - not_null
              - ユニーク
        - name: contact_email
          meta:
              センシティブ: true
        - name: contact_name
          meta:
              センシティブ：true
```

そして、ソースモデルから2つの独立したモデル、つまり感度の高いモデルと低いモデルが作成されます。

non-sensitiveモデルでは、hash_sensitive_columnsというdbtマクロを使用します。これはソーステーブルを受け取り、metaフィールドにsensitive: trueを持つすべてのカラムをハッシュ化します。すべてのカラムが同じようにハッシュ化されるため、特定のジョインキーは指定されません。必要であれば、マクロの外で別のカラムをジョインキーとしてこのモデルに追加することができます。sfdc_contactモデルは、この良い例です。2つのカラムがハッシュ化されていますが、contact_idという追加の主キーが指定されています。

センシティブモデルでは、dbt マクロ nohash_sensitive_columns を使用してジョインキーを作成します。このマクロは、ソーステーブルとジョインキーとしてのカラムを受け取り、ジョインキーとしてハッシュ化されたカラムと、ハッシュ化されていない残りのカラムを返します。sfdc_contact_piiモデルは、このマクロが使われている良い例です。

すべてのハッシュにはソルトも含まれています。これらは環境変数で指定します。データの種類によって、さまざまなソルトがあります。これらは get_saltマクロで定義され、ローカル開発用の dbt コンテナを使用する際にも設定されます。

一般的に、チーム メンバーは、Snowflake UI 内のクエリ文字列で使用されるソルトを見ることを許可されるべきではありません。テーブルモデルでは、この目標はSnowflakeの組み込みENCRYPT関数を使用することで満たされます。ビューに実体化されたモデルでは、ENCRYPT 関数は機能しないようです。代わりに、セキュアなビューを使用した回避策が利用されます。セキュアビューは、DDL の表示を所有者のみに制限し、ハッシュの可視性を制限します。セキュアなビューを作成するには、モデル構成で secure を true に設定します。説明したようにハッシュ機能を利用したビューであっても、セキュアなビューとして構成されていないものは、おそらく問い合わせができません。

ステージング

Kimballモデリングを導入する前は、ほとんどのモデルがStagingカテゴリーに分類されていました。

このディレクトリには、ビジネスに特化した変換の大半が保存されます。この層のモデリングは、ソースモデルの作成よりもかなり複雑で、ビジネスの分析ニーズに合わせて高度に調整されたモデルとなります。これには

- 無関係なレコードのフィルタリング
- アナリティクスに必要なカラムの選択
- 抽象的なビジネスコンセプトを表現するためのカラム名の変更 他
- のテーブルへの結合
- ビジネスロジックの実行
- Kimballの方法論に従ったfct_*およびdim_*テーブルのモデリング

Dimensional Modeling

{:次元}

ファクトテーブルとディメンショナルテーブルは、ソースデータから抽象化され、ビジネスに関連するエンティティとプロセスを表します。なぜディメンショナル・モデリングを使用するのかという情報は、ハンドブックのEDWの部分に記載されています。

モデリング開発プロセス

1. ディメンション・バス・マトリックスをEDMの青写真として拡張する。
2. 適切なLucidChart ERDにテーブルを追加します。
3. ディメンション・テーブルを展開します。各ディメンションには、未知のものを表す-1キー・バリューの共通レコード・エントリも含まれています。
4. ファクト・テーブルを作成します。正しいディメンジョン・キーをファクトに入力し、不明なキーには-1のキー値を使用します。

命名基準

セルフサービス型のデータ環境を構築する際には、命名規則をはじめとする意図的な取り組みが重要です。目標は、初心者、中級者、上級者がデータウェアハウスを簡単に操作できるようにすることです。以下のベストプラクティスに従うことで、これを可能にします。

1. **FACT TABLES:** `fct_<verb>` **事実は、出来事や現実世界で起きているプロセスを表しています。ファクトは、アクションや「動詞」を表しているため、しばしば識別することができます。(例：セッション、トランザクション)**

2. **DIMENSION TABLES**：dim_<noun>＝ディメンション・テーブル。ディメンションは、ファクト・レコードに記述的なコンテキストを提供します。ディメンションは、人、場所、物などの「名詞」であるため、しばしば識別することができます（例：顧客、従業員）。ディメンション属性は「形容詞」として機能します。(例：顧客タイプ、従業員部門)
3. 例えば、**dim_customers**ではなく**dim_customer**のように、単数形のネーミングを使用する必要があります。
4. 同じようなデータをグループ化するには、テーブル名やカラム名にプレフィックスを使用します。例えば、**dim_geo_location**、**dim_geo_region**、**dim_geo_sub_region**のように、アルファベット順にソートしてもデータは論理的にグループ化されます。
5. 主キーおよび外部キーの命名には、ディメンションテーブル名を使用します。これにより、追加の属性を取り込むためにどのテーブルを結合する必要があるかが、ユーザーに明確になります。たとえば、**dim_crm_account**のプライマリ・キーは**dim_crm_account_id**です。このフィールドが**fct_subscription**にある場合、**dim_crm_account_id**という名前になり、追加のアカウント詳細を得るために**dim_crm_account**に結合する必要があることが明確になります。

モデリング要件

- 一般的には、まず共通スキーマにディメンジョンを作成し、次にファクト・テーブルを構築するコードがこれらの共通ディメンジョンを参照して（**LEFT JOIN**）、共通のキーが利用できるようにします。論理的には、ファクト・テーブルにあるすべてのキーがディメンジョンで利用可能であると**100%**確信できる状況もあります。これは、ディメンジョンのソースと適用されるロジックに依存します。
- ファクトとディメンションの両方を、**TDF（Trusted Data Framework）**テストを用いてテストする必要があります。
    - ディメンションについては、**-1**（未知）の**dimension_id**の存在や、総行数をテストすることができます。
    - ファクトについては、不適切な粒度の結合によってレコード数/行数が拡大していないかどうかをテストし、関連するファクト・レコードのディメンション・テーブルからディメンション属性を引き出して、ゴールデン・データ・レコードの期待値と比較するゴールデン・データ・テストを追加することができます。
- **fct_**および**dim_**モデルは、クエリのパフォーマンスを向上させるためにテーブルとして
- 実体化する必要があります。
    - ハッシュ化されたサロゲートキーです。
    - 自然なユニークキーです。このキーの値は、複数のカラムから構成され、一意性を生み出すことができます。
- モデルはテストされ、モデルと同じディレクトリにある**schema.yml**ファイルで文書化されま
- す。 すべてのファクトとディメンションには、以下の監査カラムがあります。
    - **revision_number - モデルの論理的な変更を表す、手動で増やされた番号 created_by - GitLab** のユー
    - ザー **ID**。
    - **updated_by - これはGitLabのユーザーIDです。**
    - **model_created_at timestamp - これは、モデルが作成されたときの静的な値です**
    - **model_updated_at timestamp - これは、モデルが誰かによって更新された最後の時間で**
    - す **dbt_created_at timestamp - これは、テーブルが作成されたときにdbtによって入力さ**
      れます
    - **dbt_updated_at タイムスタンプ - データが最後に読み込まれた日付です。ほとんどのモデルでは dbt_created_at** と同じになりますが、インクリメンタルモデルでは例外です。
- ディメンションテーブルをサポートする**Prep(prep_)**と**Mapping/look-up(map_)**テーブルは**common_mapping**に作成する必要があります。
  スキーマです。
- 追加の**Bridge(bdg_)**テーブルは、共通のスキーマに存在する必要があります。これらのテーブルは、**2**つのテーブル間の多対多の関係を解決するための中間テーブルとして機能します。

ERDの要件

- **Lucidchart**で生成されます。
- すべての関連モデルの**dbt**ドキュメントに**iframe**として埋め込ま
- れる **ERD**から与えられたモデルの**dbt**ドキュメントへのクロスリ
- ンク 適切な関係の接続
- 主キーと外部キーの一覧
- テーブルの性質を示すもので、変更される可能性が低い他のカラムを少なくとも**3**〜**5**個 ワーキ
- ング**SQL**の参照例

追加ガイドライン

次元モデルは、使いやすく、ユーザーのために設計されていることを意味します。次元モデルは、ソース・モデルとは逆に非正規化されている可能性が高く、読みやすく、解釈しやすいだけでなく、結合数を減らすことで効率的なクエリを可能にします。

- **CTE**が**2**つのモデルに重複している場合、別の**fct_/dim_**テーブルになることがよくあります。

マルツ

Martモデルは、ビジネスエンティティとプロセスを記述します。マーケティング、財務、製品などのビジネスユニットごとにまとめられていることが多い。

モデルがこのディレクトリにあると、データがきれいにモデリングされていて、クエリの準備ができていることをビジネス関係者に伝えることができます。

ファクトテーブルおよびディメンションテーブルの命名規則に従い、すべてのマートは `mart_` という接頭辞

で始まります。マートは、他のマートの上に構築してはいけません。マートは、**FCT**および**DIM**テーブルの上

に構築する必要があります。

ワークスペース

dbtプロジェクトでは、チーム独自のコードを書くためのスペースを提供しています。これは、すべてのコーディングガイドやスタイルガイドを遵守する必要のない、チーム独自のコードです。これは、チームがより強固である必要のないソリューションを使って、より早く反復できるようにするためのものです。

プロジェクト内には`/models/workspaces/`フォルダがあり、チームはそこに`workspace_<team>`というスタイルのフォルダを作成してコードを保存できます。このコードはデータチームによるスタイルのレビューを受けません。マージされる前に懸念されるのは、コードが動作するかどうかと、本番稼動に影響を与えるようなひどい非効率性がコードにないかどうかだけです。

新しいスペースを追加するには

- 分析プロジェクトで課題を作成し、新しいマージリクエストを開く
- `/models/workspaces/`に新しいフォルダを作成する 例：

  `workspace_security`
- `dbt_project.yml`ファイルに、新しいワークスペースのエントリを追加します。書き込むべきスキーマを含めます。

```
# ------------------
# ワークスペース
# ------------------
ワークスペースです。
  +tags:["ワークスペース"]

  workspace_security:# これは `/models/workspaces/` にあるフォルダにマッピングされます。
    +schema: workspace_security # これは `prod` データベースのスキーマです。
```

- フォルダに`.sql`ファイルを追加します。
- **CODEOWNERS**ファイルに任意のエントリを追加
- **dbtWorkspaceChanges MR**テンプレートを使用し、その指示に従って**MR**を提出し、レビューと最終的なマージを行います。

新たに追加されたコードがデータウェアハウスに表示されるまでには、最大で**24**時間かかります。

データチームは、本番の**dbt**実行を大幅に遅らせるようなコードを拒否する権利を持っています。もしこのようなことがあれば、ワークスペースのためだけに別の**dbt**実行ジョブを構築することを検討します。

一般

- モデル名はできるだけわかりやすく、可能な限り完全な単語を使用してください（例：`accounts`の代わりに`accts`.
- 新しいデータモデルのドキュメント化とテストは、データモデル作成のプロセスの一部です。新しい**dbt**モデルは、テス

トとドキュメントなしでは完成しません。

- 分析タイプ、データソース（複数の場合はアルファ順）、モノ、集計の命名規則に従う。

```
-- Good
retention_sfdc_zuora_customer_count.sql

悪いリテンション
.sql
```

- すべての {{ ref('...') }} ステートメントは、ファイルの先頭にある **CTE** の中に置く必要があります。**(**これらは**import**文と考えてください。**)**
    - これは、{{ ref('...') }} を持つすべての **CTE** が SELECT * のみであるべきだということではありません。モデルにとって意味のあることであれば、ref を持つ **CTE** で追加の操作を行っても構いません。
- 複雑な**SQL**を別のモデルに分けたい場合は、**DRY**で理解しやすくするために絶対にそうすべきです。materalized='ephemeral'という設定は、基本的にモデルを**CTE**のように扱う**1**つのオプションです。

モデル構成

モデルの構成定義を提供する方法は複数あります。**dbtdocsforconfiguringmodels**には、モデルの設定方法についての簡潔な説明があります。

モデルを構成するためのガイドラインです。

- デフォルトのマテリアライゼーションはビュー
- デフォルトのスキーマはprep.preparationです。
- モデルを無効にするには、dbt_project.ymlの+enabled: false宣言で行う必要があります。 設定は最小の場所に適
- 用する必要があります。
    - ディレクトリ内の**50%**以下のモデルが同じ設定を必要とする場合、個々のモデルを設定します。
    - ディレクトリ内のモデルの**50%**以上が同じ設定を必要とする場合、以下のようなデフォルトの設定を強く検討してください。
      dbt_project.ymlを使用していますが、この設定がディレクトリ内の新しいモデルのデフォルトとして適切かどうかを考えてください。

による。

通常の使い方では、**dbt**は{{ ref('...') }}構文の使い方に基づいて、すべてのモデルを実行する適切な順番を知っています。しかし、モデルをいつ実行すべきかを**dbt**が知らない場合があります。具体的な例としては、schema_union_allやschema_union_limitマクロを使用した場合です。この場合、コンパイル時に明示的な参照が行われていないため、**dbt**はモデルが先に実行されると考えてしまいます。この問題に対処するために、ファイルの中で設定の後に、どのモデルに依存しているかを示すコメントを追加することができます。

````
```sql
 {{config({
     "materialized": "view"
   })
 }}

 --に依存します。{{ ref('s snowplow_sessions') }}。

 {{ schema_union_all('snowplow_', 'snowplow_sessions') }}。
```
````

dbtはrefを見て、指定されたモデルの後にこのモデルを構築します。デ

ータベースとスキーマの名前の生成

dbtでは、カスタムのデータベース名やスキーマ名を生成することができます。これは、私たちのプロジェクトでは、以下の場所を制御するために広く使用されています。
モデルは実体化されており、生産や開発のユースケースに応じて変化します。

データベース

デフォルトの動作は、**dbt**のドキュメントの「データベースの使用の項に記載されています。というマクロがあります。`generate_database_name`は、書き込み先のスキーマを決定します。

このマクロの動作を独自の `generate_database_name`定義でオーバーライドします。このマクロは **profiles.yml** で提供された設定 **(**ターゲット名とスキーマ**)** と **model config** で提供されたスキーマ設定を取り込んで、最終的なスキーマが何であるべきかを決定します。

スキーマ

デフォルトの動作は、**dbt**ドキュメントの「**Usingcustomschemas**」セクションに記載されています。というマクロがあります。`generate_schema_name`は、書き込み先のスキーマを決定します。

このマクロの動作を独自の `generate_schema_name`定義でオーバーライドします。このマクロは **profiles.yml** で提供された設定 **(**ターゲット名とスキーマ**)** と **model config** で提供されたスキーマ設定を受け取り、最終的なスキーマが何であるべきかを決定します。

開発行動

FY21-Q4では、スキーマではなく開発用のデータベースを持つように変更しました。**dbt**のユーザーは、`TMURPHY_PROD`や`TMURPHY_PREP`のように、モデルが書き込まれるスクラッチデータベースを定義する必要があります。

このスイッチは、`profies.yml`ファイルで定義されたターゲット名によって制御されます。ローカル開発では、決して`prod`や`ci`をターゲット
にしています。

マクロ

命名規則

- ファイル名がマクロ名と一致していること

構造

- マクロの説明は、**macros.yml**または**macros.md**のいずれかのファイルに記述してください。
-

dbt-utils

私たちの**dbt**プロジェクトでは、**dbt-utils**パッケージを使用しています。これにより、一般的に便利なマクロがいくつか追加されています。重要なものを挙げてみましょう。

- **group_by** - このマクロは、フィールド**1...N**に対して**Group By**ステートメントを構築します。
- **star** - このマクロは、**except** 引数にリストされている列を除いて、テーブルからすべての列を取り出します。
- **surrogate_key** - このマクロは、フィールド名のリストを受け取り、値のハッシュを返して一意のキーを生成します。

種子

{:#seeds}になります。

シードは、**csv**ファイルからデータウェアハウス（**dbt**ドキュメント）にデータをロードする方法です。これらの**csv**ファイルは**dbt**のリポジトリに置かれているため、バージョン管理されており、コードレビューが可能です。この方法は、頻繁に変更されない静的なデータを読み込むのに適しています。`dbt seed`コマンドで使用する**csv**ファイルは、長さが最大**1k**行で数キロバイト以下のものが良いでしょう。

コラムの整理

ベースモデルを作成する際には、列に何らかの論理的な順序付けをする必要があります。私たちは、以下の**4**つの基本的なグルー

- プ分けを推奨します。プライマリデータ
- 外部キー

- 論理データ - このグループは必要に応じてさらに細分化できる
- メタデータ

プライマリデータは、テーブルを説明する重要な情報です。主キーは、名前などの他の関連する固有の属性とともに、このグループに含まれるべきです。

外部キーとは、別のテーブルを指すすべての列のことです。

論理データは、参照しているオブジェクトを説明する追加のデータディメンションです。**Salesforce**のオポチュニティの場合は、オポチュニティの所有者や契約額などです。意味のあるものであれば、さらに論理的なグループ化が推奨されます。たとえば、契約額のすべてのバリエーションをグループ化することは意味があります。

任意のグループ内では、カラムはエイリアス名に基づいてアルファベット順に表示されます。

グループ化の推奨事項の例外は、定義されたマニフェストファイルによって抽出を制御する場合です。この完璧な例が **gitlab.com** のマニフェストで、アプリケーションデータベースからどのカラムを抽出するかを定義しています。これらのテーブルのベースモデルは、マニフェストと同じ順序で並べることができます。ファイル間の差分を比較して正確性を確保するのが簡単だからです。

- グループ内では、エイリアスのアルファベット順に並びます。

```
--良い

SELECT
  ID名                  AS account_id,
                        AS account_name,

  外部キー
  オーナーID            AS owner_id,
  PID                   AS parent_account_id,
  ジッド                AS zuora_id,

  ロジカルインフォ
  opportunity_owner c AS opportunity_owner,
  account_owner cAS opportunity_owner_manager,
  owner_team_o cAS opportunity_owner_team,

  メタデータ
  isdeleted ラスタ      AS is_deleted,
  クティヴィティデー    AS last_activity_date
  ト
FROMテーブル
```

- グループを介さず、エイリアスのアルファベット順に並べる

```
Less Good

SELECT
                        idAS account_id,
                        nameASアカウント名。
  isdeletedAS is_deleted, lastactivitydateAS
  last_activity_date, opportunity_owner c AS
  opportunity_owner, account_owner cAS
  opportunity_owner_manager, owner_team_o cAS
  opportunity_owner_team, owneridAS owner_id,
  pidAS parent_account_id,
  zidAS zuora_id
FROMテーブル
```

- オリジナルの名前でアルファベット順に並んでいます。

```
バッド

セレク

ト
  account_owner cAS opportunity_owner_manager,
                        idAS account_id,
  isdeletedAS is_deleted, lastactivitydateAS
  last_activity_date     nameAS account_name,
  opportunity_owner c AS opportunity_owner,
  owner_team_o cAS opportunity_owner_team,
  owneridAS owner_id,
  pidAS parent_account_id,
  zidAS zuora_id
FROMテーブル
```

タグ

dbtのタグは、プロジェクトの様々な部分にラベルを付ける方法です。これらのタグは、実行するモデル、スナップショット、シードのセットを選択する際に利用できます。

タグは**YAML**ファイルやモデルのコンフィグ設定で追加することができます。`dbt_project.yml`ファイルには、タグの使用例がいくつかあります。**Trusted Data Framework**のタグを追加する具体的な例を以下に示します。

分析やデータテストのプロジェクトでは、すべてのタグについて**Single Source of Truth**（真実の情報源）を徹底しています。どのタグが使用されているかを文書化するために、**ValidTagsCSV**を使用しています。マージリクエストでは、すべての**dbt CI**ジョブの中に検証ステップがあり、プロジェクトで使用されているすべてのタグに対してこの**CSV**をチェックし、不一致があった場合はジョブを失敗させます。将来的には、この**csv**ファイルにタグに関するより多くのメタデータを含めることを目指しています。

どのレベルでも適用されるタグは、どのテストにも適用されないことに注意してください。テスト用のタグは、`schema.yml` ファイル内でテストごとに明示的に適用する必要があります。

トラステッドデータフレームワーク

Trusted Data Frameworkの背景にある哲学については、**Platform**ページの**Trusted Data Framework**セクションを参照してください。

スキーマからゴールデンデータカバレッジ

TDF（**Trusted Data Framework**）テストを**5**つのカテゴリーに分けて実施しています。

1. スキーマの整合性を検証するスキーマテスト
2. 列のデータ値が事前に定義されたしきい値やリテラルに一致するかどうかを判断する列値テスト
3. **Rowcount** は、事前に定義された期間におけるテーブルの行数が、事前に定義されたしきい値またはリテラルに一致するかどうかを判定するテストです。
4. あらかじめ定義された価値の高いデータがテーブルに存在するかどうかを判断する**Golden Data**テスト
5. **Custom SQL**は、上記のカテゴリーに適合しない有効な**SQL**をテストします。

テストは主に**2**つの場所に保存されます。メインプロジェクト内の**YAML**ファイルか、**DataTests**プロジェクト内に保存されます。

スキーマとカラム値のテストは、通常、メインプロジェクトにあります。これらは、モデルと同じディレクトリにある`schema.yml`と`sources.yml`ファイルに格納されます。

Rowcount、**Golden Data**、その他のカスタム **SQL** テストは、常に **DataTests** プロジェクトにあります。これは、**GitLab** 内部でのみ使用されるプライベートプロジェクトです。

タギング

テストのタグ付けは、新しいテストを追加する際の重要なステップです。**dbt**タグでテストをラベル付けすることは、信頼できるデータダッシュボードを構築する際に、テストを解析して識別する方法です。テストをタグ付けする方法は、その種類によって**2**通りあります。

一つ目は、YAML 定義にタグを追加することです。これは、ソーステストの場合は YAML 定義の最上位レベルで、モデルテストの場合はカラムレベルで行います。

```
## Source Labeling in
sources.yml version: 2

のソースになります。
  - 名前: ズーラ
    のタグが表示されます。["tdf", "zuora"] です。

## Model Labeing in
schema.yml version: 2

モデルを使用しています。
  - 名前: zuora_accounting_period_source
    の説明を参照してください。Zuora会計期間のソースレイヤーで、列のクリーニングや名前の変
    更を行います。
      - name: accounting_period_id
        tags:["tdf", "zuora"] テ
        スト。
          - not_null
          - ユニーク
```

これらの例では、下の階層でネストされたすべてのテストにタグが適用されます。タグを

追加するもうひとつの方法は、テストファイルの先頭にある `config` 宣言を使用することです

。

```
{{ config({
    "tags "を使用します。["tdf", "zuora"] です。
    })
}}

WITH test AS (...)
```

一般的なガイダンス

- すべてのモデルは`schema.yml`ファイルでテストされなければなりません。
- 最低限、ユニーク・フィールド、**not nullable** フィールド、外部キー制約をテストする必要がある（該当する場合
- ） **dbt** テストの出力を **MR** に貼り付ける必要がある
- テストが失敗した場合は、レビューを依頼する前に修正するか説明する必

要があります。 スキーマテスト

スキーマテストは、既知のテーブル、カラム、およびその他のスキーマ構造の存在を検証するように設計されています。スキーマテストは、計画的または偶発的なスキーマの変更を特定するのに役立ちます。

すべてのスキーマテストは、**PASS** または **FAIL** ステータスになります。

スキーマテストの例

目的：このテストでは、重要なテーブルが**Zuora Data Pipeline**に存在するかどうかを検証します。

スキーマテストを**dbtのマクロ**として実装しました。つまり、ユーザーは**SQL**を書く代わりに、マクロを呼び出すだけでテストを追加することができます。これは `raw_table_existence` マクロによって制御されます。

```
-- ファイル: https://gitlab.com/gitlab-data/analytics/-/blob/master/transform/snowflake-
dbt/tests/sources/zuora/existence/zuora_raw_source_table_existence.sql
```

```
{{ config({
    "tags "を使用します。["tdf", "zuora"] です。
    })
}}

{{ raw_table_existence(
    'zuora_stitch',
    ['account', 'subscription', 'rateplancharge'] です。
)}}
```

カラムの値のテスト

列値テストは、列のデータ値があらかじめ定義されたしきい値内にあるかどうか、または既知のリテラルと一致するかどうかを判定します。列値テストは、幅広い用途に使用できるため、**TDF**テストの中で最も一般的なタイプです。列値テストは、以下のような場面で役立ちます。

- 変更管理：リリース前とリリース後のテスト
- 重要な過去のデータの合計/集計が、過去に報告された結果と一致していること 既知
- の「承認済み」データが常に存在すること

dbtには、カラムが**NULL**でないこと、ユニークな値を持つこと、特定の値のみを含むこと、カラムのすべての値が他のモデルで表現されていること（参照整合性）を保証するテストがあります。

また、**dbt-utils**パッケージを使用して、さらに多くのテスト機能を追加

しています。すべての列値テストは、**PASS**または**FAIL**のステータスにな

ります。

コラムバリューテスト例1

目的: このテストは、**Zuora** のアカウント **ID** フィールドを検証します。このフィールドは常に**32**文字で、数字と小文字しか持ちません。

dbtを使用しているので、すべてのソーステーブルと下流のモデル化されたデータのほとんどについてドキュメントがあります。**yaml**ドキュメントファイルを使って、個々のカラムにテストを追加することができます。

```
# ファイル: https://gitlab.com/gitlab-data/analytics/-/blob/master/transform/snowflake-
dbt/models/sources/zuora/sources.yml
    - 名前：アカウント
    description: '{{ doc("zuora_account_source") }}'
    columns:
        - name: id
        説明アカウントの主キー タグ("tdf",
        "zuora")
        のテストを行います。
            - dbt_utils.expression_is_true:
                expression:"id REGEXP '[0-9a-z]{32}'"
```

行数テスト

行数テストは、列値テストの特殊なタイプであり、その重要性と実用性のために分類されています。行数テストは、一定期間におけるテーブルの行数が、あらかじめ定義された期待値を満たしているかどうかを判定します。正当な理由でデータ量が急激に変化する場合は、行数テストを適切に更新する必要があります。

行数テスト例1

目的：このテストでは、**2019**年に作成された**18,849**件の**Zuora**購読レコードが常にあることを検証します。

このテストは**dbt**のマクロとして実装されています。つまり、**SQL**を書く代わりに、ユーザーはマクロを呼び出すだけでテスト

を追加することができます。これは、`source_rowcount` マクロによって制御されます。

```
-- https://gitlab.com/gitlab-data/data-
test/-blob/main/tests/sources/zuora/rowcount/zuora_subscription_source_rowcount_2019.sql
{{ config({
    "tags "を使用します。["tdf", "zuora"] です。
    })
}}

{{ source_rowcount(
    'zuora',
    'subscription',
    18489,
    "autorenew = 'TRUE' and createddate > '2019-01-01' and createddate < '2020-01-01'"
)}}
```

Rowcountテスト例2

目的：私たちは急成長しているビジネスを行っており、前日からロードされた新規購読が常に**50**件以上、最大で**200**件必要です。これは、`model_new_reocrds_per_day`マクロで制御されます。

```
-- https://gitlab.com/gitlab-data/data-
tests/-blob/main/tests/sources/zuora/rowcount/zuora_subscription_source_model_new_records_per_day.s
ql
{{ config({
    "tags":["tdf", "zuora"],
    "severity":"warn",
    })
}}

{{ model_new_rows_per_day(
    'zuora_subscription_source',
    'created_date',
    50,
    200,
    "date_trunc('day',created_date) >= '2020-01-01'"
)}}
```

ゴールデンデータテスト

トップ顧客の記録、**2019**年の全世界のユーザー数、**100**万契約を突破したときの**KPI**の結果など、データウェアハウスに常に存在し、決して変更してはならないほど重要なデータもあります。このようなケースの中には、新しいデータベース機能を開発することで解決できるものもありますが、これは複雑で、既存のデータ処理のワークフローとは必ずしも一致しない可能性があります。さらに、データ変換に誤ってバグが追加されたり、重要な本番テーブルに対して誤って悪い**UPDATE**を実行してしまうこともあります。ゴールデンデータテストは、列値テストの特殊なタイプで、既知のデータリテラルの存在を検証し、これらの問題が発生したときにキャッチするのに役立ちます。

Golden Dataテストは**CSV**を使って実装されています。プライバシーを守り、ユーザーが必要なデータをエンコードできるようにするため、**Golden Data**の**CSV**を**DataTests**プロジェクトの`/data`フォルダに保存しています。これらのファイルは**dbt**パッケージとして本番環境にインポートされ、`prep.tdf`スキーマでアップロードされます。

ユーザーは、**Golden Data Macros**を使用して比較を実行するテストを作成できます。

ゴールデンデータのテスト例

目的：**ACME**は当社の最も重要な顧客です。このテストでは、**ACME**が常に**DIM_ACCOUNT**テーブルに入っていることを検証します。これは、`model_golden_data_comparison`マクロによって制御されます。

```
-- dim_account_golden_data
account_name, account_status, account_currency, is_deleted, crm_id
ACME, Active, USD, FALSE, 0016100001BrzkTQZY
```

```
{{ config({
    "tags":["tdf", "dim_account"]
    })
}}

{{ model_golden_data_comparison('dim_account') }}。
```

同様に、この同じデータが**RAW**データベースのソーステーブルに存在する場合は、以下のようなフォーマットになります。これは、`source_golden_data_comparison`マクロで制御されます。

```
-- sfdc_account_raw_golden_data
name, status, currency, deleted,
id
ACME, アクティブ, USD, FALSE, 0016100001BrzkTQZY
```

```
{{ config({
    "tags":["tdf", "sfdc"] です。
    })
}}

{{ source_golden_records_comparison('sfdc','account') }}。
```

カスタムSQL

上記のカテゴリーに当てはまらないテストを考えているかもしれません。また、任意の**SQL**をテストとして書くこともできます。留意すべき点は、行が返されなければテストは*合格ということ*です。もし、クエリから何らかの行が返された場合、テストは失敗します。

dbtのドキュメントに掲載されている例です。

```
{{ config({
    "tags":["tdf", "fct_payments"]
    })
}}

-- 返金にはマイナスの金額があるため、合計金額は常に >= 0 である必要があります。
そのため、テストを失敗させるために、これが真実ではないレコードを返します。
    order_id,
    sum(amount) AS
total_amount FROM {{
ref('fct_payments' )}}。GROUP
BY 1
HAVING total_amount < 0
```

任意の有効な**SQL**を記述し、任意の**dbt**モデルやソーステーブルを参照することができます

。マージ要求ワークフロー

テストを追加・更新する際には、いくつかのシナリオが考えられます。

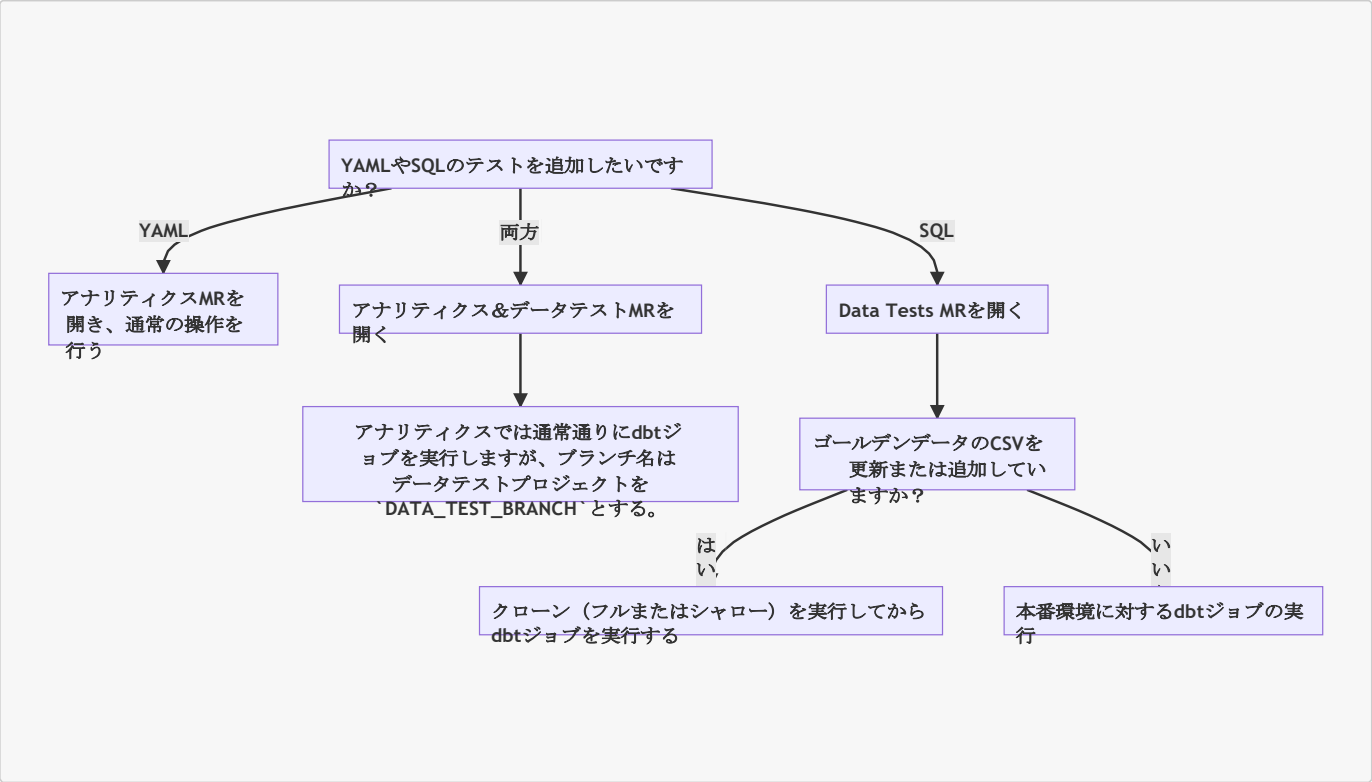
最初のシナリオは、メインプロジェクト内のYAMLファイルのテストを修正または追加することです。これは標準的なMRのワークフローに沿ったもので、何も変わりません。通常通りCIジョブを実行します。

2つ目のシナリオは、data-testsプロジェクト内のテストまたはゴールデンデータのレコードを、analyticsプロジェクト内のMRを介して更新または追加されているテーブルに対して追加することです。これは最も一般的なシナリオです。この場合、data-testsプロジェクトのMRでパイプラインを実行する必要はありません。分析用MRの通常のdbtパイプラインは実行でき、唯一の変更点は、data-testsプロジェクトのブランチ名を環境変数DATA_TEST_BRANCHを介してジョブに渡す必要があることです。

3つ目のシナリオは、data-testsプロジェクトにテストが追加されているが、ゴールデンデータのCSVファイルが更新または追加されておらず、analyticsプロジェクトに対応するMRが存在しない場合です。このシナリオでは、本番データに対してテストを実行するCIジョブがいくつか表示されます。これは、マイナーな変更（構文やタグなど）が機能することを確認するのに便利です。

4つ目のシナリオは、ゴールデンデータのCSVファイルが追加・更新されているのに、対応するアナリティクスMRがない場合です。この場合、ゴールデンデータファイルがテーブルとしてデータベースに挿入されるため、本番環境とのテストは行いたくありません。このシナリオでは、追加のCIジョブが表示されます。倉庫のクローンを作成するもの、Analyticsプロジェクトに格納されているdbtモデルをクローンに対して実行するもの、クローンに対してテストを実行するものがあります。

このフローチャートは、何をすべきかの大まかな目安となります。より詳細な手順については、プロジェクト内の関連するMRテンプレートの指示に従ってください。



data-testsにマージ要求が、analyticsにもマージある場合、analyticsMRはdata-testsMRのdependencyとして設定べきですつまり、data-testsのMRがマージされる前に、analyticsのMRがマージされなければなりません。

信頼できるデータダッシュボード

Trusted Data Dashboardは、データウェアハウスの健全性を迅速に評価するために使用されます。最も重要なデータは、テストのPASSまたはFAILステータスを明確に色分けした内容で、ビジネスに適したシンプルな方法で表示されます。

スナップショット

{:#snapshots}になります。

スナップショットは、ソーステーブルのポイントインタイムコピーを取る方法です。 dbtには、スナップショットがどのように機能するかについての優れたドキュメントがあります。スナップショットについての説明や定義に注意してください。

`dbt snapshot`**によるスナップショットテーブルの作成**

スナップショットの定義は、dbtプロジェクトのsnapshotsフォルダに保存されます。発見しやすいように、異なるスナップショットをデータソースごとに整理しました。

以下は、スナップショットの実装例です。

```
{% snapshot sfdc_opportunity_snapshots %}。

    {{
        config(
          unique_key='id',
          strategy='timestamp',
          updated_at='systemmodstamp',
        )
    }}

    SELECT *
```

```
    FROM {{ source('salesforce', 'opportunity') }}。

{% endsnapshot %}。
```

注目すべきキーアイテム

- データベースとスキーマはdbt_project.ymlで設定します。データベースは環境変数で、スキーマはスナップショットに設定されています。
- *常に*ソーステーブルから選択する。重複排除が必要な場合でも、下流のdbtモデルからの選択は失敗しやすいため、ソーステーブルから選択する必要がある
- スナップショットはRAWで保存されているため、スキーマやテーブルへのアクセスをロールに明示的に付与
- する必要がある ファイル名は{ソース名}_{ソース_テーブル名}_snapshotsという命名規則に従うこと 重複排
- 除作業以外のスナップショットの変換は一切行わないことデータは常に下流側でクリーンにする
- 信頼できるupdated_atフィールドがない場合を除いて、（チェックよりも）戦略としてtimestampを使用することをお勧めします。戦略についてのドキュメントはこちらをご覧ください。

テストのスナップショット

スナップショットのテストは、CIジョブのspecify_snapshotを使ってマージリクエストで行うことができます。エンジニアは **Airflow** を使用してローカルでテストする必要があります。これは、適切な環境変数が **git** ブランチに基づいて処理されるためです。テストは **master** ブランチで行ってはいけません。SNOWFLAKE_SNAPSHOT_DATABASE環境変数を設定してローカルでテストすることは推奨されません。これは本番データを上書きしてしまうので、RAWに設定してはいけません。

スナップショットとGDPR

GDPRを理由に、データチームが**Snowflake**データウェアハウスから個人データを削除するリクエストを受けることがあります。このような削除に対応するために、dbtマクロを使用します。マクロは、削除が必要な該当データをすべてスキャンします。これはスナップショットテーブルにも適用されます。

フレーバーは**2**種類。

1. **GDPR**の削除要求は、すべての**GitLab**ソースに適用されるため、データウェアハウスのすべてのテーブルをチェックして更新する必要があります。マクロです。
2. **GDPR**の削除要求は、**GitLab.com**に関連するソースのみに適用されるため、データウェアハウス内の**Gitlab.com**に関連するテーブルのみをチェックして更新する必要があります。マクロ

2つ目の例では、新しいスナップショットモデルを作成するときや、既存のスナップショットモデルの名前を変更するときに、dbtマクロが対象となるモデルをカバーしているかどうかを確認します。**Gitlab.com**に関連する**GDPR**削除要求があった場合に、マクロのフィルタリングが該当するスナップショットテーブルに適用されるかどうかを確認します。

スナップショット・テーブルを**prod**データベースで利用可能にする

スナップショットが撮影されると、それはデータソースとなり、またデータソースとして扱われるべきです。

私たちは現在、スナップショットに基づいてモデルを生成する従来の方法を採用しています。つまり、ソースモデルはありません。スナップショットのベースモデルは、**dbt**プロジェクトの**/models/snapshots**フォルダにあります。注意すべき重要な項目

- スナップショットのベースモデルを書く前に、**sources.yml**ファイルに追加することを忘れないでください（このファイルはソートしておいてください）。
- データウェアハウスのテーブル名は、当社のデータウェアハウス設計ガイドラインと一致している必要があります。理想的には{source_name}_{source_table_name}_snapshotsを命名規則としたいと考えています。しかし、**dbt**はプロジェクト内での重複したファイル名を認めていません。スナップショットとスナップショットのベースモデルが同じ名前になることを避けるために、このパターンを採用しています。
    - ベースモデルファイルの名前は、ソースのスナップショットテーブルの名前に_baseというサフィックスを付けたものになります。例: gitlab_dotcom_members_snapshotsというスナップショットファイルがあり、このスナップショットのベースモデルをgitlab_dotcom_members_snapshots_baseとします。
    - **dbtconfig**の**alias**引数を使用して、_baseサフィックスを削除してテーブル名を変更し、テーブル名をきれいに保ちます。

- スナップショットされたソーステーブルに基づいて構築されたベースモデルが存在する場合は、既に記述されているクエリを再利用し、以下の修正を行ってください。
  - 重複排除プロセスを削除すると、必要ありません。
  - dbt_scd_idをスナップショットのベースモデルにプライマリキーとして追加し、名前をもっと明確なものに変更してください（スナップショットのメタフィールドに関するドキュメントはこちら）。
  - dbt_valid_fromとdbt_valid_toのカラムをクエリに追加します。
  - スナップショットのベースモデルとソースモデルの良い例です。 スナッ

プショットの上にモデルを構築する

場合によっては、時間的制約のある変更記録ではなく、1日あたりの記録が必要になることもあります。dbt_valid_from と dbt_valid_to です。この場合、日付スパイニングと呼ばれる技術を使って、毎日のスナップショットを持つモデルを作成することができます。

日付スパイニングでは、スナップショット・モデルが dbt_valid_from と dbt_valid_to に基づいて日付テーブルに結合されます。ここでは、zuora_subscription_snapshotsのソース・モデルがdbt_valid_fromおよびdbt_valid_toに基づいてdim_datesに結合されています。この結合により、指定された日 (snapshot_date と呼ばれる) にアクティブだったサブスクリプションのバージョンを持つ、1 日あたりのサブスクリプションごとの 1 つのレコードが得られます。

日々の記録を生成するもう一つの方法は、dbtユーティリティー関数date_spineを使用することです。私たちは現在、日付の詳細なソースモデルを生成するためにこの関数を使用しています。

また、便利なマクロcreate_snapshot_baseがあり、date_spineを利用して、任意のスナップショットテーブルから日次レコードを持つモデルを生成します。例えば、sfdc_opportunity_snapshots_baseモデルの実装を見てみましょう。

スナップショット上のインクリメンタルモデル

日ごとにレコードを生成するために日付スパイニングを使用している場合は、モデルをインクリメンタルとして実体化することを検討してください。この方法では、snapshot_dateの条件に基づいて新しいレコードのみが追加されます。mart_arr_snapshotsモデルの実装例を見てみましょう。

テストのダウンストリームへの影響

dbtモデルに加えられた変更によるダウンストリームへの影響を手動でテストする方法として、sisense_check.pyがtransform dbtプロジェクトに含まれています。このスクリプトはPeriscopeとSisenseのセーフリポジトリをチェックし、指定されたdbtモデルを参照しているビュー、スニペット、ダッシュボード、チャートを探します。このスクリプトの出力を使って、個々のビュー、スニペット、チャートを手動で評価し、カラムレベルの影響を調べることができます。

チェックするモデルのリストを提供するには、dbtlistコマンドを関連する条件で使用して、必要なモデルのセットを出力し、to_check.txtというファイルにリストをエクスポートします。

```
dbt list -m sfdc_account_source+ netsuite_transaction_lines_source+ --exclude date_details dim_date
> to_check.txt
```

このスクリプトは、PeriscopeとSisense Safeのリポジトリが Analyticsのリポジトリと同じ親ディレクトリにチェックアウトされていることを前提としています。リポジトリがチェックアウトされていない場合、スクリプトは結果を返しません。また、リポジトリが最新でない場合、結果が不完全になる可能性があります。スクリプトは入れ子になった参照をチェックします。テーブルがスニペットで参照され、それがチャートで参照されると、チェックの出力に入れ子になった参照が追加されます。ただし、このスクリプトは、次のスキーマの直接のデータベース参照のみをチェックします。

- レガシー ボンヤ
- ード コモン* リ
- ストリクト_セー
- フ* ワークスペ
- ース*

スクリプトを実行すると、チェック用に提供されたモデルを参照しているすべてのビュー、スニペット、ダッシュボード、チャートの名前を含むJSON、sisense_elements.jsonファイルが出力されます。

```
"mart_arr":{
        "periscope-snippets":[
            "base_pricing_customer_overview.sql",
            "mrarr_base.sql",
            "nf_ptb_account_features.sql"
        ],
        "periscope-dashboards":{
            "Investor Relations":[
                「ライセンスユーザー -- Saas/Self-
                managed」、「トータルARPUの内訳」。
                "ARRバケットによる顧客（親顧客）カウント"
            ]
        },
        "sisense-safe-dashboards":{
            "TD: ARR per Licensed User (ARPU)" を参照してください。[
                "Customer ARR by Product, Sales Segment, Account Owner
                Team", "Total ARPU Breakdown",
                "ARPU。ARPU: 顧客セグメント別自己管理」、
                「ARPU: 顧客セグメント別その他」。顧客セグ
                メント別のその他
                "ARPU:合計
            ],
        }
}
```

出力に含まれるビューやスニペットは角括弧[]で囲まれ、チェックしたモデルと同じレベルで表示されます。

```
    "[nf_ptb_account_features]" を
        参照してください。{ "sisense-
        safe-dashboards":{
            "NF - WIP Sandbox "です。[
                "ARR per Periods - Extra data "です。
                "SMB - 期間別ARR - エクストラデータ",
                "当期間 - PTB For Prediction"
            ]
        }
    }
```

---

layout: handbook-page-toc title:"Enterprise Dimensional Model" このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

背景

EDM（Enterprise Dimensional Model）は、GitLabの一元化されたデータモデルで、レポーティングやアナリティクスにおいて最高レベルの精度と品質を実現し、サポートするために設計されています。このデータモデルは、バスマトリックスやエンティティリレーションシップダイアグラムなど、キンボール技法に基づいています。次元モデリングは、データ開発アプローチの第3段階（要件定義、UIワイヤーフレーム作成に続く）であり、この全体的なアプローチにより、高品質なデータソリューションを繰り返し生み出すことができます。EDMはSnowflake EnterpriseDataWarehouseに格納されており、dbtを使用して生成されています。

2021年4月現在、EDMはGo-To-Marketのファネル分析を解決し、製品の使用状況分析を解決するために積極的に拡張されています。EDMを利用したSiSenseのダッシュボードの例は以下の通りです。

- **TD：セールスファネル**
- **TD: Customer Segmentation**
- **TD: Drillable Net Retention**
- **TD: Pricing Dashoard**

一次元モデリングの成果物

- **EnterpriseBusMatrix**は、すべての**Fact**と**Dimension**のテーブルを使いやすいテーブルに統合したもので、**Kimball**のバスマトリックスを模しています。
- エンタープライズ・エンティティ・リレーションシップ・ダイアグラムは、ファクトテーブルとディメンションテー
- ブルのエンティティレベルの統一ビューを示します。**DimensionalModellingDevelopmentProcess**（モデリング開発プロセス）は、命名規則を含む当社のモデリング標準をカバーしています。

エンタープライズ次元モデルの「BIG PICTURE」図

- **Lucidchart**社の**ER**図テンプレートを使用して、**EnterpriseEntityRelationshipDiagram**のソースを構築します。

ディメンショナル・モデリングとは？

次元モデリングは、RalphKimballによって開発されたBusiness Dimensional Lifecycle方法論の一部であり、データウェアハウスの設計に使用する一連のメソッド、テクニック、コンセプトを含んでいます。

高性能なアクセスを可能にする、標準的で直感的なフレームワークでデータを表示することを目指す論理的設計手法

Dimensional Modelingは、ビジネスプロセス指向で、4つのステップで構築することができます。

1. **ビジネスプロセスの選択 例：月次収益の把握**
2. **穀物を宣言する 例：お客様ごとに**
3. **ディメンションの確認**
4. **事実を確認する**

ファクトテーブルとディメンションテーブル

ディメンションモデリングでは、常にファクト（測定値）とディメンション（コンテキスト）の概念を使用します。ファクトは一般的に（必ずしもそうではないが）集約可能な数値であり、ディメンションはファクトを定義する階層と記述子のグループである。

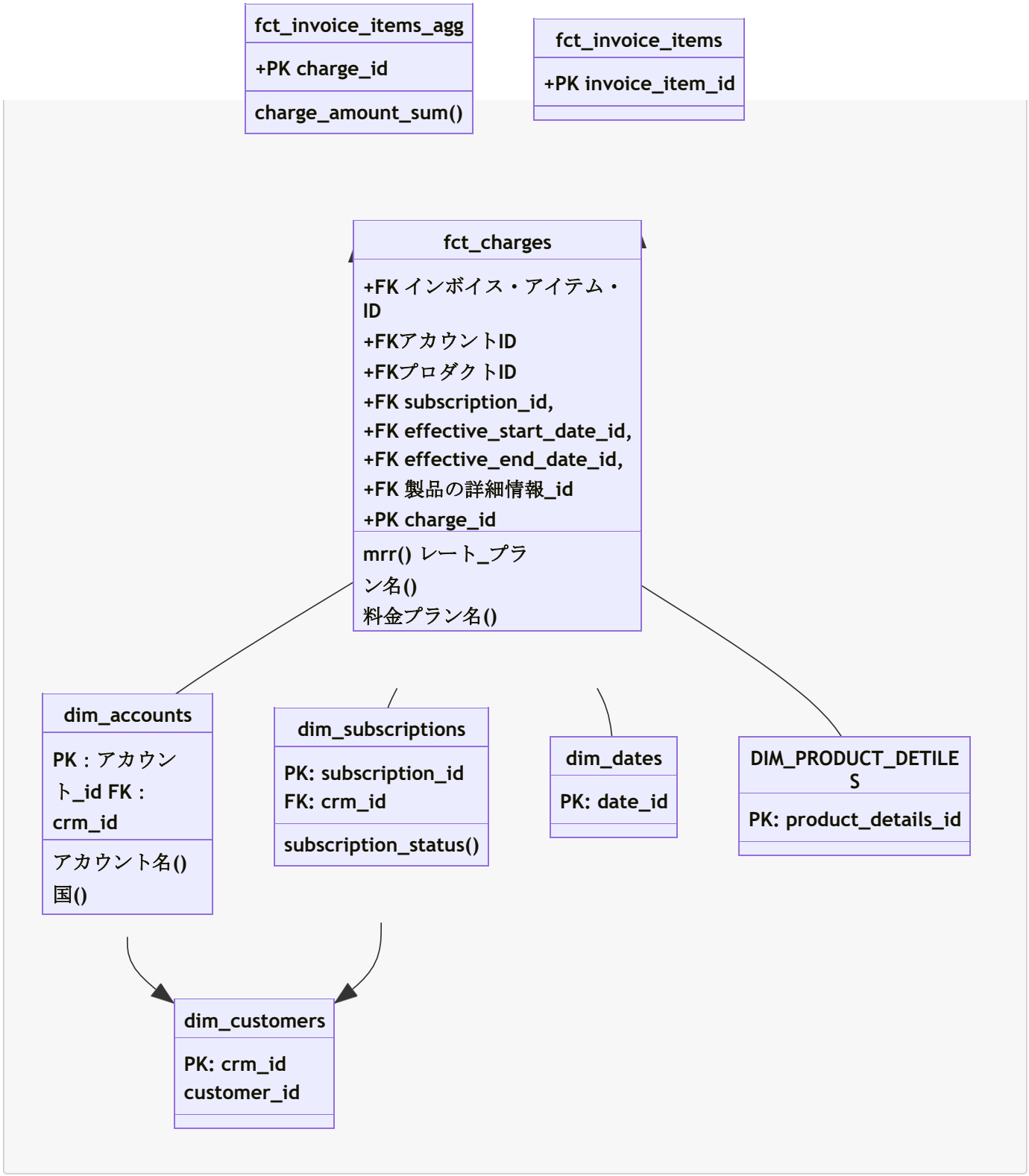
最も単純なバージョンでは、ファクトテーブルは中心的なテーブルであり、外部キーで次元テーブルにリンクされ、スタースキーマを形成している。スタースキーマは、次元テーブルがさらに次元テーブルにリンクしているもので、スノーフレークスキーマと呼ばれ、マルチファクトテーブルスキーマは銀河と呼ばれます。

なぜディメンションモデリングを使う価値があるのか

- **Dimensional Modeling**にはいくつかの種類がありますが、全体的なデザインは業界標準であり、何十年にもわたって使用されてきました。
- **FACT**と**DIM**の構造は、ビジネスチームが理解しやすく、アクセスしやすいデータになります。次元モデリングは、ビ
- ジネスロジックの集中的な実装と、ビジネスユーザー間での一貫した定義をサポートします。
- このデザインは、新しい主題分野の「プラグ・アンド・プレイ」をサポートしており、実際、次元を追加することでモデルの力が増していきます。

最初の反復 - ARRと顧客数の解法

2019-12月に初期イテレーションを提案し、**2020-5**月に**ARR/顧客数**をサポートするモデルを展開しました。



便利なリンクとリソース

- **dbt** 最新のデータウェアハウスにおける**Kimball**の次元モデリングについての談話で、なぜ**Kimball**を使い続けるべきなのかという重要なアイデアを含む。
- ディメンショナル・モデリング・マニフェスト
- キンボールグループの次元モデリング技術

---

layout: handbook-page-toc title:"データインフラ" このページについて

{:.no_toc .hidden-md .hidden-lg}。

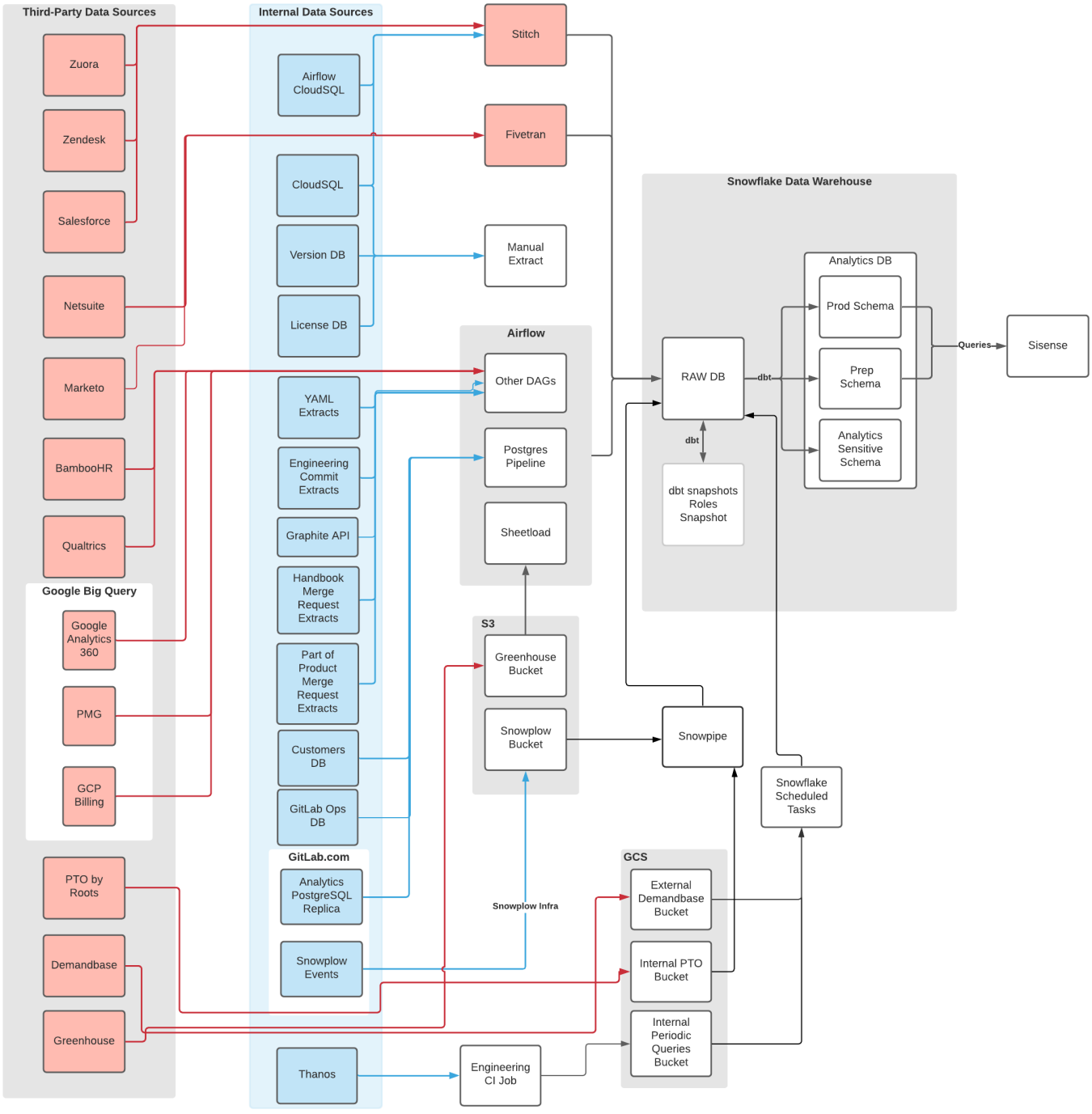
- **TOC {:toc .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

クイックリンク

Airflow{:.btn .btn-purple-inv} DataImageProject{:.btn .btn-purple-inv} GitLabDataUtilsProject{:.btn .btn-purple-inv} PythonGuide{:.btn .btn-purple-inv} Meltano{:.btn .btn-purple-inv}。

システム図



システム図の編集

システム図を編集するには、Lucidchartのドキュメントにアクセスして、必要な変更を加えます。赤枠内の変更であれば、GitLabに自動的に反映されます。表示領域を変更する必要がある場合は、Lucidchartドキュメントを使って新しいリンクを生成して公開し、画像を取り込むために上記で使用したリンクを更新するためのMRが必要です。

エアフロー

オーケストレーションにはすべてAirflowを使用しています。

命名規則

DAGとタスクの名称は、エンジニアや他のユーザーが、気流UIのコンテキストの中でも外でも、DAGが何をするのかを理解できるように、できるだけ説明的であるべきです。名前には、DAGやタスクが何をどのように行っているかという情報を含める必要があります。

例として、`gitlab_com_db_extract` DAG の DAG とタスクの名前を考えてみましょう。この DAG には `gitlab-com-packages-package-files-db-incremental-pgp-extract` のような名前のタスクがあります。タスクと DAG の名前の重複を避けたいところですが、タスクの名前しか見ないこともあるので（Slack や Prometheus のエラー、あるいは airflow で 'Browse' を使ったときなど）、結果的には便利です。

2021年夏以降、私たちが作成した最も一般的なDAGを、より省略して、しかし同じように説明する方法を導入しました。この規則は、それ以降に作成されたすべてのDAGで使用されるべきです。そのDAGで何が行われているかを示すために、プレフィックスを使用してください。例えば、gitlab extract DAGは抽出とロードの両方のフェーズを行うので `el_gitlab_com_db_extract`という名前になりますが、 `greenhouse_extract`はS3から抽出したデータをSnowflakeにロードするだけなので`l_greenhouse`となります。

プレフィックス指標の一覧

プリフィックスインディケータ
eExtract
lLoad
tTransform
pump
mMetadata

製作中

すべてのDAGは`KubernetesPodOperator`を使って作成されるため、エアフローポッド自体の依存関係は最小限に抑えられ、インフラに大きな変更がない限り再起動する必要はありません。

現在のAirflowのデプロイメントでは、deployment.ymlで定義されている通り、4つのコンテナが稼働しています。

1. サイドカーコンテナは、レポのアクティビティフィードをチェックして、masterへのマージがないかを確認します。マージがあった場合、サイドカーはレポをリクローンし、Airflowが最も新鮮なDAGを実行するようにします。
2. スケジューラ「Airflow」について
3. Airflowウェブサーバー
4. Airflow が私たちの cloudsql インスタンスに接続するための cloudsql プ

ロキシ Kubernetes セットアップ

Google Cloud Platform（GCP）の`gitlab-analysis`プロジェクトで実行しています。Airflowは`data-ops`クラスタで動作しています。

ノードプール

このクラスタ内には、`highmem-pool`、`production-task-pool`、`testing-pool`、`sdc-1`という4つのノードプールがあります。各ノードプールには、監視とリソース管理を容易にするために専用の用途があります。

1. `highmem-pool` - Airflowサーバー、スケジューラー、ネットワークコンポーネントの実行に使用されます。1-2ノードからオートスケールが可能です。
2. `production-task-pool` - SCDタスクを除く、ほとんどのプロダクションAirflowタスクの実行に使用されます。2〜5ノードでオートスケールします。
3. `sdc-1` - SCD抽出の実行に使用されます。1〜3ノードでオートスケールします。
4. `testing-pool` - 通常は稼働しているノードを持たないプールですが、エンジニアがローカルに起動したAirflowタスクを実行するために使用されます。1-2ノードからオートスケールが可能です。

`highmem-`poolを除くすべてのnodepoolには、どのnodepoolがどのAirflowタスクを起動するかを管理するためのラベルとテイントが

あります。タスクがノードプールでスケジュールされるためには、タスクはプールに対して**nodeAffinity**を持っていなければならず、また、以下の条件にマッチする**toleration**を持っていなければなりません。

ティングを使用しています。**postgres**パイプラインジョブの**Slowly-Changing Dimensions**タスクに**affinity**と**toleration**を追加したこの**MR**をご覧ください。

名前空間の作成

クラスタが作成されると、デフォルトでは`default`という名前のネームスペースが**1**つ関連付けられます。このプロジェクトのエアフロー設定では、本番環境で**2**つのネームスペースを設定する必要があります。

1. デフォルト
2. テスト

名前空間： - **Kubernetes**では、名前空間は、単一のクラスタ内でリソースのグループを分離するためのメカニズムを提供します。リソースの名前は、ネームスペース内では一意である必要がありますが、ネームスペース間では一意ではありません。

クラスタのプロビジョニングが完了したら、名前空間と秘密のファイルを設定します。**names_space_testing.yaml**には、名前空間の詳細に関する情報が含まれています。

`airflow_image/manifests/names_space_testing.yaml`というフォルダから`kubectl apply -f name_space_testing.yaml`というコマンドを実行します。すると、以下のような出力が得られるはずです。 `namespace/testing`が作成されました。

作成/存在する名前空間のリストを確認するには、`kubectl get namespace`を使用します。

kubeのシークレットの作成

名前空間の作成後、名前空間 `default` と `testing` の両方にエアフローシークレットを作成します。`default` 名前空間での秘密の作成には **kube_secret_default.yaml** を使用します。 `airflow_image/manifests` ディレクトリから以下のコマンド`kubectl create -f kube_secret_default.yaml` を実行すると、以下のような出力が得られます。

```
シークレット/エアフロー作成
```

秘密が正常に作成されたかどうかを確認するには、以下のコマンドを使用します。以下のコマンドを使用します。 `kubectl get secrets --namespace=default` .

テスト用ネームスペースでのエアフローシークレットの作成」と同じ手順で行います。テスト中のシークレットの作成については **namespace use kube_secret_testing.yaml.**

以下のコマンドを実行してください。`kubectl create -f kube_secret_testing.yaml airflow_image/manifests/.`

秘密ファイルのすべての値は、ドキュメントとして**1**パスワードデータチームセキュアボールトに存在し、そのドキュメントの**1**から秘密ファイルに値を追加します。ドキュメントを**testing**の名前空間で編集するには、`kubectl edit secret airflow -o yaml --namespace=testing`というコマンドを使い、デフォルトで作成するには、`kubectl edit secret airflow -o yaml`というコマンドを使います。注：このコマンドは、**deployment.yml**ファイルを適用する前に実行して作成する必要があります。

DNS

airflow.gitlabdata.comという**URL**が私たちのクラスタを指すようにするために、`gcloud compute addresses create airflow-west --region=us-west1`というコマンドを使って、**us-west1**というリージョンに固定**IP**をプロビジョニングしました。生成された**IP**は`35.233.169.210`でした。これは、`gcloud compute addresses list`を実行することで確認できます。**TLS**ステップが機能するためには、スタティック**IP**はグローバル**IP**ではなくリージョナル**IP**でなければならないことに注意してください。

これは、インフラストラクチャチームによって**Route 53**のドメインにマッピングされ、この問題で記録されました。

TLS

このインフラストラクチャーの問題により、**airflow.gitlabdata.com** ドメイン用の証明書が生成されました。この証明書は、`kubectl create secret tls airflow-tls -- cert=airflow.gitlabdata.com.chained.crt --key=airflow.gitlabdata.com.key` というコマンドを実行することで、**kubernetes** のシークレットに保存されました - これら

は、証明書の実際のファイル名です。

証明書と鍵を連結したものです。これにより、秘密のairflow-tlsが作成されました。証明書ファイル（site、chain、chained（site+chain）、key）も1passwordのData Team Secure vaultに保存されています。

NGINX IngressControllerには、リダイレクトやTLSの強化のための優れた組み込み機能があるので、これを使うことにしました。NGINXをクラスタにインストールするには、以下の手順に従います。

1. helmがシステムにインストールされていない場合は、`brew install helm`コマンドでインストールします。
2. その後、コマンド `helm repo add nginx-stable https://helm.nginx.com/stable` を使って、nginx-stable バージョンを helm repo に追加します。
3. 安定版の最新バージョンを取得するには、`helm repo update`を使用します。これにより、以下のような出力が得られるはずです。

```
あなたのチャートリポジトリから最新の情報を取得するまでお待ちください。
...チャートリポジトリ「nginx-stable」からのアップデートに成功しました。
...Successfully got an update from "ingress-nginx" chart repository
Update Complete.⎈Happy Helming!⎈
```

4. helm repo の状態を確認するには、コマンド `helm repo list` を使用します。

```
               NAMEURL
ingress-nginx  https://kubernetes.github.io/ingress-nginx
               nginx-
stablehttps://helm.nginx.com/stable`。
```

5. `NGINX Ingress Controller helm install airflownginx nginx-stable/nginx-ingress -- values nginx_values.yaml` 以下のような出力が得られます。

```
名前：エアフロージンクス
LAST DEPLOYED:Wed Oct 20 19:01:38 2021
NAMESPACE: default
STATUS: deployed
REVISION: 1
テストスイート：
なし NOTES
NGINX Ingress Controllerのインストールが完了しました。
```

NGINXのバリューファイルでは、ロードバランサーのIPアドレスを何にするかを定義しています。ロードバランサーのIPには、前のセクションで生成されたアドレスが設定されます。installコマンドに渡された値は、controller-deployment.yamlファイルで展開されます。

NGINXを削除して再インストールする必要がある場合は、`helm delete airflownginx`で行うこ

とができます。また、ingressの定義もこれらの設定で更新されました。

- force-ssl-redirectはtrue
- tlsはairflow.gitlabdata.comに設定されており、kubernetesのsecretである`airflow-tls`を使用しています。
- ホストにマッチするすべてのトラフィックは、デフォルトのバックエンドではなく、`airflow-webserver`サービスに送られます。

厳密には必要ではありませんが、変更を適用する際にはingressを削除した方がすっきりします。これは、GCPのUIまたは`kubectl delete ingress airflow-ingress`コマンドで行うことができます。新しい設定を適用するには、`kubectl apply -f ingress.yaml` というコマンドを使用します。

GCPでエアフローのログを見る

Airflow のあらゆる種類のログは、GoogleCloudのLogsExplorer で確認できます。このアプリケーションは、収集されるさまざまなログに少し圧倒されることがありますので、いくつかのヒントをご紹介します。

- ログフィールドビューが開いていることを確認します。このビューは、ウェブサーバや特定のDAGの実行など、関心のあるものだけにログをフィルタリングするのに重要です。このビューを開くには、右上の「PAGE LAYOUT」をクリックし、「Log fields」がチェックされていることを確認してください。Log fieldsビューは、実際のログのすぐ左に表示されます。
- Log fieldsビューを使って、特定のプロセスからのログにフィルタリングすることができます。container_nameは、非常に有効なフィルターです。コンテナのほとんどは、エアフローの展開自体に関係しています。特定のDAGランを探している場合は、コンテナ名はすべてのDAGで共有されるベースであるため、より多くのフィルタリングが必要になります。ポッド名は各DAGで一意であるため、特定のDAGランのログを調査する場合は、ポッド名をフィルタリングすることができます。
- 閲覧中のログのタイムフレームを変更するには、現在のタイムフレームが表示された時計のアイコンがあります。このボタンをクリックすると、さまざまなオプションに基づいて新しいタイムフレームを表示することができます。

GCP IAM

{:#gcp-iam}。

エンジニアには、GCPの以下の権限を持つことが推奨されています： Cloud

- SQL Admin
- Kubernetes Engine Admin
- Storage Admin

Airflow をローカルで開発するためには、エンジニアはサービスアカウントも必要です。これらの認証情報は、あなたのマネージャーが用意したサービスアカウントを示すものでなければなりません。アカウントは、あなたの電子メールと同じパターンでなければなりません。例えば、 tmurphy@gitlab- analysis.iam.gserviceaccount.com です。推奨されるパーミッションは以下の通りです。

```
* クラウド SQL クライアント
* Kubernetes Engine Developer
* Storage Object Creator
* ストレージオブジェクトビューア
```

サービスアカウントキーは、JSONとしてダウンロードし、ユーザーのコンピュータに安全に保存するためにユーザ

ーに送信する必要があります。サービスアカウントのプロビジョニング

- gitlab-analysisGCPプロジェクトのServiceAccountセクションに移動して
- Create Service Account をクリックします。
- 個人ユーザーの場合、名前はファースト・イニシャル・ラスト・ネームのパターンに
- 従う 特定の人のためのサービスアカウントであることを示す
- サービスアカウントに関連する権限を付与する サ
- ービスアカウントをユーザーに付与する
- サービスアカウントのJSONキーを生成し、ユーザーに送信する

失敗したジョブの処理

一度に複数の失敗したDAGが表示されることはありません。gitlab.comデータベースからの抽出のようにexecution_dateに依存するインクリメンタルジョブの場合、失敗したDAGはタスクインスタンスをクリアして、修正が適用された後に再実行できるようにする必要があります。

execution_dateに依存しないジョブについては、修正が適用された時点でジョブを手動で再実行し、失敗したDAGrun(s)を削除する必要があります。あるDAGに対して失敗したDAGrunがある場合、それはそのDAGの現在の状態が壊れており、修正する必要があることを意味します。

これにより、AirflowのDAGリストを見て、注意が必要なものとそうでないものがすぐにわかるようになります。バックフィル

あるDAGで増分実行ができなかった場合や、テーブルにデータがない場合、バックフィルを行うには2つの方法があります。テーブルが小さく、バックフィルが比較的短時間で済む場合は、テーブルをドロップして完全同期を行う方法があります。 しかし、アップストリームのエラーでDAGが停止した場合など、テーブル数が多い場合には、この方法が取れない場合があります。

後者の場合は、**Airflow Scheduler**のポッドコンテナで**backfill**コマンドを実行するのが良いでしょう。そのコマンドは

```
airflow backfill gitlab_com_db_extract -s 2019-10-30T00:00:00 -e 2019-11-04T12:00:00 --delay_on_limit 30
-- reset_dagruns
```

これにより、指定された時間枠に既に存在する**DAGrun**とタスクインスタンスがクリアされる一方で、その時間枠に存在しない新しい**DAGrun**が生成されます。フラグが何であるかは、**CLIのAirflow**ドキュメントで詳しく説明されています。

`--task_regex` フラグを使用して、トリガーされるタスクをフィルタリングすることを検討してください。これは、**gitlab.com**のインクリメンタルロードの場合に、**"pgp-extract "**を使用して下流の**dbt**タスクをスキップするのに便利です。

バックフィルされるタイムフレームの**DAG**ランがすでに存在している場合、上記のコマンドを実行すると、タスクが同時に実行されます。**DAG**を連続して実行する必要がある場合。

- まず、**DAG**のスケジューリングをオフにします。
- 次に、**Airflow UI**に入り、「**Browse**（ブラウズ）」から「**DAG runs**（**DAG**ラン）」をクリックします。次に、検索バーを使って、バックフィルされる**DAG**にフィルタリングします。バックフィルされる**DAG**ランをすべて選択し、「`Actions --> Delete`」をクリックします。
- **DAG**ランが削除された今、エアフローはタスクを連続して実行します。しかし、タスクが削除されていないので、エアフローは実際にはタスクを実行しないでしょう。そこで、タスクをクリアするために、`Browse --> Task Instances` に行き、検索バーとソートを使って、再度実行する必要のあるタスクインスタンスを選択します。それらのタスクインスタンスを選択した状態で、`Actions --> Clear`に進み、それらのタスクインスタンスの状態をクリアします。
- この問題が解決したら、**DAG**に戻ってスケジューリングを再開します。
- その後、上記の**backfill**コマンドを実行するが、リセットする**dagruns**がないので、`--reset_dagruns`は省略する。これで**backfill**が連続して実行されるはずです。

In Merge Requests

ローカルでの**Airflow**の使用を容易にしつつ、**Kubernetes**での**DAG**の適切な実行をテストするために、**docker- compose**を使用してローカルの**Airflow**インスタンスをスピンアップし、**KubernetesPodOperator**を使用して**Kubernetes**での**DAG**の実行を可能にしています。**Dockerの項**を参照して、適切な環境変数を設定してください。

Airflowでのコード変更からテストまでの流れは、以下のようになります（これは、そのタスクの**DAG**がすでにあることを前提としています）。

1. コードをコミットして、リモートブランチにプッシュします。
2. `make init-airflow` を実行して、**postgres db** コンテナをスピンアップし、**Airflow** テーブルを初期化します。**Docker**が起動していない場合は、エラーが発生します。
3. `make airflow`を実行して**Airflow**を立ち上げ、コンテナの**1**つにシェルを取り付ける
4. **Web**ブラウザを開き、`localhost:8080`にナビゲートすると、あなた自身のローカル**Web**サーバが表示されます。一般的な**Admin**ユーザーは、ユーザー名とパスワードを`admin`に設定して、**MR airflow**のインスタンスに自動的に作成されます。
5. **airflow** シェルで、テストしたい **DAG/Task** をトリガーするコマンドを実行します。例えば、`airflow run snowflake_load snowflake-load 2019-01-01`（**docker-compose** ファイルで設定されているように、すべての **kube** ポッドは `testing` 名前空間に作成されます）。また、**DAG**全体（例えば、分岐ロジックをテストするための`dbt` **DAG**）を実行したい場合は、`airflow backfill dbt -s 2019-01-01T00:00:00 -e 2019-01-01T00:00:00`のようなコマンドになります。
6. ジョブが終了すると、**DAG/Task**インスタンスにナビゲートしてログを確認することができます。

また、どのようなコマンドがあり、何をするのかを説明する`make help`コマンドもあります。いく

つかのゲッチュ。

- 最新バージョンの**Docker**を使用していることを確認してください。これにより、`ERROR`のようなエラーを防ぐことができます。`./docker- compose.yml "`のバージョンはサポートされていません。
- **dag**で新しい**python**スクリプトを呼び出す場合は、`chmod +x`を実行してファイルが実行可能であることを確認してください。`your_python_file.py`を使用します。これにより、パーミッション拒否のエラーを避けることができます。
- **dag** で追加された新しいシークレットが `kube_secrets.py` にもあることを確認してください。これは、**Airflow** がどの秘密を使用するかの真実の源です。実際のシークレットの値はこのファイルには保存されず、ポインターだけが保存されます。
- イメージが古くなっている場合は、`docker pull <イメージ名>`というコマンドを使って、最新のイメージを強制的にプルし
-

ます。次のようなエラーが発生した場合「could not find an available, non-overlapping IPv4 address pool among the defaults to assign to the network」のようなエラーが出る場合は、実行中のVPNをオフにしてみてください。

ビデオウォークスルー

- エアフロー pt1
- エアフロー pt2
- Airflowのテスト環境。のAirflowとポッドオペレータ（Kubernetes）のテスト環境を動画で解説しました。GCPです。

ローカルエアフローの設定。

すべてのDAGはKubernetesPodOperatorを使用して作成されるため、ローカルで作業する際には、ローカルでタスクを実行する際にポッドをスピンさせることができるクラスタが必要になります。これを実現するためには、クラスターに接続する必要があります。クラスタに接続するには `gcloud container clusters get-credentials data-ops --zone us-west1-a -- project gitlab-analysis` を使用します。**cluset** でローカルセットアップを動作させるためには、**we name** `testing` という名前の名前空間を作成し、**testing** の名前空間に `airflow` **secret** が存在することを確認する必要があります。これを作成するための手順は上記のとおりです。これらの設定が完了すると、ローカルセットアップがクラスター内のポッドをスピンできるようになります。

ローカルエアフロー設定のトラブルシューティング

No Such File or Directory: 'Users/(user)/google-cloud-sdk-bin/gcloud'

```
FileNotFoundErrorです。[Errno 2] No such file or directory: '/Users/(user)/google-cloud-
sdk/bin/gcloud': '/Users/(user)/google-cloud-sdk/bin/gcloud'.
```

これは、MacでのGoogle Cloud SDKのデフォルトのインストール先が上記のようになったためですが、linuxやコンテナでは/usr/lib/google-cloud-sdk/bin/gcloudにインストールされます。この値は、コンテナに渡される/.kube/configファイルです。

このエラーを修正するには、`/.kube/config` を編集して、コマンドパスのパラメータをコンテナ内の場所に更新するだけです。`/usr/lib/google-cloud-sdk/bin/gcloud` このファイルは、SDKをインストールしたり、次のコマンドを実行したりするたびに更新されます： `gcloud container clusters get-credentials data-ops`.詳細については、関連する問題を参照してください。

プロジェクト変数

現在の実装では、以下のプロジェクト変数を使用しています。

- **snowflake_account**
- **snowflake_report_warehouse**
- **snowflake_{flavor}_user**
- **snowflake_{flavor}_password**
- **snowflake_{flavor}_database**
- **snowflake_{flavor}_role**
- **snowflake_{flavor}_warehouse**

以下のようなフレーバーが定義されています。

- LOADフレーバーは、**Extract & Load**プロセスで使用さ
- れます TRANSFORMフレーバーは、**Transform**プロセス
- で使用されます TESTフレーバーは、**Snowflake**を使
- 用したテスト用です PERMISSIONフレーバーは、許可
  ボット用です
- SYSADMINフレーバーは、ハウスキーピング・タスク（レビュー・インスタンスの設定など）のためのものです。このフレーバーではSNOWFLAKE_SYSADMIN_DATABASE」と「SNOWFLAKE_SYSADMIN_WAREHOUSE」です。

以下の変数は、実行環境に応じてジョブレベルで設定されるため、プロジェクトの設定では設定しないでください。

- **snowflake_user**
- **snowflake_password**
- **snowflake_role**
- **snowflake_database**
-

snowflake_warehouse

エアフローのモニタリングとアラート

モニタリング

- エアフローの指標値はThanosで探索することができます。
- Airflow は社内の Prometheus クラスタによって監視されています。Prometheus が Airflow を監視するには、次のような GET リクエストを行います。
  /admin/metricsエンドポイントを定期的に使用しています。
- メトリクスエンドポイントは、GitLab AirflowのDockerイメージにパッケージされているairflow-prometheus-exportをカスタマイズして利用可能にしたメトリクスを公開します。
- airflow-prometheus-exporterパッケージは、airflowサーバー、タスク、DAGに関する様々なメトリクスを取得するようにあらかじめ設定されています。また、このパッケージは、XCOMと呼ばれるものからメトリクスを引き出すこともできます。タスクは、XComを使って相互にデータをやり取りすることができます。タスクから XCom が返されると、それは Airflow データベースに保存されます。prometheus exporterは、Airflowデータベースを読み込むので、XCom の値をメトリクスとして使用することができます。
- KubernetesPodOperatorがXComを返すようにするには、まずタスクがこのようにXComの受け渡しを有効にしている必要があります。私たちが使っているAirflowのバージョンは、XComを有効にする方法のバージョンが混在しているため、このようにXComパッシングを有効にするためのブーリアンが2種類ある場合があります。KubernetesPodOperatorでXComを有効にした後、実行するコードでは特定のファイルにjsonオブジェクトを書き込む必要があります。これを簡単にし、ファイルの場所を抽象化するために、gitlabdata.orchestration_utils.push_to_xcom_file が作成されました。この関数は、JSON オブジェクトを受け取り、それを XCom ファイルに書き込みます。この関数は、タスクごとに一度だけ呼び出す必要があります。XCom内の値をメトリックとして使用するには、メトリックがXComファイルに書き込まれたJSONオブジェクトのファーストクラスのメンバーである必要があります。たとえば、以下のようになります。
  {"record_count":5, "other_record_count":6}であれば、record_countと
  other_record_count as metrics.
- prometheus-exporter で XCom をメトリックとして使用できるようにするには、この設定にタスク名とメトリック名を追加するだけです。すべての気流タスクにインポートされるべきメトリックがある場合は、タスク名 all を使用できます。設定を変更した後、メトリックを表示させるには、「デプロイメントとポッドの再起動」で説明したように、エアフローイメージを再構築して再デプロイする必要があります。

アラート

- 上記のメトリクスに基づくアラートは、runbooks リポジトリの rules/airflow.ymlファイルに定義されています。新しいアラートは、ルールリストに追加することで追加できます。expr要素はPromQL式で、アラートが発生した場合は1を返し、それ以外の場合は0を返す必要があります。for要素は、アラートが実際にトリガーされる前に、どのくらいの時間、式が1と評価されなければならないかを定義します。ラベルは重大度を含みます。重大度は現在GitLabシステム全体で定義されているので、待機中のGitLabインフラメンバーが注意を払わない限り、Airflowルールの重大度は低くすべきです。チームラベルは、最終的にどの slack チャンネルがアラートを受け取るかを決定します。チームラベルは以下のように設定します。
  data-analyticsはランブックのサービスカタログで設定されたチーム名であるため、データチームのすべてのアラートには
  data-analyticsを使用します。
- データ分析チームに割り当てられたPrometheusアラートは、data-prom-alerts Slackチャンネルに送信され、トリアージを担当するチームメンバーが調査・対処する必要があります。

AirflowとKubernetesの共通タスク

ヒント

- kubectlをkbcとエイリアスすることをお勧めします。

エアフローの問題を解決する

時には、何が起こっているのかわからないような故障が起こることがあります。ここでは、よくあるチェック項目をご紹介します。

- GitLab.comとGoogleCloudの状況確認
- 分析プロジェクトやデータイメージプロジェクトへのマージ要求が最近あったかどうかを
- 確認する コンテナログを確認する
- GKEダッシュボード - 現状の把握に役立つ
- GCPエラー報告 - Airflowやプロジェクト全体でどのようなエラーが発生しているかを判断するのに非常に役立ちま

す。

    - GCPのエラーレポートは、2020年2月にクラスタがダウンしたときに役立ちました。エラーは、アプリケーションがストレージ不足であることを報告していました。これにより、「Managing Persistent Volume Claim」で説明されている修正が行われました。
- GCPのKubernetesセクションにある「Workloads」と「Services & Ingress」セクションを確認します。警告やエラーが出ていないか確認してください。
- `kubectl get pods`を実行して、`airflow-deployment`というラベルが付いたものが戻ってくるかどうかを確認します。ポッドに入ってみる
- `kubectl get pods -A`を実行し、名前空間のボトルネックの原因となっている古いポッドや陳腐化したポッドがないかどうかを再確認します。

セミオーソドックスな問題

- ログに`<task> had an event of type DPendingD`と表示され、タスクが開始されません。
    - k8sがレジストリからコンテナを取得できないことが原因の可能性があります。最近の事例については、この問題
    - を参照してください。 GKEのリソース不足が原因で発生する可能性があります。

Kubernetes Airflow Clusterへの接続。

1. Kubectlのインストール
2. `gcloud container clusters get-credentials data-ops --zone us-west1-a --project gitlab-analysis` を実行して、データチームのクラスタに接続します。
3. `kubectl get pods`を実行して、正常に返ってくることを確認します。
4. *すべてのコマンドは本番環境に接続されるため、現在Kubernetesにはテスト環境がありません。*典型的なテスト方法は、ローカルのdocker-composeセットアップを使用することです。

Airflow Webserver UIへのアクセス

- `kubectl port-forward deployment/airflow-deployment 1234:8080`を実行します。これで、ブラウザで`localhost:1234`に移動すると、ポートフォワードしたインスタンスのウェブサーバに移動します。注：安定したURLにアクセスできるようになったので、もうこの作業は必要ありません。

エアフローの更新

- 主要なステークホルダーにアップグレードのタイミングを知らせるイシューを用意する。アップグレードを実行する前に、少なくとも1日前に告知すること
- `airflow_image/Dockerfile`にあるバージョンをバンプさせます。そのラインは`ARG AIRFLOW_VERSION=<version_number>`
- のようになります。この手順の具体的な方法については、以下の「デプロイメントとポッドの再起動」セクションを参照してください。
- ポッド内のいずれかのコンテナに実行し、エアフローのアップグレードを行います。

リソースを見る

- `kubectl get all`を実行します。これで、すべてのポッド、デプロイメント、レプリカセットなどが表示されます。
- `kubectl get pods`コマンドで、現在の名前空間にあるすべてのポッドのリストを見ることができます。

パーシステント・ボリュームの表示

- パーシステントボリュームやパーシステントボリュームクレーム（ログが保存されている場所）の一覧を見るには、それぞれ`kubectl get pv`と`kubectl get pvc`というコマンドを使います。パーシステントボリュームは名前空間に属していないため、get persistent volumesのコマンドは名前空間に関係なくすべてのボリュームを表示します。しかし、パーシステントボリュームの主張は特定の名前空間に属しているため、現在のコンテキストの名前空間内のもののみを表示します。

デプロイメントとポッドの再起動

kubernetesのリソースマニフェストは、`airflow-image/manifests/`に格納されています。kubernetesでこれらのリソースを作成または更新するには、まず`kubectl delete deployment airflow-deployment`を実行し、次に`kubectl apply -f <manifest-file.yaml>`を実行します。

一度に1つのポッドにしか要求できない永続的なボリュームを使用しているため、通常の`kubectl apply -f`による修正はできません。毎回、新しいデプロイメントを設定する必要があります。

- 例えば、**Airflow**のロックアップや再起動の継続、コンテナが使用している**Airflow**イメージの更新など、**Pod**を強制的に再起動する必要がある場合は、`kubectl delete deployment airflow-deployment`を実行します。これにより、すべてのポッドが消去されます（**airflow**によって実行されているものも含まれるので、注意が必要です）。その後、データイメージリポジトリのルートフォルダから`kubectl apply -f airflow_image/manifests/deployment.yaml`を実行すると、マニフェストが**Kubernetes**に送り返され、ポッドが再起動します。

ポッド付きアクセスシェル

{:#access-pod}になります。

- **kube**ポッド内に存在するシェルに入るには、`kubectl exec -ti <pod-name> -c <container-name>`というコマンドを使います。`/bin/bash`です。これにより、選択したポッドやコンテナ内のシェルに入ることができます。この機能は、ウェブサーバの**UI**ではなく、シェルで直接エアフローのコマンドを実行したい場合に便利です。
    - `kubectl exec -ti airflow-deployment-56658758-ssswj -c scheduler /bin/bash` そのポッドと`scheduler`というコンテナにアクセスするコマンドの例です。コンテナ名は、`airflow_image/manifests/deployment.yaml` に記載されています。この情報は、`kubectl describe pod` を実行した場合にも得られます。
      `<pod-name>`さんは、読みにくくなったと思ったようです。
        - 補足：リソース/名前の形式で引数を渡す場合、リソースタイプを別の引数として指定する必要はありません（例：**'kubectl get resource/<resource_name>'**ではなく**'kubectl get resource resource/<resource_name>'**
    - また、ローカルでコンテナを実行する際には、**docker compose**コマンドを使って行うこともできます。
    - 起動している既存のコンテナに入るには
        - `docker-compose exec airflow_scheduler bash`
    - スピンアップしてバシッと決めること。
        - `docker-compose run airflow_scheduler bash`

- シェルに入ってからやることダグで特定の
    - タスクを起動する。
        - テンプレート：`airflow run <dag> <task_name> <execution_date> -f -A`
        - 具体的な例：`Airflow run dbt dbt-full-refresh 05-02T15:52:00+00:00 -f -A`
        - また、`-A` フラグは依存関係を無視します (つまり、上流にブランチされていなくても気にしません)。

秘密の更新

- **prod**環境で**secret**を更新する最も簡単な方法は、`kubectl edit secret airflow -o yaml`というコマンドを使うことです。このコマンドを使うと、テキストエディタで**secret**が開かれ、そこから編集することができます。新しいシークレットは**base64**エンコードされている必要があり、最も簡単な方法は`echo -n <secret> | base64 -`です。シークレットファイルを編集する際に、いくつかのヌル値があります。ファイルを正常に保存するためには、ヌル値を `""` に変更しなければなりません。
- 新しいシークレットを追加する際には、そのシークレットを適切な値でテスト環境に追加することも忘れてはいけません。そのためのコマンドが`kubectl edit secret airflow -o yaml --namespace testing`です。このコマンドは、上記の本番環境用のガイドラインと同じです。テスト環境に新しいシークレットを追加しないと、そのシークレットを使用する**DAG**はテスト時に実行されません。

稼働中の**DAG**を停止する

- 問題のダグのグラフビューに移動する グラフビュ
- ーでタスクを選択する
- ポップアップしたモーダルで、「**Mark Failed**」または「**Mark Success**」のいずれかを選択し、「
- **Downstream**」オプションを選択します。**Kubernetesworkloads**」タブで、該当するポッドが停止している
  ことを確認します。必要に応じて削除します。

DAG内で特定のタスクを実行する

- `kubectl exec -ti <pod_name> -c <webserver|scheduler>` というコマンドで、コンテナ内で動作しているシェルを取得します。
  `/bin/bash`

- 最も単純なケースでは、`Airflow run <dag_id> <task_id> <execution_date>`で十分です。

- `dbt full-refresh`を実行する必要がある場合など、より複雑なケースでは、さらにいくつかのフラグが必要です。`airflow run dbt dbt-full-refresh <execution_date> -f -A.f` フラグは、すでに成功または失敗とマークされていても、タスクを強制的に実行します。`A` フラグは、持つべき依存関係に関係なく実行するように指示します。

エアフロー変数の設定・変更

- 時折、**Airflow** 変数を使用することがあります。現在、**BambooHR** データ抽出時のデータ整合性チェックをオーバーライドするために、**Airflow** 変数を使用しています。**Airflow** の変数は **Airflow** のデータベースに格納されており、`Airflow` で実行されるコードから簡単にアクセスすることができます。
- 変数の設定や値の変更を行うには、**Airflow UI**に入ります。**"Admin "**タブをクリックし、**"Variables "**をクリックします。新しい変数を追加するには、＋ボタンを押します。必要なキーと値を入力します。値を暗号化する場合は、ボックスをチェックします。その後、保存を押します。変数を編集するには、対象となる変数の横にある編集アイコンをクリックし、必要な内容を変更します。その後、保存ボタンを押します。
- **BambooHR**の整合性チェックのバイパスでは、変数のキーを`BAMBOOHR_SKIP_TEST`とします。値は、一時的にテストをスキップしたいテーブルの名前をコンマで区切ったリストにします。**BambooHR** の抽出が成功したら、この変数を削除するか、値を空の文字列に変更してください。

パーシステントボリュームクレームとそのデータの管理

{:#managing-pvc}になります。

Airflow の **persistent volume claim** は、`persistent_volume.yaml` マニフェストで定義されます。これは、ログ用のデプロイメントで使用され、次のディレクトリに書き込まれます。 `/usr/local/airflow/logs` ディレクトリに書き込みます。.パーシステントボリュームクレームがいっぱいになった場合、**2**つの解決策があります。

- クレームに利用できる容量を増やす クレー
- ムからデータを削除する

データの削除

- **airflow-deployment podに入って、**`logs`
- ディレクトリに移動する。
- ファイルの削除

クレームサイズの拡大

- オブジェクトブラウザを開き、**core > PersistentVolume**で**DataOps**クラスタのクレームを見つけます。
- **PersistentVolume**をクリックします - `pvc-<GUID>`である必要があります。
- そのボリュームの**YAML**を見て、`gcePersistentDisk`の下にある`pdName`を見つけます。これが、アップデートが必要な**GCE**ディスクです。
- **GCPのComputeDisksセクション**に行き、参照されているディスクを
- 見つけます。 そのディスクをクリックして、サイズを更新するために
  編集します。
- **PersistentVolume**の**YAML**に戻り、`spec:capcacity`の下にある`storage`を更新します。
- これでクレームサイズが大きくなります。

または、プロジェクト内の`persistent_volume.yaml`定義を更新することもできます。ただし、これを再デプロイすると、すでにクレームにあるデータが削除される *可能性があります*。これはまだテストされていません。

Postgres パイプライン

開発

postgres pipelineによって引き出される新しいテーブルやフィールドを追加するには、特定のソースデータベースのマニフェストファイルを変更する必要があります。これらのマニフェストファイルは、ここのフォルダにあります。新しいテーブルを追加するには、テーブルリストに項目を追加します。`import_db` の値は、そのマニフェストの他の項目と一致する必要があります。`import_query` はソース・ターゲット・データベースに対して直接実行されます。
可能であれば、次のような`WHERE`句を追加して、インポートクエリをインクリメンタルにしてください。

```
WHERE updated_at BETWEEN '{EXECUTION_DATE}'::timestamp - interval '{HOURS} hours' and
'{EXECUTION_DATE}'::timestamp
```

`export_schema` の値は、マニフェストの他の項目と一致する必要があります。`export_table`値は、インポートされるテーブルの名前と一致する必要があります。`export_table_primary_key`は、**postgres**データベース内の特定のテーブルの主キーに設定する必要があります。インポートクエリがインクリメンタルでない場合は、データがいつロードされたかを区別する別のカラムを作成する `advanced_metadata: true` を追加します。

マニフェストの変更が完了したら、`add_manifest_tables` テンプレートを使用してマージ リクエストを作成します。その後、そこでの指示に従います。

技術的な実装の詳細については、こちらの**README**をご覧ください。

エアフローでのテスト

Airflowでローカルに**postgres pipeline(pgp)**をテストする場合、いくつか注意すべき点があります。

- プールは、どの**pgp DAG**にも存在する必要があります。プールを追加するには、**[Admin] > [Pools]**に進み、**https://airflow.gitlabdata.com/pool/list/** に記載されているのと同じプールを追加します。

Bashスクリプト

- データチームでは、基本的なコマンドラインの使い方と高度なコマンドラインのに関するトレーニングを実施しています。このトレーニングには、アナリティクス・レポ内での高度な**bash**スクリプトの使用に関する詳細も含まれています。

ドッカー

コンポーズ

私たちは**Docker compose**を使って、さまざまなイメージやアプリケーションを定義し、実行しています。これらはユーザーが**Makefile**を介して起動します。

環境変数

Makefileと**Docker Compose**を適切に使用するために、いくつかの環境変数が必要になります。

- `GIT_BRANCH` - 通常、`2112-my-feature`のような機能ブランチです。
- `KUBECONFIG` - **Kubernetes**の認証情報をローカルマシンに保存する場所を定義します。
  `KUBECONFIG="/Users/tmurphy/.kube/config"`
- `GOOGLE_APPLICATION_CREDENTIALS` - **GCP**に接続するための**CloudSQL**認証情報。通常は、**JSON**ファイルを指します。
  `GOOGLE_APPLICATION_CREDENTIALS="/Users/tmurphy/Projects/GitLab/gcloud_service_creds.json"`
    - これらの認証情報は、**GCP IAM**セクションの指示に従ってプロビジョニングする必要があります。

イメージ

データ

`data_image`ディレクトリには、`data-image`の構築とプッシュに必要なものがすべて入っています。バイナリをインストールする必要がある場合は、**Dockerfile**の中で直接行います。**python**パッケージは`requirements.txt`ファイルに追加し、動作確認済みのバージョンに固定します。

エアフロー

`airflow_image`ディレクトリには、`airflow-image`だけでなく、対応する**Kubernetes**のデプロイメントマニフェストをビルドしてプッシュするために必要なものがすべて含まれています。新鮮なデプロイメントのために必要な唯一の手動作業は、`airflow`のシークレットを設定することです。必要なシークレットは、`airflow_image/manifests/secret.template.yaml`に記載されています。

デフォルトの **airflow** インスタンスは **production** インスタンスで、`airflow` の **postgres db** を使用します。テスト用のインスタンスで

は

`airflow_testing` **db.**

デフォルトのインスタンスのログは `gs://gitlab-airflow/prod` に、テスト用のインスタンスのログは `gs://gitlab-airflow/testing` に保存されます。

dbt

`dbt_image`ディレクトリには、データイメージのビルドとプッシュに必要なものがすべて入っています。バイナリをインストールする必要がある場合は、**Dockerfile**で直接行います。**python**パッケージは`requirements.txt`ファイルに追加し、動作確認済みのバージョンに固定します。このイメージはデータアナリストが使用するものなので、イメージの中に**dbt**以外のものがあってはいけません。

新規画像の作成

{:#new-images}をご覧ください。

本番用のイメージは、**master** ブランチに **git** タグがプッシュされたときにのみ作成されます。現在のリリースフローは以下の通りです。

```
# master ブランチか feature ブランチのどちらかで実行: git
tag v<sem_ver>

# タグを押す
git push origin $(git describe --tags --abbrev=0)
```

ビルドパイプラインが自動的に実行されます。これらの画像が参照されている場所で新規に**MR**を作成し、タグを更新します。

パイソンのハウスキーピング

レポの**python**をきれいに維持するために、複数の`make`コマンドと**CI**ジョブが設計されています。**Makefile**の中の以下のコマンドは、レポの分析に役立ちます。

- `make lint` は `black` **python linter** を実行し、ファイルを更新します **(**これは単なるチェックではありません**)**
- `make pylint`は、**pylint**チェッカーを実行しますが、コードのフォーマットをチェックするために`black`を使用しているため、チェックしません。重複したコード、エラー、警告などをチェックします。コードの品質を高めるための一般的なものです。**DAGs dir**は、一般的なコード標準に従うことが期待されていないため、無視されます。
- `make radon`は、関連する**Python**コードのサイクロマティックな複雑さをテストし、スコアが`B`の関数やモジュールを表示します。またはそれ以下です。
- `make xenon`は複雑性のチェックを行い、閾値を満たさない場合はゼロ以外の終了コードを返します。を無視します。`shared_modules`や`transform` **repos**は、削除されたり、廃止されたり、後日更新されたりするまで。

ピアリングされたVPCへのアクセス

GitLab固有の**ELT**の中には、ピアリングされた**GCP**プロジェクト内のデータベースに接続するものがあります（**ping**など）。接続を許可するために、いくつかのアクションが取られています。

1. ランナーが実行される**Kubernetes**クラスターは、**IP**エイリアシングを使用するように設定されているので、各ポッドは**GCP**内で実際にルーティング可能な**IP**を取得します。
2. **2**つのプロジェクトとそのネットワークの間には、**VPC**ピアリング関係が確立されています。
3. ランナーとなる**Kubernetes**クラスターのポッドサブネットからのアクセスを許可するために、上流のプロジェクトでファイアウォールルールが作成されました。

Extractorのドキュメント

- **BambooHR**
- コミット統計（エンジニアリ
- ング） グラファイト（エンジ
- ニアリング） **SheetLoad**
- ハンドブックの**YML**ファイル

ランナーの更新

CIジョブは`gitlab-data`グループで、**GCP**プロジェクトの`gitlab-analysis`を介して**Kubernetes**で実行しています。すべてのリポジトリで共有するためにグループランナーを設定しています。

新しいグループのランナー・トークンを関連付ける必要がある場合や、ランナー・イメージを更新する必要がある場合などです。以上が基本的な手順です。注 - **helm 3**のリリース以降、これらのコマンドはすべて**GCP**の**Cloud Shell**コンソールで実行することが推奨されています。ランナー用のデプロイメント（現在は`gitlab-data-gitlab-runner`）に移動し、**kubectl**のドロップダウンを使用してシェルに入ります。

クレデンシャルを取得するには

```
gcloud container clusters get-credentials bizops-runner --zone us-west1-a --project gitlab-analysis
```

ヘルスリリースを見るには

```
helm list --namespace <namespace> を表示します。
```

特定のリリースのチャート値を取得するには

```
helm get values --namespace <namespace> <chartname>
```

準備のためのコマンド

```
helm repo add <chart>
```

```
<url> helm repo update
```

ランナーを削除するには

```
helm delete --namespace <namespace> <chartname>
```

Helmチャートでランナをインストールするには

```
helm install --namespace <namespace> --name <chartname> -f <valuesfile> <chartrepo/name>
```

ランナーのバージョンやグループトークンを更新する例

```
helm list --namespace gitlab-data
helm get values --namespace gitlab-data gitlab-runner
helm get values --namespace gitlab-data gitlab-runner >
values.yaml helm repo add gitlab https://charts.gitlab.io
helm repoの更新
helm delete --namespace gitlab-data gitlab-runner
helm install --namespace gitlab-data --name gitlab-runner -f values.yaml gitlab/gitlab-runner
```

私たちの**YAML**設定は以下の通りです。

```
アフィニティです。{}
checkInterval:30
コンカレント：10
gitlabUrl: https://gitlab.com
hostAliases:[]
imagePullPolicy:IfNotPresent
metrics:
  enabled: true
nodeSelector:{}
podAnnotations:{}
podLabels:{}
rbac:
  clusterWideAccess: false
  create: false
  podSecurityPolicy:
    enabled: false
```

```
      resourceNames:
      - gitlab-runner
    serviceAccountName: gitlab-data-gitlab-runner
  resources:{}
  runnerRegistrationToken: <token found in https://gitlab.com/groups/gitlab-data/-/settings/ci_cd> ランナー。
    のビルドを行います。{}
    キャッシュ
    を使用しま
    す。{}
    helpers:{}
    image:  ubuntu:16.04
    outputLimit: 4096
    pollTimeout:180
    privileged: false
    services:{}
    タグ：アナリティクス,ハウスキーピングセキュリティコンテクスト
    fsグループです。65533
    runAsUser: 100
```

Snowflakeにデータをロードする自動化プロセス

バージョンDBとライセンスDBのロードタスク

ソースPostgres DBデータエクスポート

バージョンプロジェクトとライセンスプロジェクトで実行される1日1回のCIジョブがあります。このジョブでは、データベースエクスポートのバージョンスクリプトまたはライセンススクリプトを実行し、GCSバケットにCSVファイルをエクスポートします。これらのファイルは`gitlab-version-{table_name}-`という名前です。
それぞれ`{monday_of_week}`または`gitlab-license-{table_name}-{monday_of_week}`となります

。スノーフレークステージ

バージョンDBのCSVファイルがエクスポートされるGCSバケットは、Snowflakeではステージとして設定されています。`raw.version_db.version_dump`.ライセンスDBファイルがエクスポートされるGCSバケットは、ステージ`raw.license_db.license_dump`として設定されています。つまり、Snowflakeからは、すべてのファイルを一覧表示したり、ファイル内のデータをテーブルにコピーしたり、ファイルを削除したりすることができるのです。

カラムマッチングとテーブル生成

CSVファイルは自己記述的ではありません。どの列がどの位置にあるかを知るための列ヘッダーがありません。このため、RAWのテーブルは、CSVファイルの順番と正確に一致した列の順番を持つ必要があります。このテーブルを簡単に作成するために、このbashスクリプトを作成しました。create文を生成するために。

1. 目的のデータベースのdeclare文のコメントを外す
2. 環境変数PATH_TO_MANIFESTSに、クローン化したバージョンのsqlソースフォルダへのパスを設定します。version dbでは、このフォルダのクローンバージョンを指すようにします。license dbの場合は、このフォルダを指定します。
3. `do_create_tables.sh`を実行すると、すべてのcreate table文が出力されるはずです。

スノーフレークのタスク

CSVファイルは、スノーフレークステージからスノーフレークタスクで毎日読み込まれています。タスクは以下のようなSQLを実行して生成されています。

```
タスク users_load_task WAREHOUSE =
       LOADING を作成または置換します。
       SCHEDULE = '1440
       minute' AS
        COPY INTO users
        from @raw.version_db.version_dump/gitlab-version-users-
        file_format = (TYPE = CSV
        FIELD_OPTIONALLY_ENCLOSED_BY='"')
```

を **LOADER** ロールで実行します。これらのタスクは毎日実行され、新規または更新されたファイルのみをロードします。タスクの定義を確認するには、**LOADER**ロールで`show tasks`を実行し、コンテキストの一部として`version_db`または`license_db`スキーマを指定します。

タスクモニタリング

タスクは**dbt**テストで監視されており、各タスクが過去**1**日以内に正常に実行されたことを確認します。バージョンテストとライセンステストは、それぞれ内部のデータテストプロジェクトのこことここにあります。タスクの問題を診断するために、`raw.snowflake.task_history_view`を照会して、失敗したタスクの`error_message`列を検査することができます。

PTO by Roots Snowpipe

本号で紹介したように、**people**グループが維持しているプロセスでは、定期的に**Roots API**で**PTO**に問い合わせを行い、その結果を**GCS**バケットの`gitlab-pto`にある**JSON**ファイルにダンプしています。このバケットは`gitlab-analysis`という**GCP**プロジェクトの中にあります。

このデータを**Snowflake**に読み込むために、**Snowflake**のドキュメントを参考にして、**GCS**用の**Snowpipe**を設定しました。**gitlab-pto-snowpipe**という名前の**GCP pubsub**トピックは、新しいファイルが`gitlab- pto`バケットに書き込まれるたびに新しいメッセージを受け取るように設定されました。`gitlab-pto-snowpipe`という名前の**GCP Pub/Sub**サブスクリプション（配信タイプは`Pull`）を作成し、`gitlab-pto-snowpipe`トピックを購読しました。

そして、**Snowflake**で以下のコマンドで通知統合を作成しました。

```
通知統合の作成 pto_snowpipe_integration type = queue
    notification_provider = gcp_pubsub
    enabled = true
    gcp_pubsub_subscription_name = 'projects/gitlab-analysis/subscriptions/gitlab-pto-snowpipe';
```

そして、このコマンドで**Snowpipe**を作成しました。

```
CREATE OR REPLACE PIPE
    raw.pto.gitlab_pto AUTO_INGEST =
    true
    インテグレーション = 'pto_snowpipe_integration'
    AS copy into raw.pto.gitlab_pto (jsontext, uploaded_at)
    from (select $1, current_timestamp() as uploaded_at from
    @raw.pto.pto_load) file_format=(type='json' strip_outer_array = true);
```

Snowpipeが有効になっていたので、`gitlab-pto`バケットに書き込まれた新しいファイルをうまく拾うことができました。

```
alter pipe raw.pto.gitlab_pto enable;
```

デバッグ

PTOのスノーパイプに問題がある場合は、まずスノーパイプの状態を確認します。これは**Snowflake**で次のように実行することで可能です。

```
select SYSTEM$PIPE_STATUS('gitlab_pto')。
```

Demandbase Load Tasks

今回の課題の一環として、**Demandbase**のデータロードを実施しました。**Demandbase**のデータは、**Demandbase**によって毎日、`datastream-hosted-gitlab-3750`という**Demandbase**が所有する**GCS**バケットに**parquet**形式でロードされます。**GitLab**のスノー

フレーク**GCS**

統合サービスアカウントは、**Demandbase**から**Snowflake**にデータをロードするために、このバケット内のファイルの読み取りとリスト化の権限を与えられました。

デマンドベースのデータとのインターフェイスとして、**Snowflake**ステージを作成しました。

```
CREATE STAGE "RAW".demandbase.data_stream
STORAGE_INTEGRATION = GCS_INTEGRATION
URL = 'gcs://datastream-hosted-gitlab-3750/datastream-gitlab/';
```

その後、**GCS**から**Snowflake RAW**データベースにロードするために、各**demandbase**関係の**Snowflake**タスクを作成しました。例えば、**Snowflake**タスクのロードアカウントは次のように定義された。

```
create task demandbase_account_load_task
       WAREHOUSE = LOADING
       SCHEDULE = '1440
       minute' AS
copy into raw.demandbase.account (jsontext, uploaded_at)
    from (select $1, current_timestamp() as uploaded_at
    from)
@raw.demandbase.data_stream/db1_accounts/)
    file_format=(type='parquet');
```

その後、`alter task DEMANDBASE_ACCOUNT_LOAD_TASK resume;`を実行して、タスクを有効にしました。各タスクは毎日実行し続け、**GCS** からの新しいファイルを **Snowflake** の `raw.demandbase` スキーマにロードします。

Thanos ロードタスク

この問題の一部として、**thanos**のメトリクスを毎日取得し、**GCS**のバケットに書き込むプロセスが設定されて

います。**GCS**から**Snowflake**にメトリクスを取り込むために、ステージを作成しました。

```
create stage "raw"."PROMETHEUS".periodic_queries
STORAGE_INTEGRATION = GCS_INTEGRATION URL = 'gcs://periodic-queries/';
```

その後、**Snowflake**タスクが設定され、新しいデータファイルが毎日読み込まれます。

```
タスク prometheus_load_task WAREHOUSE =
       LOADING を作成または置換する。
       SCHEDULE = '1440
       minute' AS
    raw.prometheus.periodic_queries (jsontext, uploaded_at)にコピーします。
    from (select $1, current_timestamp() as uploaded_at from
    @raw.prometheus.periodic_queries) file_format=(type='json');
```

データの更新

スティッチ管理データ

SLO違反につながるテスト失敗のためにフルリフレッシュが必要な場合、テスト失敗の根本原因を調査する時間が必要です。フルリフレッシュを行う前にインシデントを起こし、発見事項を文書化する。また、データを別の一時的なテーブルにコピーして、フルリフレッシュが妨げられないようにすることも有効です。

Stitchで抽出されたデータソース（プラットフォームページの抽出テーブルを参照）の場合、フルリフレッシュを行う推奨方法は以下の通りです。

- **Stitch**のレプリケーションジョブが現在実行されていないことを確認します。統合を一時停止して、このプロセス以外の実行を開始しないようにします。
- **STITCH**ロールとして、リフレッシュ・プロセスが本番環境に何の影響も与えないように、ターゲット・スキーマをクローンする。これは、以下の**SQL**を実行し、関連するスキーマ名を記入することで行うことができる。

```
create schema clone_<DATA_SCHEMA> clone <DATA_SCHEMA>;
```

- 次の**SQL**クエリを実行して、すべてのテーブルを取得します（関連するスキーマに変更します）。

```
SELECT
 'clone_<DATA_SCHEMA>が存在すればテーブルを切り捨てる。|| LOWER(テーブル名) ||
';' FROM "RAW".「INFORMATION_SCHEMA".テーブル
WHERE LOWER(table_schema) = '<DATA_SCHEMA>';
```

- **Snowflake**ですべての切り捨てクエリを実行する
- **Stitch**に戻り、必要なタイプの新しい統合を作成します。統合の名前を「Clone <古い統合の名前>」とします。例えば、**Salesforce**の場合、統合の名前を「Clone SalesForce Stitch」とします。
  統合を認証し、現在の**Stitch**ジョブの抽出設定に合わせて抽出を設定します。抽出の実行
- 抽出が完了すると、すべてのデータが`clone_<DATA_SCHEMA>`スキーマに入っているはずです。**SQL**クエリでサニティチェックを行い、データが期待通りに表示されていることを確認します。抽出が完了した後に**Stitch**のデータが到着する場合がありますので、**Stitch**からのデータがすべて到着していることを確認するための時間を確保してください。
- 各テーブルに必要な**SQL**を取得して、新しいデータと古いデータを入れ替えます（`<DATA_SCHEMA>`の置き換え適切に）。)

```
SELECT
 'alter table <data_schema>.|| LOWER(テーブル名) || ' SWAP WITH CLONE_<DATA_SCHEMA>.'.||
LOWER(テーブル名) || ';'
FROM "RAW"."INFORMATION_SCHEMA".tables
WHERE LOWER(table_schema) = '<DATA_SCHEMA>';
```

- 上記の**SQL**で得られた各**SQL**文を実行します。
- 抽出」→「設定」→「削除」で、クローン化された抽出ジョブを削除します。 古い
- **Stitch**レプリケーションジョブを再開し、増分データが再び流れ始めるようにします
  。
- リフレッシュされたデータが完成し、期待通りであることが明らかになった時点で、`drop schema clone_<DATA_SCHEMA> cascade;` を実行して古いデータを削除します。

dbtモデルフルリフレッシュ

`dbt_full_refresh` **DAG**を使用して、**dbt**にインクリメンタルモデル全体を一から作り直させます。

1. **Airflow**で、変数`DBT_MODEL_TO_FULL_REFRESH`に、リフレッシュするモデルの名前を**dbtmodelselection構文**に従って設定します。例えば、バージョンモデルを更新するには、`sources.versionstaging.`**version**とします。**gitlab_dotcom**モデルを更新する場合、値は`sources.gitlab_dotcom staging.gitlab_dotcom`となります。  **airflow_variable_setting**
2. **DAG**の電源を入れ、起動させま

す。後ろで実行される**dbt**コマンドは

```
dbt run --profiles-dir profile --target prod --models DBT_MODEL_TO_FULL_REFRESH --full-refresh
```

GitLabデータユーティリティ

チーム内で使用している便利な機能を集約するためのプロジェクトです。プロジェクトは **https://gitlab.com/gitlab-data/gitlab-data-utils**

リリースのカット

1. **setup.py**のマイナーバージョンをインクリメントします。マイナーバージョンは**2**番目の数字で、`0.1.0`の場合は`1`となります。現在、マイナーバージョンの更新のみサポートしています。
2. この変更に対して**MR**を作成し、**master**にマージします。
3. チェックアウトしてローカルに **master** をプルし、今行ったバージョンの変更を手元に残しておきます。`gitlab-data-utils`のルートフォルダで`make release`を実行すると、保留中の変更がないことが確認され、新しいタグが**GitLab**リポジトリにプッシュされます。これにより、**pypi** に公開するためのパイプラインが実行されます。
4. **pypi**との一貫性を保つために、**GitLab**のリリースを以下のようなコマンドでカットします。

```
curl --header 'Content-Type: application/json' --header "PRIVATE-TOKEN: <your-private-token>" ⦿。
 --data '{ "name":"GitLab Data Utilities v0.0.1", "tag_name":"v0.0.1",
"description":"Initial tagged release"}'\
 --request POST https://gitlab.com/api/v4/projects/12846518/releases
```

5. このマージリクエストで行われたように、**Docker**イメージを更新します。

本番用にdbtをアップグレード

1. 主要なステークホルダーにアップグレードのタイミングを知らせるイシューを用意する。アップグレードを実行する前に、少なくとも**1**日前に通知すること
2. 新規イメージの作成」の手順に従って、新しいバージョンの`dbt-`**image**を作成します。
3. アナリティクスプロジェクトに**MR**を作成し、以下の項目を更新します。例として、このMRをご覧ください。

- **airflow_utils.py dbt_project.yml** 以
  - 下の場所で **dbt-image** を使用し
  - ます。
  - **snowflake-dbt-ci.yml**
  - **.gitlab-ci.yml**
  - **docker-compose.yml**
- `packages.yml`に含まれる任意
  のパッケージ
- `index.html`を以下のようにして作成します。
  - `make dbt-docs` コマンドで **docs** サイトを読み込む。
  - 右クリックして「**View Page Source**」を選択するか、**Mac**の場合は「**CMD + u**」、**Linux**の場合は「**CTRL + u**」を押すと、ページのソースコードが表示されます。
  - `<body>`からファイルの一番下までを更新する
  - `</head>`からファイルの先頭までのコードに大きな変更がないことを確認します。

1. `data-tests`プロジェクトに**MR**を作成し、**image**バージョンと`require-dbt-version`を更新します。例として、このMRを参照してください。
2. リリースの機能アップデートに基づいて、関連する**dbt**ジョブを実行する。
3. マージされたら、全員にアップグレードを通知し、ジョブの失敗やユーザーの問題を監視する

---

layout: handbook-page-toc title:"Jupyter Guide" 説明"JupyterLabを使ってSnowFlakeと内部でやりとりするためのガイダンス"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

関連リポジトリを見る

特徴

- 一般的なpythonのDS/MLライブラリ（pandas、scikit-learn、sci-pyなど）。
- dbtの認証情報を使ってSnowflakeにネイティブ接続。ログインは必要ありません。
- Git機能：JupyterLab内でGitlabリポジトリへのプッシュ／プルが可能（ssh認証が必要）。
- コンピュータやGitlabのリポジトリにある任意のpythonファイルやノートブックを実行します。
- 使っているのに見当たらない機能が必要ですか？ぜひ#bt-data-scienceでお知らせください！追加します。

はじめに

data-scienceプロジェクトでjupyterをセットアップする際には、2つのオプションがあります。以下のいずれかを選択

- してください。フル・インストール（推奨）。Pipfileで定義されたすべてのライブラリと、完全な

  anacondaのインストールを行います。
- Lightweight install: Pipfileで定義されたライブラリのみをインストールします。ローカルマシンにpython環境があり、それをベースイメージとして使用したい場合に使用します。

インストール手順

1. レポをローカルマシンにクローンします `git clone git@gitlab.com:gitlab-data/data-science.git`
2. 実行 `cd データサイエンス`
3. どのバージョンをインストールするかに応じて、以下のいずれかを実行してください。
   - *完全にインストールするには*、`make setup-jupyter-local`を実行します。
   - *軽量インストールの場合*： run `make setup-jupyter-local-no-conda`
4. `make jupyter-local` を実行します。
5. Jupyterが自動的に起動するはずです。起動しない場合は
   1. まず、Google Chromeがデフォルトのブラウザであることを確認します（「システム環境設定」から「一般」をクリックし、「デフォルトのウェブブラウザ」セクションのドロップダウンメニューからGoogle Chromeを選択します）。
   2. 次に、Chromeで、dockerイメージが作成された後にターミナルで見つけたURLとトークンをコピーペーストします。`http://127.0.0.1:8888/lab?token=5c7f7da79f4a0968501f087f3c79ee4dd8bd7a63e0f088a8` のように表示されるはずです。トークンは、dockerコンテナをスピンアップするたびに変わります。

Snowflakeへの接続

1. パスワードが含まれていない`{User}/.dbt/profiles.yml`ファイルを設定していることを確認してください。ここで提供されている例を参考にしてください。
2. レポにあるauth_exampleノートブックを実行して、すべての設定が正常に行われたことを確認します。初めて実行すると、Okta経由でsnowflakeのクレデンシャルを認証するためのブラウザのリダイレクトが表示されます。その後、ノートブックを再度実行すると、Snowflakeからデータを照会できるようになるはずです。
3. エラーが出る場合は、Snowflakeがマシン上で正しく設定されていない可能性があります。DataOnboardingIssueのSnowflakeとdbtのセクションを参照してください。.dbt/profiles.ymlが正しく設定されていない可能性があります。

ローカルディレクトリのマウント

デフォルトでは、ローカル・インストールでは、data-scienceフォルダをjupyterのルート・ディレクトリとして使用します。これは、すべてのコード、データ、ノートブックがコンピュータの他の場所にある場合、あまり便利ではありません。これを変更するには、jupyter notebook configファイルを作成して変更する必要があります。

1. `jupyter-lab --generate-config`を実行します。すると、`/Users/{user}/.jupyter/jupyter_lab_config.py`というファイルが作成されます。
2. ファイルの場所を参照し、エディターで開く
3. ファイル内で以下の行を検索します。`c.ServerApp.root_dir = ''`と書き換えてください。よくわからない場合は、自分のレポのディレクトリに設定してください（例：`c.ServerApp.root_dir = '/Users/{user}/repos'`）。行頭の`#`が削除されていることを確認してください。
4. パスには必ずフォワードスラッシュを使用してください。フォルダ名にスペースが含まれていても、二重引用符で囲まれていれば、バックスラッシュを使うことができます。
5. data-scienceディレクトリから`make jupyter-local`を再実行すると、ルートディレクトリが上記で指定したものに変更され

るはずです。

Dockerのメモリ割り当ての増加

デフォルトでは、**docker**はコンテナを実行するために**2GB**のメモリを割り当てます。**jupyter**や**python**ではデータがメモリ内に保持されるため、これでは十分な**RAM**が確保できない可能性があります。メモリ不足のエラーを避けるために、**docker**のメモリ割り当てを増やすことをお勧めします。

1. **Docker**のダッシュボードを開く。
2. 右上の歯車アイコンをクリックすると、設定が表示されます。
3. リソース」では、**Docker**が使用するメモリを追加で割り当てます。**8GB**を推奨しますが、大規模なデータセットを扱う場合はさらに増やす必要があるかもしれません。
4. **Docker**を再起動します。

Jupyter Extensionsのセットアップ

- **data-science**のリポジトリには、**git**、変数インスペクタ、折りたたみ可能な見出し、実行時間、システムモニタなど、多くの便利な**Jupyter Lab**拡張機能がプリインストールされています。
- これらを最大限に活用するために（そして、コンテナを実行するたびに設定しなくて済むように）、以下のファイルを作成します。`/Users/{user}/.jupyter/lab/user-settings/@jupyterlab/notebook-extension/tracker.jupyterlab-settings`
- そのファイルの中に、以下を貼り付けて保存します。

```
{
    "codeCellConfig":{
        "codeFolding": true,
        "lineNumbers": true,
    },

    "recordTiming": true,

}
```

興味深いライブラリーがあります。

データ/モデル解析

- **ELI5**
- **QuickDA**ビジュアラ

イゼーションツール。

- **Plotly**
- **Seaborn**

MLライブラリ

- **SKlearn**
- **Tensorflow**
- **Torch**
- パイアース（線形回帰、ロジスティック回
- 帰） プロフェット（時系列
- オートット（時系列
- **XGBoost** (強力なブラックボックス手法) 簡

単なコンカレンシー

- モディン
- **Dask** (セルフインストールが必要)

GPUスピードアップ

- プレーンML

---

layout: handbook-page-toc title:"Meltano At Gitlab"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

クイックリンク

Airflow{:.btn .btn-purple-inv} DataImageProject{:.btn .btn-purple-inv} GitLabDataUtilsProject{:.btn .btn-purple-inv} PythonGuide{:.btn .btn-purple-inv}。

メルタノ

Meltanoは、独自のKubernetesクラスターで、デフォルトの名前空間で運用しています。現在、3つのプロジェクトリポジトリが設置されています。このKubernetesクラスタはmeltano-masheyとしてGCPで稼働しています。

MeltanoのUIはインターネットに公開されていません。ログを見るには、kubernetesコンテナのログを見る必要があります。これは、「LOGS」タブ、または概要ページの「ワークロード」でmeltano-gitlabクラスタを選択することで見ることができます。

Meltanoについては、以下のリポジトリがあります。

		RepositoryDescriptionリポジトリの
1	Gitlab-data-meltano	彼のプロジェクトはインフラ関連のコードを持っています。`gitlab-app.yaml`と`meltano.yml`の設定。
2	メルトノ_タップス	TAPのソースコードを保持するプライマリリポジトリです。現時点では、以下のソースコードがあります。`TAP-XACTLY`と`TAP-ADAPTIVE`
3	tap-zengrc	tap-zengrcのソースコードを保持するプロジェクトです。

新しく開発されたタップについては、tap-zengrcのように、タップごとに新しいリポジトリを作成します。

使用している抽出器を更新するには、メインプロジェクトであるGitlab-data-meltanoの`meltano.yml`ファイルを更新します。変更がマージされたらgitタグを追加し、gitlab-app.yml kubernetes manifestを更新して新しいイメージを指すようにします。

Meltanoは内部でAirflowを使用しており、メタデータのデータベースにはCloud SQLを使用しています。メルターノのデータベース。

```
ローカルからKubernetesクラスタに接続（Google Cloud SDKがインストールされていることが前提）。このコマンドが動作しない場合は、GCPに接続し、KubernetesのClusterを選択し、connect to clusterを選択してください。最新のコマンドが表示されます。
gcloud container clusters get-credentials meltano-mashey --zone us-west1-a --project gitlab-analysis
```

現在の設定では、クラスターに以下のシークレットが定義されています。

- **cloud-sql**
- **meltano-db**
- **airflow**
- **TAP-**
- **SECRET**
  AIRFLOW-

DB

秘密ファイルの作成

```
kubectl create secret generic cloud-sql
kubectl create secret generic meltano-db
kubectl create secret generic airflow
kubectl create secret generic tap-
secrets kubectl create secret generic
airflow-db
```

編集の秘密

```
kubectl edit secrets cloud-sql
kubectl edit secrets meltano-db
kubectl edit secrets airflow
kubectl edit secrets tap-
secrets kubectl edit secrets
airflow-db
```

コンテナへの格納

```
kubectl exec -it gitlab-production-5dd4c79694-vwwtm -c tap-actly /bin/bash
```

注意事項ポッド名は変更される可能性がありますので、UIまたは`kubectl get pods`コマンドで確認してください。

```
# 更新されたマニフェストの適用には、名前空間は必要ありません
kubectl apply -f gitlab-app.yaml
```

```
# 名前空間のデプロイを削除する kubectl
                delete-f gitlab-
app.yaml
```

GKEでのMeltanoセットアップのウォークスルーのビデオを見る

メルトノの設定に、TAPで使用するConfig変数を追加。

コンフィグをMeltanoのタップに渡す必要があるときはいつでも、以下の5つのアクティビティを行う必要があります。以下の5つのアクティビティを行う必要があります。

1. **ユニークな変数名を決め、それらの変数値をクラスタ内の`tap-secrets`という名前のKubernetesのシークレットに追加します。シークレットファイルを編集するコマンドは、`kubectl edit secret tap-secrets -o yaml`です。**
2. **これらの変数を `gitlab-app.yaml` ファイルに以下のように参照して追加します。**

```
 # Kubernetes Secrets::zengrc
        - name: ZENGRC_USERNAME # 環境で一意になるようにする
          valueFrom:
            secretKeyRef:
              name: tap-secrets # これは、秘密を追加するための秘密の名前で、存在するどのような秘密ファ
イルでもよい。
              キーを使用します。ZENGRC_USERNAME # シークレットファイルでユニークにしておく。
        - name: ZENGRC_PASSWORD # 環境で一意になるようにする
          valueFrom:
            secretKeyRef:
              name: tap-secrets # これは、秘密を追加するための秘密の名前で、存在するどのような秘密ファ
イルでもよい。
              キーを使用します。ZENGRC_PASSWORD # 環境に合わせてユニークに保つ
```

3. ファイルを修正した後、新たに用意した配置ファイルをクラスタに適用する必要があります。これを行うには、既存のデプロイメントを削除して、新しいデプロイメントを適用する必要があります。以下のコマンドを使用して、この作業を行います。

- `kubectl delete -f gitlab-app.yaml`で既存のデプロイメントを削除します。
- `kubectl apply -f gitlab-app.yaml` で新しいデプロイメントを適用します。

4. 設定ファイル（`meltano.yml`）の中で、これらの定義された変数を以下のように参照します。

```
config:
  ユーザー名: $ZENGRC_USERNAME パス
  ワード: $ZENGRC_PASSWORD
```

このようにする理由は、ユーザー名とパスワードは環境内で一意のキーではなく、他のタップでも使用されているため、正しいTAPS設定を渡すために、lubeの秘密変数名への参照を渡します。5) 最新の`meltano.yml`ファイルをコンテナにコピーします。
`kubectl cp meltano.yml default/<**pod-name**>:/projects`

すべてのタップのスケジュールを実行して、問題がないことを確認します。

meltano.ymlファイルへのタップの追加

現在の設定では、MRを作成してタップ情報をmeltano.ymlファイルに追加しています。必要な情報が meltano.yml ファイルに追加されたら、以下の手順で TAP を有効にしてください。以下は、各TAPに追加する必要があるRequire情報のサンプルです。

```
## The Name of the TAP under plugins:-- > エクストラクタ

- 名前：tap-zengrc 名前空
    間：tap_zengrc
    pip_url: git+https://gitlab.com/gitlab-data/tap-zengrc 実行ファイ
    ル: tap-zengrc
    の能力を発揮します。
    - の設定を
    発見しまし
    た。
    - 名前: ベースURL
    - 名前：ユーザー名
    - name:
    password
    config:
      base_url: https://gitlab.api.zengrc.com/api/v2
## スケジュールセクションのスケジュールの下にあるタッ
プのスケジュール。
- name: zengrc-to-snowflake
  extractor: tap-zengrc
  loader: target-snowflake
  transform: skip
  interval: '@daily'
  start_date: 2021-07-13
```

その後、以下の手順でmeltano.ymlを実行中のコンテナにコピーします。

- 必要なクラスタに接続します。`gcloud container clusters get-credentials meltano-mashey --zone us-west1-a --project gitlab-analysis kubectl exec -it gitlab-production-5f8fd9ccb-npvxl -c tap-xactly` コマンドを使
- って、現在アクティブなクラスタである `meltano-mashey` に接続します。
  `/bin/bash`を使用しています。注:- ポッド名が変更されている場合がありますので、正しいポッド名を取得するには`kubectl get pods`を使用してください。
- 修正したmeltano.ymlをローカルからコンテナにコピーする `kubectl cp meltano.yml default/gitlab-production- 5f8fd9ccb-npvxl:/projects`
- 最初にschedule `meltano schedule run zengrc-to-snowflake`を実行してみると、インストールを求められます。

抽出器を使ったセッションでも、以下のようなエラーが発生しました。

```
    メルト            | Running extract & load...
    ノ                | ELTを完了できませんでした。Cannot start extractor:実行可能な'tap-zengrc'
メルト
が見つかりませんでした。エクストラクタ 'tap-zengrc' が `meltano install' を使ってまだインストールされて
いない可能性があります。
抽出器 tap-zengrc`、または実行ファイル名が間違っている可能性があります。
    ELTを完了できませんでした。Cannot start extractor:実行可能な 'tap-zengrc' が見つかりませんでした。
meltano install extractor tap-zengrc`で抽出器'tap-zengrc'がまだインストールされていないか、実行ファイ
ル名が間違っている可能性があります。
```

- エキスパンダーをインストールした記事

```
root@gitlab-production-5f8fd9ccb-q6gt4:/projects# meltano install extractor tap-zengrc
Installing 1 plugins...
エキスパンダー「tap-zengrc」のインストール...エ
キスパンダー「tap-zengrc」のインストール
```

- その後、コマンド `meltano schedule run zengrc-to-snowflake` を再実行します。これで **snowflake** にデータがプッシュされるようであれば、**TAP** は期待通りに動作しています。

---

layout: handbook-page-toc title:"Sisense For Cloud Data Teams" description:"GitLabにおけるSisense For Cloud Data Teams"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

{::options parse_block_html="true" /}。

クイックリンク

Sisense{:.btn .btn-purple-inv} dbtdocs{:.btn .btn-purple-inv}。

アクセス

Sisenseは、当社のエンタープライズ標準のデータ可視化アプリケーションであり、当社のエンタープライズデータウェアハウスへの接続が承認されている唯一のアプリケーションです。

セルフサービス・ダッシュボードへのアクセス

1. **GitLab**の全員に、**Sisense**の**View-only**ライセンスが提供されています。
2. 表示のみ」では、キーレビューで定期的に確認されるものを含む、機密性のないダッシュボードにアクセスできます。
3. **Okta**で**Sisense**にログインします。

セルフサービスのダッシュボード開発

1. **Sisense**のレポートやダッシュボードを開発するには、エディターライセンスが必要です。エディターライセンスはお金がかかりますので、リクエストする前に必要性を確認してください。私たちの倹約の価値に留意してください。
2. エディターを効果的に使うには、ちょっとした技術やデータモデリングの経験が必要です。
3. 正しいユーザーの役割を確認するなど、アクセスリクエストを作成します。

ライセンスの再取得

1. データチームは、定期的にライセンスを見直し、**90**日間使用されていないライセンスを回収しています。
2. **View-only**や**Editor**のライセンスが不要になった場合は、**DataTeamIssue**を作成していただければ、あとは私たちにお任せください。

トレーニングリソース

セルフサービス・ダッシュボードへのアクセス

- **Sisense（GitLab内部）製品ドキュメント**
- **へのアクセス**
- **はじめに**
- **はじめに - ビューア**

セルフサービスのダッシュボード開発

- **製品ドキュメント はじめに**
- **GettingStarted-セルフサービス・ダッシ**
- **ュボード GitLab'sSisenseProject**
- **GitLabのSisenseEditorトレーニング（GitLab社内）**
- **SisenseDataCommunity**
- **Sisense Plotlyテンプレート**

Sisenseへのアクセス

GitLabの全員がSisenseへの閲覧のみのアクセス権を持っています。Oktaを使ってログインしてください。自分のチャートを作成するエディター権限など、高度なアクセスが必要な場合は、アクセスリクエストを作成してください。ユーザーロールのセクションもご覧ください。

Sisenseリソース

- **Sisenseプロジェクト**
- **Sisenseトレーニング(GitLab社内)**
- **Sisenseエディタトレーニング(GitLab社内)**
- **Sisenseデータオンボーディング。チャート/ダッシュボードの作成と分析**
- **Sisenseコミュニティ**

セルフサービスのダッシュボード開発

このセクションの目的は、読者がGitLabのデータに関する疑問に答える独自のSisenseダッシュボードを構築できるようにすることです。最後に紹介する例は、製品組織に特化したものですが、GitLabの他のチームにも一般化できます。

Sisenseの基礎知識

独自のSisenseダッシュボードを構築する最初のステップは、正しい権限を持っているかどうかを確認することです。

Oktaを使ってSisenseにログインすると、右上にNew Chartボタンが表示されるはずです。何も表示されていない場合は、表示専用のアクセス権しかないので、上記の手順で編集者のアクセス権を取得してください。

New Chart」が表示されるようになったら、自分のダッシュボードを作成してみましょう。左側のナビバーでDashboardsを見つけ、+アイコンをクリックしてダッシュボードを作成します。ディレクトリを整理するために、必ず名前を付けてください。

ダッシュボードの構築と名前の決定が完了したら、右上の「New Chart」をクリックして、チャートの追加を開始できます。これで、クエリを書き始める準備が整いました。

適切なデータソースを見つける

データに関する質問に答えるための次のステップは、クエリを実行するための関連テーブルを見つけることです。そのためには、Snowflakeデータウェアハウスとそこに供給されるデータソースの背景を知る必要があります。Snowflakeに保存するデータには、一般的に3つのタイプがあります。外部、内部フロントエンド、内部バックエンドです。

外部データ

外部データとは、**GitLab**で使用しているサードパーティのソフトウェアによって生成されたすべてのデータのことですが、本番データを自分たちで保存しているわけではありません。外部データには、**Salesforce**、**Zuora**、**Netsuite**、**Greenhouse**、**BambooHR**などがあります。これらのデータは、**API**を使用してデータウェアハウスにロードします。

内部のバックエンドデータ

GitLab.comは**Ruby on Rails**のアプリで、バックエンドに**Postgres**データベースを使用しています。**GitLab.com**のユーザーが新しい**MR**、課題、コメント、マイルストーンなどを作成するたびに、データベースに新しい行が作成されます。データチームはカスタム**ELT**を作成し、これらの**Postgres**テーブルをデータウェアハウスに同期させています。

自己管理型のインスタンスでは、**Usage ping**を使ってこれらのバックエンドデータベースの匿名化されたサマリーを毎週取得するようにしています。

内部のフロントエンドデータ

さらに、**gitlab.com** のフロントエンドのインタラクションを追跡するために **Snowplow** というツールを有効にしました。**Snowplow** は、ページビューの自動追跡に加え、フォームやリンククリックの追跡も行います。**Snowplow** は、ユーザーのセッションやブラウザに関する情報を含むメタデータを、すべてのイベントとともに送信します。

注：**Snowplow**はバックエンドのイベントを捕捉することもできますが、現時点では主に**javascript**（フロントエンド）のトラッキングに使用しています。**{: .alert .alert-info}**。

フロントエンドデータとバックエンドデータの違いは何ですか？バックエンドデータとは、アプリケーションにとって何らかの目的を果たすために、すでにアプリケーションのデータベースに保存されているデータのことです（**MR**、課題、パイプライン）。それに対して、フロントエンドのトラッキングの主な目的は分析です。

****dbt**ドキュメント** **{: .panel-heading}**。
私たちの**[dbt Docs site](https://dbt.gitlabdata.com/)**には、**snowflake**でクエリを実行できるすべてのテーブルがリストアップされています。これらの多くは、テーブルとカラムの両方のレベルで文書化されており、クエリを書くための素晴らしい出発点となっています。

発見データセット

データディスカバリーは**Sisense**の機能で、**SQL**のスキルが低いユーザーでも、ドラッグ＆ドロップのインターフェイスで特定のデータセットのビジュアライゼーションを作成することができます。

現在、この機能の価値を理解するためにテストを行っています。社内の誰もがアクセス可能なテストディスカバリーテストをいくつか作成し、私たちの支援なしにチャートを構築できるようにしました。

GitLab.com 利用データイベント

このデータセットは、ここに書かれているモデルに基づいて作成されています。ユーザーは、特定の**namespace_id**によって実行されたすべてのイベントと、その**namespace**に関する追加のメタデータを見つけることができます。

このデータセットでどのような質問に答えることができますか？

- 名前空間の作成後、最初の**14**日間に実行されたイベントの平均数 名前空間の作成後、最
- 初の**14**日間に発見されたステージの平均数 コホートビュー
- 毎日どれだけの**CI**パイプラインが実行されているのか？
- 毎月どれだけの有料ネームスペースが新しいマイルストーンを作っているのか？

実現可能なビジュアライゼーションの例を含むダッシュボードを作成しました。このデータセットにア

クセスするには、主に**2**つの選択肢があります。

- ダッシュボード上で編集者権限が与えられている場合は、右上の「新規チャート」をクリックすることができます。 また、左
- メニューのコンパスのアイコンをクリックすると、新しい探査を開始することができます。

チャートエディターが開いたら、以下のように「発見」ボタンをクリックします。

リストメニューから`gitlab_dotcom_usage_data_events`というデータセットを探します。

例

Question 1:ページビューについて質問しているので、この質問に答えるために内部のフロントエンドデータを使用できます。以下のように**Snowplow**のページビューを照会することができます。

```
```

このクエリを**Sisense**の**SQL**エディタで実行すると、クエリの下のチャート部分に表が出力されます。そこから、**Sisense**はデータを視覚化するための様々なオプションを提供します。チャートの構築について学ぶに、**Sisense**の**10**分間**DataOnboarding**ビデオを見るのが良いでしょ

これを見ると、毎月約**11**万人のユーザーがマージリクエストを作成していることがわかります。

質問 3: 過去 30 日間に 1 つのトライアルフォームから別のフォームに変換した「ユーザー」の数は？(Conversion Funnel) snowplow CTE を使用して、**2** つのステップを別々に照会し、次にそれらを結合することができます。

```
WITH first_trial_form AS
  ( SELECT)
    user_snowplow_domain_id AS user_id,
    min_tstamp::DATEAS day_of,
    min_                    tstampAS sent_at
  FROM analytics.snowplow_page_views_30 -- 必要に応じて、`_30`を`_all`に変更してください。カスタムの日付範囲の場合は、derived_tstampでフィルタリングします。`WHERE derived_tstamp BETWEEN '2019-10-01' AND '2019-12-01'
  WHERE page_url_path = '/-trials/new'
),
second_trial_form AS
  ( SELECT
    user_snowplow_domain_id AS user_id,
    derived_tstamp::DATEAS day_of,
    derived_                tstampAS
    sent_at
  FROM analytics.snowplow_unstructured_events_30
  WHERE event_name = 'submit_form'
    AND page_url = 'https://gitlab.com/-/trials/apply'
)
SELECT
  first_trial_form.day_of,
  COUNT(DISTINCT first_trial_form.user_id)    AS "View first trial form",
  COUNT(DISTINCT second_trial_form.user_id)AS "Submit last trial form",
  COUNT(DISTINCT second_trial_form.user_id) * 100 / COUNT(DISTINCT first_trial_form.user_id)
AS "Pct"
FROM first_trial_form
  LEFT JOIN second_trial_form
    ON first_trial_form.user_id = second_trial_form.user_id
    AND first_trial_form.day_of = second_trial_form.day_of
    AND first_trial_form.sent_at <= second_trial_form.sent_at
GROUP BY 1
ORDER BY 1
```

質問 4: /merge_requests ページでは、どの「ステータス」タブ **('Open', 'Merged', etc.)** がクリックされますか？以前と同様に、スノープラウデータを使って**merge_requests**ページのクリック数とページビューを測定することができます。

```
WITH link_clicks AS
  ( SELECT
    TRY_PARSE_JSON(unstruct_event): "data": "data": "elementId"::VARCHAR AS element_id
  FROM analytics.snowplow_unstructured_events_30
```

```
  WHERE event_name = 'link_click'
    AND element_id IN ('state-closed', 'state-all', 'state-opened', 'state-
    merged') AND page_url_path LIKE '%/merge_requests'
)

SELECT
  element_id AS "Tab Name",
  COUNT(*)                                                AS "Total Clicks",
  COUNT(*) / (SELECT COUNT(*) FROM link_clicks) * 100 AS "Percent of
Clicks" FROM link_clicks
GROUP BY 1
ORDER BY 1
```

タブ名	総クリック数	クリック数の割合
ステートマージド	**200923**	**65.304300**
ステートオープン	**50291**	**16.345700**
ステート・クローズド	**31957**	**10.386700**
ステート・オール	**24501**	**7.963400**

この**4**つのタブのクリック数のうち、**65%**が「統合」タブへのクリックです。

これらのタブがクリックされる頻度を、*マージリクエストページの総ページビュー数に対する割合として*知りたい場合は、このクエリにいくつかの変更を加えます。

```
WITH link_clicks AS (
  SELECT
    TRY_PARSE_JSON(unstruct_event): "data": "elementId"::VARCHAR AS element_id,
    COUNT(*)                                                      AS total_clicks
  FROM analytics.snowplow_unnested_events_30
  WHERE event_name = 'link_click'
    AND element_id IN ('state-closed', 'state-all', 'state-opened', 'state-
    merged') AND page_urlpath LIKE '%/merge_requests'
  GROUP BY 1
),

page_views AS
  ( SELECT
    COUNT(*) AS total_views
  FROM analytics.snowplow_unnested_events_30
  WHERE event = 'page_view'
    AND page_urlpath LIKE '%/merge_requests'
)

SELECT
  link_clicks.element_id                                    AS "Tab
                                                            Name",
  link_clicks.total_clicks / page_views.total_views * 100 AS "ページビューの割合"
FROM link_clicks
  INNER JOIN
    page_views ON 1=1
```

タブ名	ページビューの割合
ステートマージド	**5.423417**
ステートオー	**1.357480**

プン

ステートクローズド	**0.862600**

タブ名	ページビューのパーセ
ンテージ **state-all0.661344**	

マージリクエストページのページビューのうち、「マージ」タブをクリックしたのは5%でした。

Sisenseのチャートパフォーマンス

Sisenseチャートがタイムアウトしたり、実行に時間がかかる場合は、Sisenseチャートを生成するためのSQLクエリをリファクタリングして、クエリを最適化する必要がある場合がほとんどです。データチームのプロジェクトで課題を作成してください。

ダッシュボードの公式バッジ

Sisenseの一部のダッシュボードには、公式バッジ（TwitterのVerified Checkmarkに似たもの）が表示されます。  **Sisense 検証済みチェックマーク**

これは、これらの分析がデータチームによってクエリの正確性が確認されたことを意味します。検証済みのチェックマークが付いていないダッシュボードは、必ずしも不正確なものではなく、データチームによるレビューが行われていないだけです。公式バッジを追加・削除できるのは、Dataロールのメンバーのみです。

スペース

Sisenseのスペースは1つです。

- **GitLab**

この2人は、それぞれ`periscope`と`periscope_sensitive`という異なるユーザーでデータウェアハウスに接続します。

ほとんどの作業はGitLabスペースで行われますが、一部の*非常にセンシティブな分析*はGitLabセンシティブに限定されます。その例としては、契約者や従業員の報酬に関わる分析や、匿名化されていないインタビューデータなどが挙げられます。

スペースはタグで整理されます。タグは機能（プロダクト、マーケティング、セールスなど）とサブ機能（クリエイト、セキュア、フィールドマーケティング、EMEA）に対応している必要があります。*タグは*課題のラベルと大まかに一致している必要があります（優先順位はつけません）。タグは無料です。人々が探している情報をできるだけ簡単に見つけられるようにしてください。現時点では、タグの削除や名前の変更はできません。

ダッシュボードを自動的にSlackにプッシュする

例えば、製品担当者は、製品の理由で失われた機会についての最新情報を毎週知りたいと思っているでしょう。この情報を定期的にSlackに流すことが望ましい場合は、Slackのネイティブな`/remind`を利用してURLを印刷することができます。ダッシュボードが自動更新されていないように見える場合は、Sisense管理者に連絡して更新スケジュールを更新してもらってください。

ユーザーの役割

{:#user-roles}

Sisenseには、admin、SQL、View Onlyの3つのユーザーロール(アクセスレベル)

があります。Sisenseのライセンスの現在の状況は、分析プロジェクトで確認でき

ます。

****Sisenseのユーザーを更新する** {: .panel-heading}。**

- **コンソールにjqueryを注入する（StackOverflowより）。**

```
let jq = document.createElement("script");
jq.src = "https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js";
jq.onload = function() {.
  コードを入力してください。
};
document.body.appendChild(jq)です。
```

- DOMで表示されるリストから全ユーザーのリストを取得するには、コンソールでこれを実行し、クラス名を実際のラベルに置き換えます。

```
$('div.list_class_namer_replace_this').map(function(i, el) {
  return $(el).text()})
).toArray()
```

アドミニストレーター

これらのユーザーは、新規ユーザーのプロビジョニング、パーミッションの変更、データベース接続の編集などの機能を持っています。(典型的な管理者の仕事)

リソースアドミンオンボーディング

編集者アクセス

ユーザーは、チャートやダッシュボードの基盤となるアナリティクスデータベースの `analytics` スキーマおよび `analytics_staging` スキーマに対して SQL クエリを書くことができます。また、これを容易にするために、SQLスニペットを作成または利用することもできます。SQLアクセスライセンスの数には限りがありますので、現時点では、ディレクターを中心としたチームごとに1つに制限することを目指しています。チーム内でSQLアクセスを持つ最適な候補者を決定するのは、ディレクターに任されています。

閲覧のみのユーザー

これらのユーザは、既存のすべてのダッシュボードを利用できます。ダッシュボードのフィルタを変更することができます。最後に、ドリルダウン機能を利用してダッシュボードを掘り下げることができます。

部門別ユーザー

エディターロールをさらに細分化した追加のロールがあります。特定のチャートは、誰も編集できないようにします。例えば、「FinanceKPIs」ダッシュボードは、「Data」および「Finance」ロールのメンバーのみが編集できるようにします。

ユーザーをプロビジョニングする際の注意点

すべてのユーザーは、Oktaを介して表示のみのアクセス権限を持っています。

ユーザーをアップグレードするには、Sisense UIで「ロールとポリシー」のセクションに移動します。そして、ユーザーを関連するグループ（Admin/Editor）とその部門（例：Marketing、Productなど）または部署（例：UX、Securityなど）に追加します。

ユーザーは、どのグループに属していても、最高のアクセス権を継承します。これが、すべての機能がデフォルトで「表示のみ」となっている理由です。

新しいダッシュボードの作成とレビューのワークフロー

このセクションでは、Sisenseでダッシュボードを「本番」にプッシュする方法のワークフローを詳細に説明します。現在、MRファーストのワークフローを持つ機能はありません。このワークフローは、社内のすべてのダッシュボードの品質を高いレベルで確保することを目的としています。ダッシュボードのビジュアル、SQL、Python、UXがデータチームのメンバーによってピアレビューされ、ハンドブックに詳述されている基準を満たしていれば、ダッシュボードは本番に間に合います。

1. `WIP:`を名前に持つダッシュボードを作成し、`WIP`トピックに追加する
2. dbtやウェアハウスのドキュメントを活用して、クエリやチャートを構築する
3. ダッシュボードをレビューする準備ができたら、Sisenseダッシュボードレビューテンプレートを使用して、データチームプロジェクトにMRを作成します。何も変更する必要はなく、これは空のMRであるべきです。
4. テンプレートの指示に従う
5. データチームのメンバーにテンプレートを割り当て、レビューしてもらいます。誰にメールを送ればいいかわからない場合は、`@gitlab-data`を使ってください。
6. すべてのフィードバックが与えられ、適用されたら、データチームのメンバーは右上のテキストタイルを更新し、ダッシュボードの作成者とレビュー者、最終更新日、関連する問題のクロスリンクを詳細に表示します（詳細は「データ分析プロセス」を参照してください）。

7. データチームメンバーのレビュアーは
    - ダッシュボードの名前を変更して「`WIP:`」ラベルを削除する

- `WIP`トピックからダッシュボードを削除する
- ダッシュボードに承認バッジを追加する
- 権限があればMRをマージするか、マージ権を持つ人に割り当てる
  - MRは、意味のある変更がない場合にも閉じることができます。その場合、データチームのレビュアーは、MRを閉じる前に「`Approved`」というラベルを追加します。

既存のダッシュボードのワークフローの更新

このセクションでは、データ・チームのピアレビュー・プロセスを経た既存のダッシュボードを更新する方法についてのワークフローを説明します。ダッシュボードが運用されると、新しいダッシュボードの作成とレビューのワークフロー全体を経ることなく、データアナリストとDRI/優先順位付けの所有者がダッシュボードへの段階的な追加を実施することができます。既存のダッシュボードを更新するには、以下の手順に従ってください。

1. 更新中のチャートのタイトルに`WIP:`を追加します。新しいチャートが追加されている場合は、タイトルに`WIP:`を追加します。
2. 新しいダッシュボードの作成とレビューのワークフローで定められているSQLスタイルガイドとSisense可視化のベストプラクティスに従って、既存のチャートに変更を加えるか、新しいチャートを追加します。
3. ダッシュボードをDRI/優先順位付け所有者がレビューする準備ができたら、Sisenseダッシュボードレビューテンプレートを使用して、データチームプロジェクトでMRを作成します。DRI/優先順位付け所有者に、MRの上部にある自分のセクションを記入してもらいます。データアナリストは、「既存のダッシュボードの更新」セクションのみを記入します。
4. ビジネス・ステークホルダー/DRIが更新をレビューして承認し、既存のダッシュボードへの更新チェックリストが完了したら、新規または更新されたチャートから「`WIP:`」を削除して、MRを閉じることができます。

マージリクエストによるダッシュボード、チャート、スニペット、またはビューの追加または更新

また、GitLab Data - Periscope Projectへのマージリクエスト（MR）により、チャート、スニペット、ビューを修正することができます。次の例をご覧ください。 例 または以下の手順に従ってください。

1. MRをGitLabDataに提出 -Periscopeプロジェクト
2. MRをスニペット/ダッシュボード/チャート/ビューのオーナーに割り
   - 当てる Periscopeプロジェクトでのスニペットオーナーの確認
   - 例 Sisenseでのスニペットオーナーの確認例
3. オーナーは "Approve "ボタンで承認する必要があります。
4. cc @gitlab-data
5. マージ権限を持つGitLab Dataチームのメンバーが、それをマー

ジします。 プロジェクトのリポジトリからの同期は双方向です。

ダッシュボードの自動更新を依頼する

データチームのプロジェクト課題を作成することで、ダッシュボードの自動更新を要求することができます。課題テンプレートを使用して、1つの特定のダッシュボードまたはダッシュボードの一括リストについて、指定された間隔での自動更新を要求できます。

Sisenseのチャートをハンドブックに埋め込む

これらのチャートをハンドブックに埋め込むのは、データチームではなく、ビジネスユニットの責任です。Sisenseには素晴らしい埋め込みドキュメントとアプリを通じたチャットサポートがあります。ハンドブックにチャートやダッシュボードを埋め込むには、主に3つの方法があります。

ハードコードされたHTML

HTMLを受け入れられるファイルタイプであれば、いつでもHTMLをハードコードすることができます。`.html`ファイルがそのわかりやすい例です。しかし、Markdown（`.md`）や埋め込みRuby（`.erb`）ファイルでは、通常のHTMLにフォールバックすることができます。

ダッシュボード

ダッシュボード全体をハンドブックに埋め込むのとても簡単ですダッシュボードを埋め込むに、まずそのダッシュボードを外部共有ダッシュボード」にする必要があり。次にURL文字列に`?embed=true`を追加して、埋め込みリンクにしますそのURLを以下のように

差し込みます。

```

```

```
<iframe class="dashboard-embed" src="https://app.periscopedata.com/shared/string-of-numbers-here?
embed=true" height="700"> </iframe>。
```

SAFEスペースからダッシュボードを使用する場合、外部共有ダッシュボードを作成すると、リンクを持っている誰もがダッシュボードを利用できるようになるので、推奨しない。ハンドブック内でダッシュボードをスクロールする必要がないようにすることを目的としているため、高さの値を適切に調整する必要があります。プログラムでこれを行う方法はありません。

チャート

ハンドブックの埋め込みチャートは、常に`signed_periscope_url`ヘルパー関数を使用して生成する必要があります。この関数は、データチームのメンバーが手伝わなくても、自動的に署名された**URL**を生成してくれます。これは、**Sisense API**にさまざまなデータオプションを渡す実験をするときに特に便利です。このヘルパー関数は、**SAFE**スペースからチャートを埋め込もうとするとエラーを返します。**SAFE**スペース内のチャートへの**URL**のみを共有することを推奨する。

ファイルは `.erb` で終わる必要があります。`index.html.md`という名前のファイルで作業している場合は、`.erb`を追加するだけで、ファイル名が`index.html.md.erb`になります。

データを引数として関数に渡すだけです。サブアレイやオブジェクトなど、**Sisense API**で必要とされるあらゆるデータを取ることができます。**Sisense**のドキュメントには、**embedAPI**で利用可能なオプションの全リストがあります。

```
<embed width="100%" height="400px" src="<%= signed_periscope_url(chart: 6114177, dashboard: 463858, embed:
'v2' ) %>">。
```

この方法は、プレーンな**Markdown**や**HTML**ファイルではレンダリング時にコードが実行されないため、機能しません。

ヒント：必要な**PERISCOPE_EMBED_API_KEY**が**CI**変数として設定されているため、埋め込みチャートはローカルには表示されません。グラフが正しく表示されることを確認するには、**MR**内でレビューアプリを起動してください。

パフォーマンス指標 YML

`data/performance_indicators.yml` は、パフォーマンス・インジケータのコンテンツを載せたハンドブック・ページを、規約に従って自動的に生成するシステムの基礎となるものです。オブジェクトに `periscope_data` プロパティとサブ値を与えると，テンプレートは自動的に署名付きの **URL** を生成して **HTML** を書いてくれます．これは，上記と同じ `signed_periscope_url` ヘルパー関数を使用します。このヘルパー関数は、**SAFE**空間のチャートを埋め込もうとするとエラーを返します。**SAFE**スペース内のチャートへの**URL**のみを共有することをお勧めします。

```
- 名称：MRレート
  の説明を参照してください。MR率は、開発エンジニアが平均してどのくらいのMRをこなしているかを毎月評
価するものです。
  periscope_data：
    チャート。
    6114177
    ダッシュボード463858 エ
    ンベッド：V2
  is_key: true
```

CSVやGoogle SheetsでSisenseにデータを追加する

CSVを使って**Sisense**にデータをアップロードする方法はいくつかあります。これらの方法はすべて文書化されており、データチームのプロジェクト課題を作成してリクエストすることができます。リンクをクリックして選択肢を確認してください。

Sisenseで**CSV**をアップロードする場合は、`New Chart`または`New Exploration`ウィンドウで`Snowflake`データベースを使用していることを確認してください。そうでない場合は、**GitLab**の内部データモデルにアクセスするために、常に`GitLab_(Use_this_one!)`をデフォルトにする必要があります。**Sisense**でデータベースを変更する方法については、以下の画像をご覧ください。**Sisense**のデータベース

SisenseのCSVアップロード機能を使って、個人情報や機密データをSisenseにアップロードしないようにしてください、このデータはすべてのGitLabチームメンバーが一般にアクセスできます。

Sisenseからのデータの取り出し

Sisenseからデータを取り出す必要がある場合もあるでしょう。これが単発のケースであれば、いつでもUIからCSVをダウンロードすることができます。ただし、CSVダウンロードの最大サイズは500MBで、クエリは4分以内に完了しなければならないことに注意してください。このデータを定期的にシートなどに取り込む必要がある場合は、「編集」→「チャートフォーマット」→「詳細」→「CSVの公開URLを公開」でCSVの公開URLを公開することができます。シートでは、`=importdata("PUBLICURLHERE")`を使用できます。

エクスポートしようとしているデータに機密性がなく、サイズが500MB以上で100万行以下の場合は、データチームのプロジェクト課題を作成します。

テキストファイルを操作するツールが必要な場合は、構造化されたテキストデータを操作するためのコマンドラインツールのリストをご検討ください。 これらの素晴らしいリソース.

ヒントとコツ

正しい会計年度の情報のみを表示するダッシュボードの設置

現在の会計四半期または次の会計四半期にのみフィルタリングするダッシュボードが必要な場合があります。Sisenseの既製の日付フィルターでは、カスタムの会計年度に対応できません。

あなたの分析では、次のように追加します。(`datevalue]`をフィルタリングしたい日付で更新してください)

```
LEFT JOIN analytics.date_details on current_date =
date_actual WHERE [datevalue] < last_day_of_fiscal_quarter
AND [datevalue] > first_day_of_fiscal_quarter
```

ダッシュボードのクエリで当月を除外する

ほとんどの場合、クエリから現在の月を除外し、完了した月のみをレポートする必要があります。当月は不完全なので、これらの数字を表示すると誤解を招く恐れがあります。ダッシュボードのクエリで以下のステートメントを使用して、当月を除外してください。

```
WHERE <month_column> < date_trunc('month', CURRENT_DATE)
```

タイムゾーン

SnowflakeのタイムスタンプはすべてUTCでなければなりません。

Sisenseの表示時間はPT（太平洋時間）に設定されています。これは、通信のガイドラインに沿ったものです。

`created_date=daterange]`を使用する場合、Sisenseは`current_timestamp`を使用し、PTに変換して比較します。例えば、10月4日の13:00PT（20:00UTC）に、過去3日間のデータをリクエストした場合、Sisenseは2019-10-02 07:00:00.000から2019-10-05 07:00:00.000までのフィルターを作成します。これらの時間はUTCで、PTの2019-10-02の開始時の午前0時とPTの2019-10-04の終了時の午前0時に対応し、つまりPTの丸3日間です。データベースがUTCで値を保存している場合（私たちはそうしています）、この比較はまさにあなたが望むものです。

エンドユーザーが気にするべきことは、チャートに表示するために日付をフォーマットすることです。タイムスタンプをPTに変換するには、`[created_at:pst]`という構文を使用します。また、次の構文を使って日付に変換することもできます。`:日付を[created_at:pst:date]`のように指定します。これは、Salesforceなどのソースシステムの日付とSisenseで表示される日付を比較する際に必要です。

覚えておくべき重要なことは

- **特に指定や宣言がない限り、データベースやクエリの日付はすべて**UTC**です。**

- **Sisense**の**daterange**フィルターは、太平洋時間に基づ

いています。 日付範囲フィルターの使い方

ダッシュボードのフィルターとして使用したい集約された日付がある場合は、集約された期間
を日付範囲の開始値とし、集計の終了日より**1**日少ない値を日付範囲の終了値とします。日付範囲の開始値は、日付期間にマッピングすることができます。

DRS

日付範囲の終了日については、集計フィルタで選択した値に基づいて終了日を自動的に計算するために、クエリに追加のカラムを作成する必要があります。仮に、`sfdc_opportunity_xf.close_date`を気になる日付として使っていた場合、以下のような例になります。 `dateadd(day,-1,dateadd([aggregation],1, [sfdc_opportunity_xf.close_date:aggregation])) as date_period_end` 次に、日付範囲の終わりのマッピングを追加します。

 **DRE**

特定のモデルが照会される場所を見つける

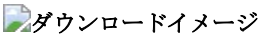
チャートやスニペットの生成に使用されるすべてのクエリは、**Sisense**プロジェクトの**periscope/master**ブランチにあります。検索フィールドにモデルや興味のあるキーワードを入力すると、関連するクエリを見つけることができます。

不明瞭な視覚化を避ける

円グラフは、データを視覚化する方法としては、一般的にはあまり良くないとされています。このブログでは、円グラフを使ってはいけない理由を紹介しています。

高画質な画像の輸出

Sisenseから静的なチャートをエクスポートする際には、スクリーンショットを撮るのではなく、内蔵のエクスポート機能を使用してください。エクスポートすると、背景が透明な高品質の画像が作成されます。**Persicope**から画像をエクスポートするには、任意のチャートの右上にある「その他のオプション」を選択し、「画像のダウンロード」を選択します。

 ダウンロードイメージ

Sisense Housekeeping

Sisenseは私たちのビジネス・インテリジェンス・ツールであり、**Single Source of Truth**（真実の情報源）として機能しています。私たちの**SSOT**として、**Sisense**は非常に高いレベルの清潔さ、整頓、正確さを維持する必要があります。

また、データチームが作成・承認したダッシュボードが正確で有益なものであることも必要です。また、データチームのメンバーが定期的にメンテナンスを行うことも必要です。

大原則。

- 承認されたダッシュボードはすべて機能し（チャートがすべて表示され）、正確なデータを表示しなければなりません。
- データチームのメンバーが作成した**WIP**ダッシュボードはすべて**Work in Progress**でなければなりません。これは、プロジェクトがまだ進行中であり、チームの誰かがこのダッシュボードのリリースをまだ計画していることを意味します。
- 空のダッシュボードは持たない タイ
- トルのないダッシュボードは持たない
- **SQL**エラーが発生したダッシュボードは保存しません：エラーを修正するか、ダッシュボードをアーカ

イブします。 未使用のダッシュボードの自動アーカイブ

現時点では、すべてのダッシュボードについて、自動アーカイブ機能が有効になっています。つまり、ダッシュボードがしばらく表示されないと
45日以上経過すると、自動的にアーカイブされます。アーカイブされたダッシュボードは削除されず、アーカイブ

を解除することができます。月例**Sisense**クリーンアップ

エントロピーは、ビジネス・インテリジェンス・ツールの自然な状態ですが、避けることができます。この傾向に対処するた

めに、データチームは**Sisense**スペースで定期的にメンテナンス作業を行っています。

毎月、データチームのメンバーがメンテナンスを行います。これは、毎月末の1週間前に行われるData Opsミーティングの中で積極的に主張することができます。

このメンテナンスタスクは、毎月第1週に完了しなければなりません。そのためには、データチームのプロジェクトで新しい課題を開き、Sisense Cleanup Issueテンプレートを選択する必要があります。このテンプレートには、完了すべきタスクのリストが表示されます。すべてのタスクが完了したら、その課題を閉じることができます。

Sisense APIキー

何らかの理由でAPI Keyをローテーションする必要がある場合は、以下の場所でローテーション

- する必要があります。ハンドブックプロジェクト
- KPIスライドProject
- Okta

パフォーマンス指標のページ生成コードを担当するチームメンバーも、ローカルでページを構築するために必要になります。

---

layout: handbook-page-toc title:"Permifrost" 説明"スノーフレークパーミッションの管理"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

クイックリンク

パーミフロストプロジェクト{:.btn .btn-purple-inv} PyPI{:.btn .btn-purple-inv}。

パーミフロスト

Permifrostは、Snowflakeデータウェアハウスのパーミッションを管理するためのPythonツールです。このツールを使用するための主なドキュメントは、プロジェクト内とPyPIにあります。

注意点とエラー

- 設定ファイルに存在しないオブジェクトがあってもエラーにならない ロール/ウェアハウ
- ス/データベースの作成と削除がパーミフロストで管理されない
    - ファイルからロール全体を削除しても、ロールは削除されません

開発

以下の手順で、仮想環境の作成と準備を行います。

```
# virtualenvの作成
python -m venv ~/.venv/permifrost

# virtualenvの起動
ソース ~/.venv/permifrost/bin/activate

# 依存関係のインストール
pip install -r requirements.txt

# pip3 開発に必要なものをすべてインストールする
pip install -e '.[dev]'
```

変更をコミットしたら、マージリクエストを送信し、デフォルトのテンプレートを更新します。

リリースプロセス

バージョニング

Permifrostでは、バージョン番号として**semver**を使用

しています。前提条件

続ける前に、最新の`master`**ブランチがローカルにあることを確認してください。**

```
git fetch origin
```

ワークフロー

Permifrostはタグを使って成果物を作成しています。新しいタグをリポジトリにプッシュすると、Dockerイメージや PyPIパッケージとして公開されます。

1. 上記の開発ワークフローに従って、インストールが最新の状態になっていることを確認します。
2. 以下のコマンドを実行してください。

```
# `origin/master` から `release-next` ブランチを作成してチェックアウトします git checkout -B release-next origin/master

# view changelog (変更点とログに記録された変更点の照合)
changelog view

# チェンジログが検証された後、リリースにタグを付ける make type=minor release

  # パッチリリースを行う場合は、
  make type=patch release を実行してください。

  # メジャーリリースの場合は、make type=major release を実行してください。

# タグの確保 タグが作成されたら、先ほどバンプしたバージョンを確認します。
`0.22.0` => `0.23.0`.
git describe --tags --abbrev=0

# タグを上流にプッシュしてリリースパイプラインを起動する
git push origin $(git describe --tags --abbrev=0)

# リリースブランチをプッシュして新しいバージョンをマージし、マージリクエストを作成します git push origin release-next
```

3. マスターをターゲットとした**re**lease-nextからのマージリクエストの作成
4. 変更がマージされたら、必ずソースブランチを削除するようにしてください。
5. パブリッシュパイプラインが成功すると、リリースは**PyPI**で公開されます。

---

layout: handbook-page-toc title:"データパイプラ

イン" このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .hidden-md .hidden-lg}。

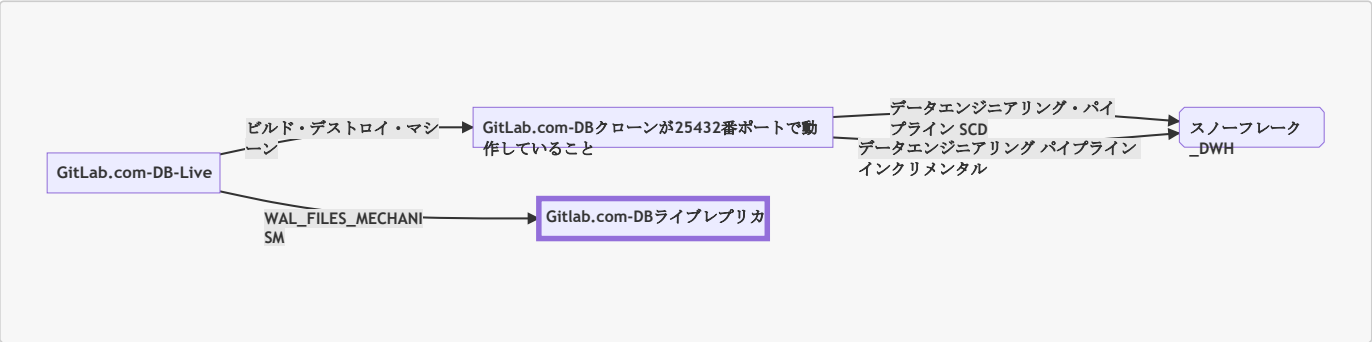
{::options parse_block_html="true" /}。

背景

データウェアハウスには、異なる抽出方法（Fivetran、Stich、Postgresパイプラインなど）を介して、異なるソースシステムからのソースデータが含まれています。このページでは、異なるデータソースと、データパイプラインを介してこれらのデータを抽出する方法について説明します。

GitLab Postgresデータベース

Snowflakeにデータを取り込むための専用のgitlab.com *read* replicaデータベースインスタンスがあります。2つのレプリカが用意されており、それぞれが独自のレプリケーション頻度と動作を持っています。



- `GitLab.com-DBのクローン`は、10:30PM UTCに破壊され、11:15PM UTCに再構築されます。この結果、このデータベースへの問い合わせが可能な時間帯は、00:00AM UTCから11:30PM UTCとなります。この時間外にこのレプリカからデータを抽出した場合、エラーが発生しますが、データが失われることはありません。
- `Gitlab.com-DB Live Replica`は、WALファイルを介して継続的にデータを取り込みます。

現在、安定したデータフィードを確保するために、インクリメンタルロードとフルロードの両方で`GitLab.com-DBの`クローンインスタンスを使用しています。開発やテストの段階では、`Gitlab.com-DBのライブレプリカ`からのロードにおいて、書き込みと読み込みが同時に行われるという問題に直面しました（クエリのコンフリクト）。クエリの競合が回復するまでの時間を長くするために、`max_standby_archive_delay`と`max_standby_streaming_delay`という設定があります。これはサーバー側で設定する必要があり、レプリケーション処理のラグを増やす結果になる可能性があります。これを避けるために、より静的なデータソースから読み出すようにしています。

クローンの設定に問題がないかどうかを確認するためのランブック項目。

モニタリング/アラート

Zlonkはジョブ完了モニタリングを実装しています。アラートはS4の深刻度で`#alerts` Slackチャンネルに流されます。以下のようなアラートが発生します。アラートが発生しました。

```
GitLab ジョブが失敗しました。GitLab ジョブの "clone" リソース "zlonk..." が失
敗しました。階層: db, タイプ: zlonk.postgres
```

私たちのパイプラインで、gitlab.comのpostgressデータベースにアクセスできない問題が発生した場合は、slackの`@sre-oncall`に連絡を取る必要があります。

インクリメンタルとフルロード

1. インクリメンタル・エキス

- これは最も便利な方法で、レコードの転送量は最小限で済みます。前提条件として、ソーステーブルレベルでデルタカラムが利用可能であること。

- **120**個のテーブルを段階的に抽出 ロード時
- **間 約1時間**
- **6時間ごとに実行される**

2. フル抽出（現時点で**±100**テーブル＆ロードタイ約**4**時間

- これは、ソース・テーブル・レベルでデルタ・カラムが利用できない場合や、ソースでレコードが削除された場合の代替手段です。
- **100**個のテーブルを段階的に抽出していく
- ロードタイム 約**4**時間
- **24**時間ごとに実行される

抽出方法は、マニフェストファイルによって決定されます。

PostgresからSnowflakeへのテーブルの手動バックフィル

このセクションは、**gitlab**の**Postgres**データベースで以下の条件を満たしているテーブルに対してバックフィルを行う必要がある場合にのみ使用します。

- テーブルのエアフロータスクは最新の日付まで完了しています。
- キャッチアップがすでに完了している日付または日付範囲について、テーブルのデータが欠落している。
- テーブルをドロップしてリロードすると**SLO**に違反する場合。

上記のケースでは、この方法を使ってデータを取り出し、**snowflake**のターゲットテーブルにロードすることができます。この問題では、テーブルの**1**つに対して行われている

以下は、そのための手順です。

Step 1:-Kubernetesのポッドに接続します。
`kubectl exec -ti <pod_name> -c <webserver|scheduler> /bin/bash`.

現在の設定では
`kubectl exec -ti airflow-deployment-7484d899c6-tfm8v -c scheduler /bin/bash`

gcpクライアントが時々変更されるため、以下のようなエラーが発生することがあります。`Unable to connect to the server: x509: certificate signed by unknown authority`.この問題を解決するには、次のコマンドを実行します。
`gcloud container clusters get-credentials data-ops --zone us-west1-a --project gitlab-analysis`

Step 2:- pythonライブラリを使って**Postgres**への接続を確立します。これは、**Postgres**に直接アクセスできないため、**Kubernetes**ポッドを経由する必要があります。データを抽出しなければならないテーブルは、それぞれのマニフェストファイルからクエリをピックアップします。以下は、クエリを実行して出力を**csv**ファイルで取得する手順です。この例では、テーブル`experiment_subjects`に対して実行しています。

```
パイソン3
import psycopg2
conn =
psycopg2.connect(
host="",
database="",
user="",
password="",
port=,
options='-c statement_timeout=900000'
)
cur=conn.cursor()
sqlstr = "COPY (SELECT id , experiment_id, user_id, group_id, project_id, variant,
created_at, updated_at, converted_at, context FROM experiment_subjects WHERE updated_at
BETWEEN '2021-04- 05T19:14:20'::timestamp AND '2021-04-26T10:08:16'::timestamp) TO STDOUT
WITH CSV HEADER "
with open('experimental_subjects.csv','w') as f:
cur.copy_expert(sqlstr, f)
```

これにより、**Kubernetes**ポッド自体にファイルが生成されます。注意：認証情報は安全な保管場所から受け取る必要があります。

Step 3: gsutilを使ってGCSからpostgres_pipelineにファイルをアップロードする。

```
gsutil cp experiment_subjects.csv gs://postgres_pipeline/ です。
```

ステップ4：タイムスタンプ変数を準備します。これは、_uploaded_at列でファイルをリッチ化するためにデータフレームからプッシュするわけではありません。そのため、コマンドラインを使って行う必要があります。

```
import time
print(time.time())
```

ターゲットテーブルには _uploaded_at カラムがあり、入力する必要があるため、返された値は snowflake へのコピーの際に使用されます。ステップ 5: マニフェスト ファイルで定義された各列を以下のように読み込み、_uploaded_at タイムスタンプ列を以下のように追加します。これは、データが正しいかどうかを検証するためにsnowflakeで実行する必要があり、特定の監査を行うために使用することもできます。

```
SELECT $1 as id ,
experiment_idとして$2。
user_idとして$3。
$4 as group_id,
project_idとして$5。
バリアントとして6ドル。
created_atとして$7。
updated_atとして8ドル。
converted_atとして9ドル。
コンテキストとして $10, _uploaded_at として
'1621841251.8090081' FROM
`@stage`/experiment_subjects.csv ;
```

ステージは、Postgresのパイプラインのステージです。

ステップ6: copy intoを使ってデータを読み込み、RAW.tap_postgres.gitlab_db_experiment_subjectsテーブルにこのデータを読み込みます。

GitLabデータベーススキーマの変更とデンジャーファイル

データエンジニアリングチームは、GitLabプロジェクトにデンジャーファイルを作成しています。ソーススキーマの変更が通知されることは、GitLab.com データベースからの抽出処理でエラーが発生しないようにするために不可欠です。なぜなら、抽出は一連の select 文を実行することで行われるからです。その日のトリアージを担当するデータエンジニアは、スキーマの変更を調査し、データチームが必要とするアクションのための課題を作成するDRIです。

トラステッドデータフレームワークテスト

このデータパイプラインでは、3種類のTrusted Data Frameworkテストが実行されます。

1. **ソースの鮮度。**
2. **ロウカウントテスト。**
3. **データの実測値です。**

dbt機能によりソースの鮮度を確認。
行数のカウントとデータの取得は、Airflowのgitlab_com_data_reconciliation_extract_loadという追加のDAGを介して行われ、結果はRAW.TAP_POSTGRES.GITLAB_PGP_EXPORTというsnowflakeテーブルに格納されます。

サービス ping

サービスpingは、GitLab Incが特定のGitLabインスタンスの使用状況データを収集するための方法です。サービスPing（旧称：Usage Ping）について、製品の観点からの詳しい情報はこちらをご覧ください。豊富なドキュメントを含む包括的なガイドは、ServicePingGuideに掲載されています。

サービスピンには大きく分けて2種類あります。

- セルフマネージドサービス
- Ping SaaSサービスPing

詳しくは「4種類のサービスPingプロセス」をご覧ください。

セルフマネージドサービスPing

セルフマネージド・サービスPingは、Versionsアプリからデータウェアハウスにロードされ、VERSION_DBに格納されます。データベースを使用しています。

SaaSサービスのPing

SaaS Service Pingは、2つの方法でデータウェアハウスにロードされます。

- Gitlab Postgres Database ReplicaのSQL文（SQLベース）とRedisのRestFUL API
- コール（Redisベース）を使用した場合

データチームによる実装の詳細は、Readme.mdファイルに記載されています

。 

インスタンスのSQLベースのメトリクスの読み込み

データはPostgresのSQLレプリカから読み込まれます。クエリは、この抽出物の中にある非常に大きなJSONファイル（ファイル内に数百のクエリがある）でバージョン管理されています。クエリは、インスタンスクエリと名前空間クエリの2つのカテゴリに分かれています。インスタンスクエリはGitLab.com全体のデータを生成し、名前空間クエリはGitLab.comの各名前空間のデータを生成します。データはテーブル（RAW.SAAS_USAGE_PINGスキーマ内）に格納されます。

- raw.saas_usage_ping.instance_sql_metrics
- RAW.SAAS_USAGE_PING.INSTANCE_SQL_ERROR - このテーブルには、SQLメトリクスのデータ処理中にエラーがポップアップするSQLコマンドが含まれています。
- raw.saas_usage_ping.gitlab_dotcom_namespace

実装の詳細はsql-metrics-implementationで公開されています。 Redisベースのメト

リクスのインスタンスの読み込み

データはAPI経由でダウンロードされますので、APIの仕様をご参照ください。UsageDataNonSqlMetricsAPIを参照してください。データはJSON形式で保存され、約2k行のサイズです。通常、1回のロードで1ファイルとなります *（現在は週1回のロードとなっています）*。Redisからデータをロードする主な目的は、SQLクエリでは取得できない細かい粒度のメトリクスを確保することです。データはテーブル（RAW.SAAS_USAGE_PINGスキーマ内）に格納されます。

- raw.saas_usage_ping.instance_redis_metrics

実装についての詳細は redis-metrics-implementation で公開されています。

シートロード

SheetLoadは、GCSやS3からのGoogleシートやCSVをデータウェアハウスに取り込むためのプロセスです。SheetLoad

の使用方法に関する技術文書は、データチームプロジェクトのReadmeに記載されています。

GoogleシートやCSVをウェアハウスにインポートしたい場合は、データチームのプロジェクトで「CSVまたはGSheetsデータアップロード」の課題テンプレートを使って課題を作成してください。このテンプレートには、インポートしたいデータの種類と、そのデータを使って何をしたいかに応じて、詳細な手順が記載されています。

考察

{:#mind-about-sheetload}。

SheetLoadは、主にスプレッドシートを正規のソースとするデータ（例：売上見積書）に使用されます。スプレッドシート以外のデータソースがある場合は、少なくとも新しいデータソースを作成してデータを自動的に取得するようにしてください。しかし、スプレッドシートがこのデータの**SSOT**であるならば、**SheetLoad**はデータをウェアハウスに取り込むための適切なメカニズムです。

スノーフレークへの読み込み

SheetLoadは、データベース内のテーブルを、読み込み元のシートの正確なコピーにするように設計されています。**SheetLoad**は、読み込み元のシートに変更があったことを検知すると、データベースのテーブルを削除し、更新されたスプレッドシートのイメージでテーブルを再作成します。つまり、列が追加されたり変更されたりしても、すべてデータベースに反映されるのです。変更は**24**時間以内に検出されます。

SheetLoadの準備

どうしても不可能な場合を除いて、**SheetLoad**シートは`importrange`関数を使って元の**Google**シートから直接インポートするのがベストです。これにより、上流のシートを残したまま、**SheetLoad**版をプレーンテキストにフォーマットすることができます。追加のデータタイプの変換やデータのクリーンアップは、ベースとなる**dbt**モデルで行うことができます。**(**これは**Boneyard**には適用されません**)**。

読み込みのためにデータを準備する際の注意点をいくつかご紹介します。

- 数字の列はすべて、カンマやその他の記号を使わないようにフォーマットします - **$1,000.46** の代わりに **1000.46** 可能な限りシンプルなヘッダーを使用します - `GitLab - User Name` の代わりに `user_name`
- データがないことを示すために、空白のセルを使用します。空白のセルは、データベースに`NULL`として保存されます。

モデリング

通常の**SheetLoad**データが**Sisense**でアクセスできるようになる前に、**dbt**でモデル化する必要があります。ソースモデルとステージングモデルの最低**2**つのモデルがシートに対して作成されます。これらのモデルは、データチームのプロジェクトで課題を作成した後、データチームのメンバーによって作成されます。

ボンヤード

ボーンヤード・スキーマは、スプレッドシートからデータをアップロードすることができ、**Sisense**内で直接クエリを利用することができます。ただし、これは**1**回限りの分析が必要な場合や、すでにウェアハウスにあるデータに結合する必要がある場合のためのものです。このデータは、*アドホック/ワンオフ*のユースケースにのみ関連し、比較的短期間で陳腐化することを強調するために**Boneyard**と呼ばれています。我々は定期的に`Boneyard`スキーマから古くなったデータを削除します。

証明書

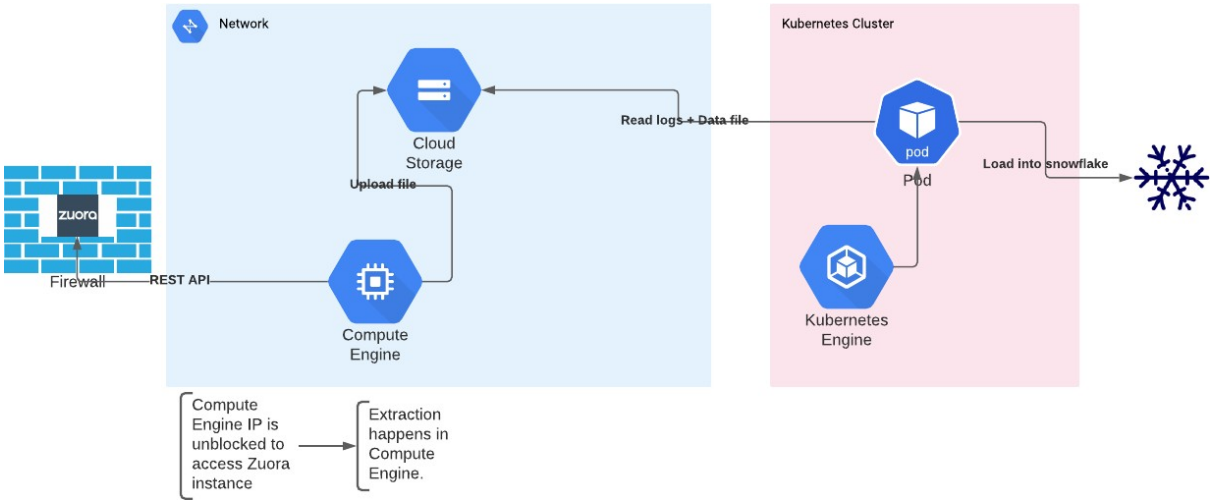
SheetLoadに証明書を追加する場合は、**People Group**ページの説明を参照してください。

Zuoraの収益

Zuora Revenueは、最新の収益基準（**ASC606**、**IFRS15**）に準拠して、複雑な収益管理プロセスを自動化できるアプリケーションです。**Zuora Revenue** の抽出プロセスのデータパイプラインの一部として、**REST Call** を使用して **Zuora Revenue** の **BI** ビュー **(Zuora Revenue** が製品にデフォルトで作成したもので、我々はこれらのビューのみを使用し、これらのビューを作成または変更することはできませんでした**)** からデータを抽出しています。**Zuora Revenue BI**ビューのデータエンティティは、主要な物理テーブルに基づいています。一部の**BI**ビューは、ベースとなっている物理テーブルと類似しています。その他の**BI**ビューは、物理テーブルに基づく計算から派生します。

Zuoraネットワークアーキテクチャ

Zuora システムはファイアウォールの後ろに置かれています。ファイアウォールを通過してアクセスするためには、**GitLab** の**IP**アドレスを許可リストに載せる必要があります。**Kubernetes Engine**は固定**IP**を持っていないので、**Zuora**にアクセスするために固定**IP**を持ったコンピュートエンジンを追加しています。



このリストから、以下のテーブルだけが、**Zuora Revenue**に**Gitlab**のデータが読み込まれているか、統合のために現時点で持っているデータです。

以下は、データがあり、**snowflake**で作成されるテーブルのリストです。データを持たないテーブルは作成されません。これは、**Zuora API**が提供するテーブル定義が、カラムの序列位置ではなく、アルファベット順になっているため、カラムのラベル付けが正しくないためです。

シリアル番号	データ主体 in Zuora	物理テーブルでの名前	ZuoraViewでの名前	ZuoraTableでの名前	SFHasレコード	プレゼントスノーフレーク
1	AccountRタイプ	RPRO_BI3_ACCT_TYPE_V	BI3_ACCT_TYPE	BI3_ACCT_TYPE	Yes	Yes
2	会計プレ概要	rpro_bi3_ri_acct_summ_v	BI3_RI_ACCT_SUMM	BI3_RI_ACCT_SUMM	はい。	はい。
3	承認	rpro_bi3_appr_dtl_v	BI3_APPR_DTL	BI3_APPR_DTL	はい。	はい。
4	ビル	rpro_bi3_rc_bill_v	BI3_RC_BILL	BI3_RC_BILL	はい。	はい。
5	カレンダー	rpro_bi3_calendar_v	BI3_CALENDAR	BI3_CALENDAR	はい。	はい。
6	削除されたスケジュール	RPR_BI3_RC_SCHD_DEL_V	BI3_RC_SCHD_DEL	BI3_RC_SCHD_DEL	はい。	はい。
7	ヘッダー	rpro_bi3_rc_head_v	BI3_RC_HEAD	BI3_RC_HEAD	はい。	はい。
8	ホールズ	rpro_bi3_rc_hold_v	BI3_RC_HOLD	BI3_RC_HOLD	はい。	はい。
9	ライン	rpro_bi3_rc_lns_v	BI3_RC_LNS	BI3_RC_LNS	はい。	はい。
10	MJE	RPRO_BI3_MJE_V	BI3_MJE	BI3_MJE	はい。	はい。
11	POB	rpro_bi3_rc_pob_v	BI3_RC_POB	BI3_RC_POB	はい。	はい。
12	スケジュール	RPR_BI3_RC_SCHD_V	BI3_RC_SCHD	BI3_RC_SCHD	はい。	はい。
13	ウォーターフォール（派生型	rpro_bi3_wf_summ_v	BI3_WF_SUMM	BI3_WF_SUMM	はい。	いいえ
14	オーガ	RPRO_BI3_ORG_V	BI3_ORG	BI3_ORG	いいえ	いいえ
15	アカウント概要 (Derived)	rpro_bi3_ln_acct_summ_v	Bi3_ln_acct_summ	Bi3_ln_acct_summ	いいえ	いいえ
16	本	rpro_bi3_book_v	BI3_BOOK	BI3_BOOK	いいえ	いいえ

シリアル番号	データ主体 in Zuora	物理テーブルでの名前	ZuoraViewでの名前	ZuoraTableでの名前	SFHasレコード	プレゼントスノーフレーク
17	コスト	rpro_bi3_rc_ln_cost_v	BI3_RC_LN_COST	BI3_RC_LN_COST	いいえ	いいえ

Zuora収入エキス

▶ 以下は、「Zuora Revenue」の抽出周辺情報です。 抽出パイプラインは、ここをクリックすると拡大します。

Compute engineでの環境設定

自分のサービスアカウントを使ってzuora compute engineにSSH接続します。以下はGCPのサーバーの詳細です https://console.cloud.google.com/compute/instancesDetail/zones/us-west1-a/instances/zuora-revenue-extract-server?project=gitlab-analysis&rif_reserved

ssh -o UserKnownHostsFile=/dev/null -o CheckHostIP=no -o StrictHostKeyChecking=no -i HOME/.ssh/google_compute_engine -A -p 22 @<external_ip> です。

別の端末から~/repos/analytics/extract/zuora_revenue/srcディレクトリに移動し、以下のコマンドを実行してディレクトリ全体をcompute engineにアップロードします。このディレクトリには、抽出処理のコードが含まれています。

gcloud compute scp --recurse src --zone "us-west1-a" zuora-revenue-extract-server:/home/vedprakash/zuora_revenue これにより、ローカルのブランチからcompute engineのブランチにsrcフォルダがアップロードされます。

接続後、ファイルをアップロードする。

Step 1 : コンピュートエンジン内に仮想環境を作成 --- zuora-revenue-extract-venv

変更を最小限にするために、同じ名前にしておきます。 python3 -m venv zuora-revenue-extract-

venv Step 2: venvの有効化

source /home/vedprakash/zuora-revenue-extract-venv/bin/activate ス

テップ3：pipをアップグレードした後

pip install --upgrade pip

ステップ4： srcフォルダに移動し、必要なパッケージをすべてインストールします。

pip install -r requirements.txt

注意事項ステップ1からステップ4は、環境がクラッシュしていて最初から構築しなければならない場合にのみ必要となり、一般的な運用では必要ありません。

以下のステップは、GCSバケットフォルダを誤って削除してしまった場合に必要となります。また、新しいテーブルをシステムに追加する必要がある場合にも、以下のステップを使用することができます。

ステップ5：テーブル名とロードデートの情報を保持するstart_date_<テーブル名>.csvファイルを作成する。例えば、テーブルBI3_MJEの場合、ファイル名はstart_date_BI3_MJE.csvとなり、ファイルの内容は以下のようになります。

table_name,load_date BI3_ACCT_TYPE 新しいテーブルのload_dateは、ファイルを最初からダウンロードし始めるので、空白にしておきます。それ以外の場合は、エアフローログから最終ロード日を拾うことができます。現在のテーブルについて、ファイルを作成するためのコマンドリストを以下に示します。これは、ローカルからでも、コンピュートエンジンからでも可能です。

```
echo "テーブル名,load_date
BI3_ACCT_TYPE," > start_date_BI3_ACCT_TYPE.csv
echo "テーブル名,load_date
BI3_APPR_DTL," > start_date_BI3_APPR_DTL.csv
echo "テーブル名,load_date
```

```
BI3_CALENDAR," > start_date_BI3_CALENDAR.csv
echo "テーブル名,load_date
BI3_MJE," > start_date_BI3_MJE.csv
echo "テーブル名,load_date
BI3_RC_BILL," > start_date_BI3_RC_BILL.csv
echo "テーブル名,load_date
BI3_RC_HEAD," > start_date_BI3_RC_HEAD.csv
echo "テーブル名,load_date
BI3_RC_HOLD," > start_date_BI3_RC_HOLD.csv
echo "テーブル名,load_date
BI3_RC_LNS," > start_date_BI3_RC_LNS.csv
echo "テーブル名,load_date
BI3_RC_POB," > start_date_BI3_RC_POB.csv
echo "テーブル名,load_date
BI3_RC_SCHD," > start_date_BI3_RC_SCHD.csv
echo "テーブル名,load_date
BI3_RC_SCHD_DEL," > start_date_BI3_RC_SCHD_DEL.csv
echo "テーブル名,load_date
BI3_RI_ACCT_SUMM," > start_date_BI3_RI_ACCT_SUMM.csv
```

このコマンドは、各テーブルのファイルを作成し、必要なカラム名と値を入力します。**load_date**には**null**を設定していますが、これは初回実行時の扱いとなるためです。注：ロード日がわかっている場合は、`2016-07-26T00:00:00`形式の`%Y-%m-`にします。
特定のテーブルの `%dT%H:%M:%S`。

Step6: ステージングエリアにファイルをアップロードする必要があります。以下は、各ファイルをステージング・エリアの各テーブルにアップロードするためのコマンド群です。

```
gsutil cp start_date_BI3_MJE.csv              gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_MJE/
gsutil cp start_date_BI3_ACCT_TYPE.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_ACCT_TYPE/
gsutil cp start_date_BI3_APPR_DTL.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_APPR_DTL/
gsutil cp start_date_BI3_CALENDAR.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_CALENDAR/
gsutil cp start_date_BI3_RC_BILL.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_BILL/
gsutil cp start_date_BI3_RC_HEAD.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_HEAD/
gsutil cp start_date_BI3_RC_HOLD.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_HOLD/
gsutil cp start_date_BI3_RC_LNS.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_LNS/
gsutil cp start_date_BI3_RC_POB.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_POB/
gsutil cp start_date_BI3_RC_SCHD.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_SCHD/
gsutil cp start_date_BI3_RC_SCHD_DEL.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RC_SCHD_DEL/
gsutil cp start_date_BI3_RI_ACCT_SUMM.csv gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_RI_ACCT_SUMM/
```

ステップ7： 抽出液を実行するには、サーバーの**.bash_profile**ファイルで以下の変数を宣言する必要があります。

```
    export zuora_bucket=""
    export zuora_dns=""
    export authorization_code=""
    export python_venv="source /home/vedprakash/zuora-revenue-extract-venv/bin/activate" #From step
2
    エクスポート zuora_extract_log="/home/vedprakash/zuora_revenue/src/logs/"
    export zuora_src="/home/vedprakash/zuora_revenue/src" #ソースコードのパスです。
```

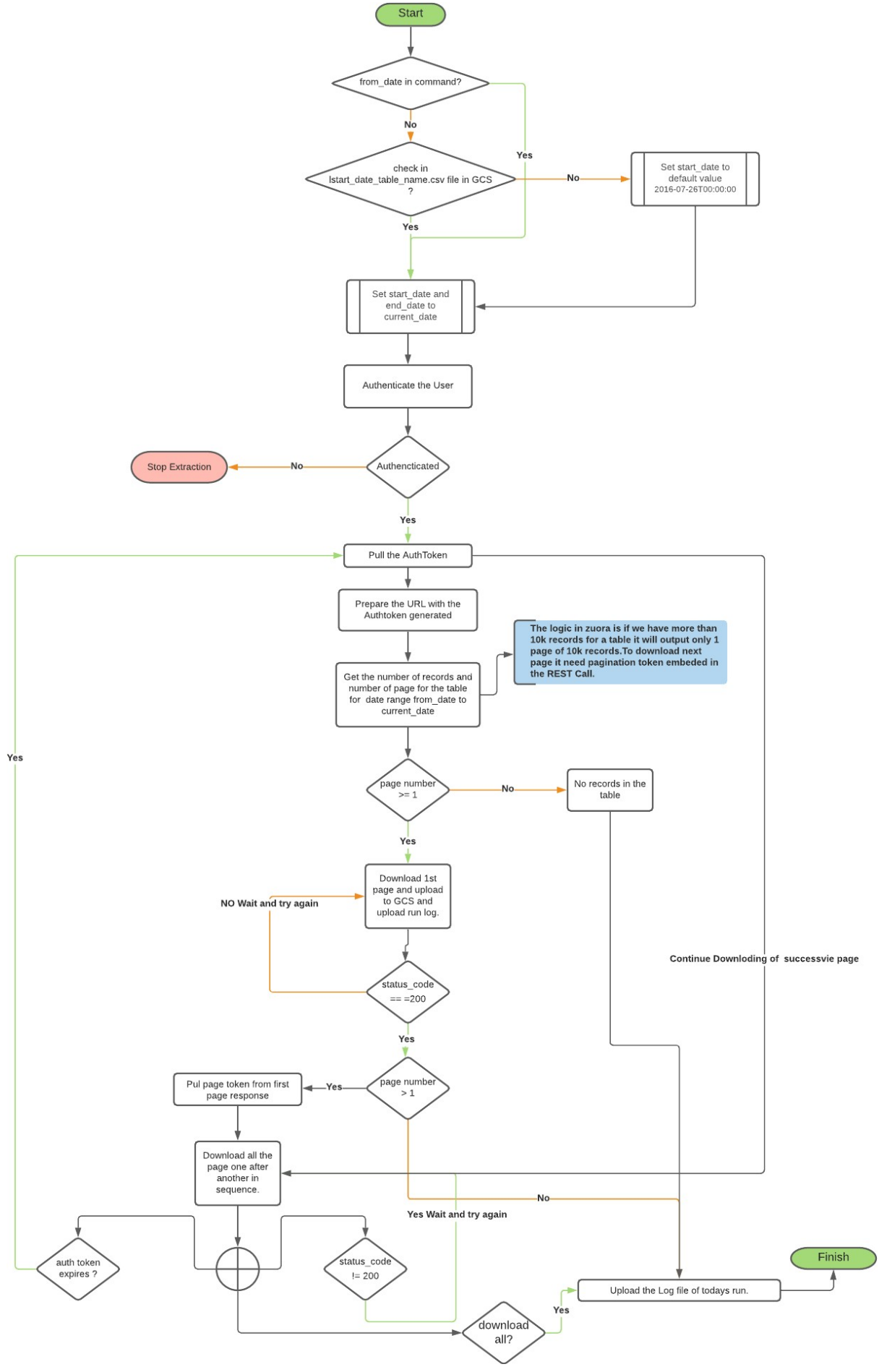
注：認証情報は`zuora_revenue_prod`の下の**1**つのパスワードに存在します。

ステップ8：最後のステップは、スケジュールを行うことです。**crontab**に以下のコマンドを追加します。編集して準備ができたら、必要なコマンドをそのマシンの**crontab**に追加します。現在のスケジュールは、毎日午前**2**時（**UTC**）に実行されるように設定されています。

```
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名                    BI3_ACCT_TYPE-バケット名 $zuora_bucket -api_dns名 $zuora_dns -
api_auth_code "$authorization_code" &>/tmp/mycommand.log
```

```
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_APPR_DTL      -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_CALENDAR      -バケット名 $zuora_bucket-api_dns名 $zuora_dns-api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_MJE           -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_BILL       -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_HEAD       -バケット名 $zuora_bucket-api_dns名 $zuora_dns-api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_HOLD       -バケット名 $zuora_bucket-api_dns名 $zuora_dns-api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_LNS        -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_POB        -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RC_SCHD       -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-table_name                   BI3_RC_SCHD_DEL-bucket_name $zuora_bucket -api_dns_name $zuora_dns
-api_auth_code "$authorization_code" &>/tmp/mycommand.log
00 02 * * * . $HOME/.bash_profile;$python_venv && cd $zuora_src && python3 extract_zuora_revenue.py
-テーブル名 BI3_RI_ACCT_SUMM -バケット名 $zuora_bucket -api_dns名 $zuora_dns -api_auth_code
"$authorization_code" &>/tmp/mycommand.log
```

Zuora Extractのフローチャート



プロセスの最後には、以下が出力されます。

1. **zuora_revpro_gitlab/RAW_DB/staging/<table_name>/<table_name>_DD-MM-YYYY.log.log**という名前の成功ログファイルが存在します。たとえば、テーブル **BI3_MJE** の場合、その日のログ ファイルの名前は `BI3_MJE_21-06-2021.log` で、パス `gs://zuora_revpro_gitlab/RAW_DB/staging/BI3_MJE/BI3_MJE_21-06-2021.log` にアップロードされます。
2. **GCS**バケットには、その日付範囲のファイルがすべて存在します。

Snowflakeに抽出用のテーブルを追加するためです。

このリストから、いずれかのテーブルがデータを取得し、そのエントリを**snowflake**に追加する必要がある場合は、以下の手順を実行する必要があります。ステップ**1**：エントリー`extract/zuora_revenue/zuora_revenue_table_name.yml`を `zuora_revenue_load_snow` **DAG**の**add task**に追加します。ステップ**2**： それぞれのテーブルについて、ステップ**5**、ステップ**6**、ステップ**8**に従います。
ステップ**3**： ダウンロードしたファイルからカラム名をピックアップし、すべてのカラムを**varchar**に設定して、**snowflake RAW.ZUORA_REVENUE**スキーマにテーブルを作成します。

注意事項テーブルの定義を作成する際に **describe column API** を使用しないでください。

派生したテーブルの場合

Zuora は、派生ビューのためのビュー定義を提供しています。本番環境では、派生ビューからデータを抽出することはできません。そのため、テーブル**BI3_WF_SUMM**については、**Zuora**から提供された**DDL**を使って**PREP**層の**DBT**モデルでデータを準備します。**DDL**の定義は、レポの**extract/zuora_revenue/README.md**にあります。

Zoominfo

ZoomInfoは、**B2B**のセールスおよびマーケティングチームのための**Go-to-Market Intelligence**プラットフォームです。この統合されたクラウドベースのプラットフォームは、営業担当者やマーケティング担当者に包括的な情報を提供し、潜在的な新規顧客を見つけるのに役立ちます。このような充実したデータを得るためには、**Gitlab**は**Zoominfo**にアウトバウンドでデータを送信し、処理後に**GitLab**は**Zoominfo**から**Snowflake**データシェアを介してインバウンドテーブルとして処理されたデータを受け取る必要があります。

Zoominfoのデータパイプラインは、**Snowflake**データ共有手法を活用した自動化された双方向のデータパイプラインです。

Snowflake Dataのシェアです。

[Snowflake data share] (https://docs.snowflake.com/en/user-guide/data-sharing-intro.html)は、**1**つのアカウントから**snowflake**データベーステーブルの共有を可能にし、また、外部アカウントから共有されたデータへのアクセスを可能にします。これには、自分のアカウントでデータベースのアウトバウンド共有を作成し、**Web**インターフェース**/SQL**のいずれかを使用して外部アカウントに共有する必要のある**snowflake**テーブルへのアクセスを許可する必要があります。

SQLを使った**Snowflake Data Share**。

以下は、**SQL**を使用して**snowflake**データシェアを介してアウトバウンド**/**インバウンドシェアを操作するための手順です。

SQLを使ったアウトバウンドシェア。

例えば、`prod` というデータベースに `share` というスキーマを持ち、`gitlab_user_outbound` というテーブルを消費者アカウント `azitest` と共有する場合を考えます。以下の**SQL**を実行してアウトバウンドシェアを作成します。

- ステップ**1**： **accountadmin**ロールを使って**Share**を作成します。
  - **USE ROLE accountadmin;**
  - **CREATE SHARE share_test;**
- ステップ**2**： 特権を与えてデータベース、スキーマ、テーブルを共有に追加する。
  - **GRANT USAGE ON DATABASE prod TO SHARE share_test;**
  - **GRANT USAGE ON SCHEMA prod.share TO SHARE share_test;**
  - **GRANT SELECT ON TABLE prod.share.gitlab_user_outbound TO SHARE share_test;**
- ステップ**3**：消費者アカウントを共有に追加します。アカウントを共有に追加するためには、消費者はアカウントの詳細を提供する必要があり、消費者とプロバイダーの両方のアカウントが同じスノーフレークリージョンにある必要があります。

- **ALTER SHARE share_test ADD ACCOUNTS = 'azitest';**

SQLを使ったインバウンドシェア。

例えば、gitlabという名前の共有がアカウントazitestから私たちに共有されている場合、以下のSQLを実行してsnowflakeにデータベースを作成し、受信した共有内のテーブルとデータにアクセスします。

```
* CREATE DATABASE zoominfo_inbound FROM SHARE azitest.gitlab;
```

Snowflake Webインターフェースを使用したSnowflake Data Share。

以下は、Webインターフェースを使用してsnowflakeデータ共有を介してアウトバウンド/インバウンド共有を操作するための手順です。accountadminロールを使用して、snowflake Webインターフェイスの共有ページに移動し、インバウンド/アウトバウンドのデータ共有タスクを実行します。

 イメージ-1.png

snowflakeのウェブインターフェイスを使ったアウトバウンドシェア。

- ステップ1： アウトバウンド共有を作成するには、snowflakeウェブインタフェースの共有ページで、アウトバウンドアイコンをクリックし、次に+createアイコンをクリックします。
- ステップ2： 安全な共有名、データベース、テーブル/ビューの詳細を追加し、下部の「作成」ボタンをクリックします。 

- ステップ3： お客様のアカウントを共有に追加し、リーダー/フルアカウントの作成を選択して、下部の追加ボタンをクリックします。これにより、共有が作成され、テーブルやビューが消費者に共有されます。

 イメージ-3.png

snowflakeのウェブインターフェイスを使ったインバウンドシェア。

インバウンドシェアは、snowflakeウェブインタフェースのシェアページの「インバウンド」タブで確認できます。インバウンド共有のテーブルやデータにアクセスするためには、共有データベースを作成する必要があります。共有データベースを作成するには、「Create database from secure share」アイコンをクリックし、データベース名を入力し、アクセス権を付与して「create database」ボタンをクリックします。このプロセスにより、snowflake 内に zoominfo_inbound データベースが作成されます。受信テーブルとデータは、snowflake のこの共有データベースの下でアクセスできます。共有データベースからのデータはプレップに取り込まれる。

 Image-4.png

Image-5.png 

建築  イメージ.png

アウトバウンドテーブル。

- "PROD"...SHARE"...GITLAB_USER_OUTBOUND - Outbound tableは、First name, Last name, email address, company nameなどのGitlabのユーザー情報を持ちます。アウトバウンドテーブルは、Snowflakeのデータ共有を介してZoominfoに一度だけ共有されます。テーブルはdbtで作成されるため、時間の経過とともに変更されます。新しく更新されたデータをこのテーブルに取り込むのは、Zoominfoの責任です。

Inboundテーブルの読み込み。

Zoominfoは、インバウンドファイルをSnowflakeデータシェア経由でGitlabに送信します。共有データベースZOOMINFO_INBOUNDは、WebインターフェイスまたはSQLを使ってインバウンドシェアから作成されます。受信テーブルは、データをRaw層に取り込み、dbtがデータを変換するという標準的なアーキテクチャには従いません。余分なプロセスを作らず、パイプラインをより効率的にするために、共有データベースを生のデータベースとして扱います。のインバウンドテーブ

ルからのデータは

ZOOMINFO_INBOUNDはSnowflake Data Exchange loaderを使用してSnowflake `PREP` **に取り込まれます。以下のインバウンド・テーブルのリストは、'zoominfo'スキーマの下でPREPデータベースに作成されます。**

- `"ZI_COMP_WITH_LINKAGES_GLOBAL"` **- 国際テーブルは、情報を持っているすべての企業のリストを持っています。これは1回の共有です。**
- `"ZI_REFERENCE_TECHS"` **- データベースに登録されている企業が使用しているテクノロジーのリストを持つテクノグラフテーブル。**
- `"GITLAB_CONTACT_ENHANCE"` **- User table company マッチしたテーブルで、Gitlab が zoominfo に送信するユーザーリストに会社情報を追加します。GitlabがZoominfoに送信するのは一度だけですが、付加されたデータは四半期ごとに更新することができます。**

---

layout: handbook-page-toc title:"Pythonガイド" 説明"私たちが選んだリンターがすべてを捕らえているわけではないので、この Python スタイルガイドを施行するのは私たちの集団の責任です。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

パイソンガイド

私たちが選んだリンターがすべてを捕らえるわけではないので、このスタイルガイドを実施するのは*私たちの責任*です。

価値観

Campsite rule - このガイドラインはまだ完成していないため、現在スタイルガイドに準拠していないコードを使用している場合は、見かけたら更新してください。

リンティング

リンターには**Black**を使用しています。初期設定のままで使用し
ています。

`レビュー`**段階では、レポ全体をlintし、ファイルをフォーマットする必要がある場合はゼロ以外の終了コードを返す手動CIジョブがあります。MRがマージされる前にこのジョブが通過するかどうかは、MRの作者とレビュアーの両方にかかっています。レポ全体をlint するには、レポの先頭から**`black .`**を実行します。**

スペーシング

PEP8では、コードの論理的な部分を空行で囲むことを推奨しています。`for`ループや`if/else`の開始時には
ブロックでは、セクションの上に改行を追加して、コードに余裕を持たせます。改行のコストは安いですが、頭を使う時間は

高くつきます。タイプのヒント

すべての関数のシグネチャには、たとえ`None`であっても、戻り値の型を含めた型ヒントを含めるべきです。これは良いドキュメントであり、型チェックやエラーチェックのために **mypy** で使用することもできます。

例

```
def foo(x: int, y: int) ->
    int: """
    2つの数字を足して返します。"""

    x + yを返す

def bar(some_str: str) ->
    None: """
```

```
    文字列を印刷しま
    す。"""
    print(some_str)
    return
```

インポートオーダー

輸入品はPEP8のルールに従うべきであり、さらに、前に出てくる輸入品...ステートメントと一緒に注文する必要があります。
`from .... import ...`

例

```
import logging
import sys
from os import environ

import pandas as pd
from requests import
get

import some_local_module
from another_local_module import something
```

Docstrings

ドックストリングはすべての関数で使用する必要があります。関数のシグネチャで型のヒントを使用しているので、各パラメータを記述する必要はありません。ドックストリングは3重のダブルクォートを使用し、句読点を含む完全な文章でなければなりません。

例

```
def foo(x: int, y: int) ->
    int: """
    2つの数字を足して、その結果を返す。"""

    x + yを返す

def bar(some_str: str) ->
    None: """
    文字列を印刷します。

    これもちゃんとした文章です。"""
    print(some_str)
    return
```

環境変数の統合方法

関数をできるだけ再利用できるようにするために、（*よほど*の理由がない限り）関数の中で環境変数を直接使うことは非常にお勧めできません（以下にその例を示します）。その代わりに、使いたい変数を指定して渡すか、すべての環境変数を辞書として渡すのがベストな方法です。これにより、任意の辞書を渡して互換性を持たせることができるとともに、環境レベルで変数を定義する必要がなくなります。

例

```
import os
from typing import Dict

## Don't do this!
def foo(x: int) ->
    int: """
    2つの数字を足して返します。"""

    return x +
os.environ["y"] foo(1)

## Do this!
env_vars = os.environ.copy() # copyメソッドはenv varsの通常のdictを返します。 def
bar(some_str: str, another_string: str) -> None:
    """
    2つの文字列を連結して表示します。"""
    print(some_str +
    another_string) return
bar("foo", env_vars["bar"])

## Or do this!
def bar(some_str: str, env_vars: Dict[str, str]) ->
    None: """
    2つの文字列を連結して表示します。"""
    print(some_str + env_vars["another_string"])
    return
bar("foo", env_vars)
```

パッケージのエイリアス

一般的なサードパーティ製パッケージには、いくつかの標準的なエイリアスを使用しています。それは以下の通りです。

- `import pandas as`
- `pd import numpy as`
  `np`

変数名の命名規則

名前に型を加えることは、自己文書化に適したコードです。可能であれば、変数には常に、特にデータ型に関して説明的なネーミングを使用してください。以下はその例です。

- `data_df`は、データフレーム
- `params_dict`は、辞書
- `retries_int`は、整数
- `bash_command_str`は、文字列

定数を関数に渡す場合は、渡される各変数に名前を付けて、それぞれが何であるかが明確になるようにします。最後に、

変数名が重複しないようにしましょう。

例

```
def bar(some_str: str, another_string: str) ->
    None: """
    2つの文字列を連結して表示します。"""
    print(some_str + another_string)
```

```
    戻る

## Do this!
bar(some_str="foo", another_string="bar")

## Or do this!
some_str = "foo"
another_string =
"bar"
バー(some_str, another_string)

## But don't do this!
bar(some_str=some_str, another_string=another_string)
```

スクリプトを実行可能な状態にする

スクリプトを作成したばかりなのに実行できない場合は、実行可能な状態にする必要があります。以下を実行してみてください。

```
chmod 755 yourscript.py
```

chmod 755の説明は、この**askubuntu**のページをご覧ください。

変更可能なデフォルト関数の引数

関数のデフォルト引数に可変型データ構造を使用すると、コードにバグが発生することがあります。これは、新しいミュータブルなデータ構造は、関数の定義時に一度だけ作成され、その後の呼び出しのたびにデータ構造が使用されます。

例

```
def append_to(element,
    to=[]):
    to.append(element)
    に戻る

my_list =
append_to(12)
print(my_list)

my_other_list =
append_to(42)
print(my_other_list)
```

出力します。

```
[12]
[12, 42]
```

参考： **https://docs.python-guide.org/writing/gotchas/** 新規

抽出のためのフォルダ構成

- クライアント固有のロジックはすべて**/extract**に格納し、再利用可能な**API**クライアントはすべてオーケストレーション
- パイプライン固有の操作は、**/extract**に格納する必要があります。
- **extract** のフォルダ構造には、スクリプトが複数のデータセットを抽出する場合、 `extract_qualtrics_mailingsends` や `extract_qualtrics` のように、 `extract_{source}_{dataset_name}` というファイルを含める必要があります。このスクリプトは、**extract**の主な機能と考えることができ、**extract DAG**の出発点として実行されるファイルです。

Pythonを使用しない場合

このスタイルガイドはデータチーム全体を対象としていますので、Pythonを使うには時と場所があり、それは通常データモデリングの段階以外であることを覚えておくことが重要です。データ操作には可能な限りSQLを使用してください。

ユニットテスト

PytestはAnalyticsプロジェクトでユニットテストを実行するために使用されます。テストはプロジェクトのルートディレクトリからpython_pytest CIパイプラインジョブで実行されます。このジョブは、テスト結果のJUnitレポートを生成し、GitLabで処理されてマージリクエストに表示されます。

新しいテストの作成

新しいテストファイルの名前は、test_*.py というパターンに従うべきです。そうすれば、pytest が見つけてくれますし、リポジトリの中でも簡単に認識できます。新しいテストファイルは、test という名前のディレクトリに置く必要があります。test ディレクトリは、テストされるファイルと同じ親ディレクトリを共有する必要があります。

テストファイルは、1 つ以上のテストから構成されます。個々のテストは、1つまたは複数のPythonのアサート文を持つ関数を定義することで作成されます。アサートがすべて真であれば、テストはパスします。一つでも偽のアサートがあれば、テストは失敗します。

インポートを書く際には、テストはルートディレクトリから実行されることを覚えておくことが重要です。将来的には、必要に応じてテストを容易にするために、PythonPathに追加のディレクトリを追加することができます。

例外処理

APIからデータを抽出するPythonクラスを書いた場合、そのクラスはAPIプロセスのエラーをハイライトする責任があります。データモデリング、ソースの鮮度やフォーマットの問題は、dbtテストを使ってハイライトされるべきです。

一般的なtry/exceptブロックの使用は避けてください。

```
# Don't do
this! try:
   print("Do something")
except:
   print("あらゆるタイプの例外を捕捉")

# Do this
while maximum_backoff_sec > (2 **
    n): try:
        print("Do something")
    except APIError as gspread_error:
        if gspread_error.response.status_code in (429, 500, 502,
            503): self.wait_exponential_backoff(n)
            n = n +
        1 else:
            レイズ
の他にもあります。
    error(f "Max retries exceeded, giving up on {file_name}")
```

---

layout: handbook-page-toc title:"SiSense Style Guide" description:"SiSense Style Guide"

このページでは

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

全体的なルック＆フィール

KPIチャート

建設中

フィルタリング可能なレポート

建設中

適切なチャートの選択

建設中

カラー、ラベル、データ値

標準的なカラーパレット

SiSenseのチャートは、GitLabの標準的なカラーパレットを使用しています。

丸め

丸めるべきか、丸めないべきか？聴衆を知る。対象者や分析のユースケースに適した数値を提示する。

- **エグゼクティブレベルのチャートでは、通常、何桁もの精度を必要とせず、十の位、百の位、千の位への丸めで十分です。**
- **グラフによっては、合計が100%になるために、パーセンテージの有効数字1桁または2桁が必要な場合があります。通貨を除いて、一般的に数字は完全な形で表示する必要があります。**

通貨

- **すべての通貨は米ドルで表示されています。**
- **10,000ドル以上の場合は、各`000`が`k`に置き換えられ、例えば10,000ドルの代わりに10kとなります。**
- **1,000,000ドル以上では、各`000,000`が`m`に置き換えられ、例えば10,000,000ドルの代わりに10mとなります。**

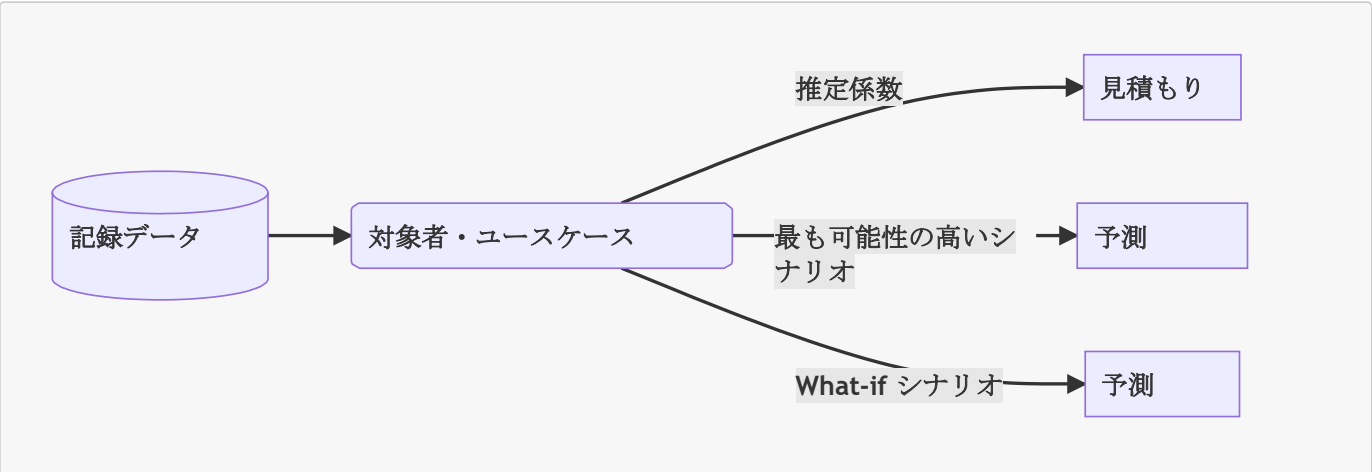
データと時間

会計日は、DIM_DATE ディメンション・テーブルから抽出する必要があります。日付の書式は、GitLabWritingStyleGuidelinesに準拠する必要があります。

- **日付はyyyy-mm-ddとなります。**
- **時刻は、UTCを使用した24時間表示です。 四半**
- **期はQn、例：Q1**
- **会計年度はFYyy、例：FY21**
- **年度と四半期はFYyy-Qn、例：FY21-Q2**

記録されたデータと計算されたデータ

データは、特定の対象者やユースケースのために、プレゼンテーションや報告のために準備されます。記録されたデータは、すべての計算されたデータの基礎となります。



- 記録されたデータ - 検証可能な情報源と観察可能なイベントに由来する「事実」のデータ。検証を助けるために、記録データには、データ作成者/ソースの名前、データ取得日時、イベントが発生した場所などの監査メタデータが一般的に含まれている。
- 推定データ - 記録されたデータに推定係数を加えて計算したデータ。推定係数は、通常、意味のある適切な期間の過去のデータトレンドに基づいています。エスティメーションは、あらゆる業界や分野で広く使用されています。
- 予測データ - 過去の記録データに加えて、「最も可能性の高い」将来のシナリオに基づいて評価した計算データ。予測はファイナンシャル・プランニングによく用いられる。
- 予測データ - 過去の記録データに基づいて計算されたデータに、「what-if」の将来シナリオに基づいた評価を加えたもの。

プレゼンテーション

記録されたデータには特別な表示は必要ありませんが、計算されたデータには特別な表示が必要です。計算さ

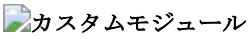
- れたデータは、グラフのタイトルや凡例などで、*常に*明確に表示する必要があります（例："Seats "の代

  わりに "Estimated Seats "を使用）。
- チャートの要素を明確に識別すること（例：同じチャートに計算タイプが混在している場合、異なるラインスタイルを使用すること

カスタムPythonモジュール

Pythonモジュールは、再利用可能で高品質でクリーンなコードを一箇所で維持するのに役立ちます。

カスタム**Python**モジュールを開始/変更するには、**Sisense**プロジェクトの**periscope/master**ブランチで**MR**を開始します。**Sisense**で利用可能なカスタムモジュールは、チャート作成ページの左サイドメニューでいつでも確認することができます。



カスタムモジュールについての公式ドキュメントはこちらをご覧ください。

---

layout: handbook-page-toc title:スノープラウの説明除雪機のインフラ管理

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。**

スノープラウの概要

Snowplowは、オープンソースのイベント分析プラットフォームです。**Snowplow**の**SaaS**を運営する事業体があり、彼らはオープンソース製品のコードも保守しています。**Snowplow**の一般的なアーキテクチャの概要は**GitHub**に掲載されており、どのように動作し、どのように設定されているのかという基本的な部分が詳しく説明されている。

2019年**6**月、**Snowplow**イベントの送信をサードパーティから**GitLab**が管理するインフラへの送信に切り替え、このページに記録しました。データチームの視点では、サードパーティの実装とあまり変わりませんでした。イベントはコレクターとエンリッチャーを経て、**S3**にダンプされます。

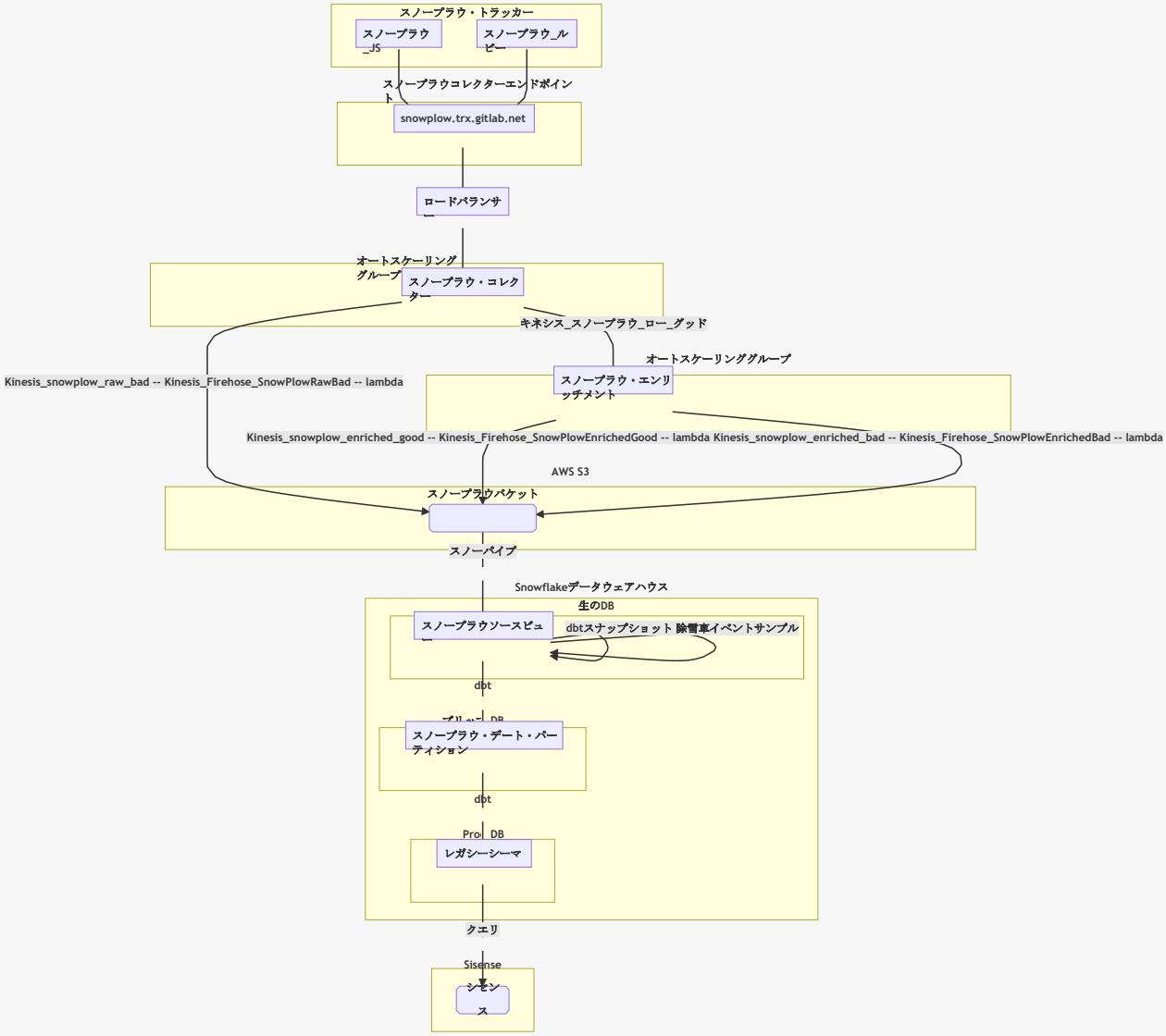
GitLabの実装

Snowplowのインフラをサードパーティのホスティングサービスから**1st-part**に移行するためのオリジナル設計書は、インフラ設計ライブラリに記載されています。これは構築を開始する前に書かれたもので、前提条件や設計上の決定事項の多くが含まれています。

Snowplowは**Terraform on AWS**で構築されており、**gitlab-com-infrastructure**プロジェクトでドキュ

メント化されています。セットアップの詳細については、**GitLabUnfiltered**の内部ビデオをご覧く

ださい。


スノープラウ・トラッカー
スノープラウ_JS
スノープラウ_ルビー
スノープラウコレクターエンドポイント
snowplow.trx.gitlab.net
ロードバランサー
オートスケーリンググループ
スノープラウ・コレクター
キネシス_スノープラウ_ロー_グッド
オートスケーリンググループ
スノープラウ・エンリッチメント
Kinesis_snowplow_raw_bad -- Kinesis_Firehose_SnowPlowRawBad -- lambda
Kinesis_snowplow_enriched_good -- Kinesis_Firehose_SnowPlowEnrichedGood -- lambda Kinesis_snowplow_enriched_bad -- Kinesis_Firehose_SnowPlowEnrichedBad -- lambda
AWS S3
スノープラウバケット
スノーパイプ
Snowflakeデータウェアハウス
生のDB
スノープラウソースビュー
dbtスナップショット 除雪車イベントサンプル
dbt
スノープラウ・デート・パーティション
dbt
Pro DB
レガシースキーマ
クエリ
Sisense
シセンス

S3

Enrichedイベントは**TSV**形式で`s3://gitlab-com-snowplow-events/output/`というバケツに格納されます。**Bad**イベントは`s3://gitlab-com-snowplow-events/enriched-Bad/`に**JSON**形式で保存されます。どちらのバケツにも、`/YYYY/MM/DD/HH/<data>`という日付形式に沿ったパスがあります。

データウェアハウス・スノ

ーパイプ

S3でイベントが利用できるようになったら、**Snowpipe**を使ってデータウェアハウスにインジェストします。これは、当社の**Snowflake**データウェアハウスの機能です。良いイベントパスと悪いイベントパスのために、**AmazonSQS**のイベントキューが設定されました。

Snowpipeが正常に動作するためには、**Snowflake**の中に「ステージ」が必要で、書き込み先のテーブルも必要です。良い**S3**パスと悪い**S3**パスには、それぞれ**Snowflake**内にステージがあります。これらはそれぞれ`gitlab_events`と`gitlab_bad_events`と名付けられています。これらは`LOADER`ロールが所有しています。

グッドイベントとバッドイベントの**create table**ステートメントは以下の通りです。

```
グッド・イベント
CREATE OR REPLACE TABLE
snowplow.gitlab_events (
    app_                    idVARCHAR,
    platformVARCHAR。
    etl_                    tstampVARCHAR,
    collector_tstamp        VARCHAR,
    dvce_created_tstamp     VARCHAR,
    eventVARCHAR,
    event_                  idVARCHAR,
    txn_                    idVARCHAR,
    name_                   trackerVARCHAR,
    v_                      trackerVARCHAR,
    v_                      collectorVARCHAR,
    v_                      etlVARCHARです。
    user_                   idVARCHAR,
    user_                   ipaddressVARCHAR,
    user_fingerprint VARCHAR,
    domain_                 useridVARCHAR,
    domain_sessionidx VARCHAR,
    network_                useridVARCHAR,
    geo_                    countryVARCHAR,
    geo_                    regionVARCHAR,
    geo_                    cityVARCHAR,
    geo_                    zipcodeVARCHAR,
    geo_                    latitudeVARCHAR,
    geo_                    longitudeVARCHAR,
    geo_region_             nameVARCHAR,
    ip_                     ispVARCHARです。
    ip_                     organizationVARCHAR,
```

```
    ip_                     domainVARCHAR,
    ip_                     netspeedVARCHAR,
    page_                   urlVARCHAR,
    page_                   titleVARCHAR,
    page_                   referrerVARCHAR,
    page_                   urlschemeVARCHAR,
    page_                   urlhostVARCHAR,
    page_                   urlportVARCHAR,
    page_                   urlpathVARCHAR,
    page_                   urlqueryVARCHAR,
    page_urlfragment VARCHAR,
    refr_                   urlschemeVARCHAR,
    refr_                   urlhostVARCHAR,
    refr_                   urlportVARCHAR,
    refr_                   urlpathVARCHAR,
    refr_                   urlqueryVARCHAR,
    refr_urlfragment VARCHAR,
    refr_                   mediumVARCHAR,
    refr_                   sourceVARCHAR,
    refr_                   termVARCHAR。
    mkt_                    mediumVARCHAR,
    mkt_                    sourceVARCHAR,
    mkt_                    termVARCHAR,
    mkt_                    contentVARCHAR,
    mkt_                    campaignVARCHAR,
    contextsVARCHAR,
    se_                     categoryVARCHAR,
    se_                     actionVARCHAR,
    se_                     labelVARCHAR,
    se_                     propertyVARCHARです。
    se_                     valueVARCHAR,
    unstruct_               eventVARCHAR,
    tr_                     orderidVARCHAR,
    tr_                     affiliationVARCHAR,
    tr_                     totalVARCHAR,
    tr_                     taxVARCHARです。
    tr_                     shippingVARCHAR,
    tr_                     cityVARCHAR,
    tr_                     stateVARCHAR,
    tr_                     countryVARCHAR,
    ti_                     orderidVARCHAR,
    ti_                     skuVARCHAR,
    ti_                     nameVARCHAR,
    ti_                     categoryVARCHAR,
    ti_                     priceVARCHAR,
    ti_                     quantityVARCHAR,
    pp_xoffset_             minVARCHAR,
    pp_xoffset_             maxVARCHAR,
    pp_yoffset_             minVARCHAR,
    pp_yoffset_             maxVARCHAR,
    useragentVARCHAR,
    br_                     nameVARCHAR,
    br_                     familyVARCHAR,
    br_                     versionVARCHAR,
    br_                     typeVARCHAR,
    br_                     renderengineVARCHAR,
    br_                     langVARCHAR,
    br_features_            pdfVARCHAR,
    br_features_flash VARCHAR,
    br_features_java    VARCHAR   、
    br_features_director  VARCHAR 、
    br_features_quicktime  VARCHAR 、
    br_features_realplayer VARCHAR、
    br_features_windowsmedia VARCHAR、
```

```
    br_features_gears VARCHAR。
```

```
    br_features_silverlight VARCHAR,
    br_
                            cookiesV
    ARCHAR,
    br_                     colordepthVARCHAR,
    br_                     viewwidthVARCHAR,
    br_                     viewheightVARCHAR,
    os_                     nameVARCHAR,
    os_                     familyVARCHAR,
    os_                     manufacturerVARCHAR,
    os_                     timezoneVARCHAR,
    dvce_                   typeVARCHAR,
    dvce_                   ismobileVARCHAR,
    dvce_                   screenwidthVARCHAR,
    dvce_                   screenheightVARCHAR,
    doc_                    charsetVARCHARです。
    doc_                    widthVARCHAR,
    doc_                    heightVARCHAR,
    tr_                     currencyVARCHAR,
    tr_total_               baseVARCHAR,
    tr_tax_                 baseVARCHAR,
    tr_shipping_            baseVARCHAR,
    ti_                     currencyVARCHAR,
    ti_price_               baseVARCHAR,
    base_                   currencyVARCHAR,
    geo_                    timezoneVARCHAR,
    mkt_                    clickidVARCHAR,
    mkt_                    networkVARCHAR,
    etl_                    tagsVARCHAR,
    dvce_sent_              tstampVARCHAR,
    refr_domain_            useridVARCHAR,
    refr_dvce_              tstampVARCHAR,
    derived_                contextsVARCHAR,
    domain_                 sessionidVARCHAR,
    derived_                tstampVARCHAR,
    event_                  vendorVARCHAR,
    event_                  nameVARCHAR,
    event_                  formatVARCHAR,
    event_                  versionVARCHAR,
    event_                  fingerprintVARCHAR,
    true_                   tstampVARCHAR,
    upload_                 atTIMESTAMP_NTZ(9) DEFAULT CAST(CURRENT_TIMESTAMP() AS TIMESTAMP_NTZ(9))
)

バッドイベント表
CREATE OR REPLACE TABLE
snowplow.gitlab_bad_events (
                jsontextVARIANTです
    upload_at TIMESTAMP_NTZ(9) DEFAULT CAST(CURRENT_TIMESTAMP() AS TIMESTAMP_NTZ(9))
);
```

TSVはCSVのようにわかりやすいものではないので、次のような記述でカスタムファイル形式を作成しました。

```
CREATE OR REPLACE FILE FORMAT snowplow_tsv TYPE = CSV FIELD_DELIMITER = '\'
```

実際にイベント用のパイプを作ったのは

```
CREATE OR REPLACE PIPE raw.snowplow.gitlab_good_event_pipe auto_ingest= TRUE
AS COPY INTO raw.snowplow.gitlab_events
    from (select $1, $2, $3, $4, $5, $6, $7, $8,
$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$20,$21,$22,$23,$24,$25,$26,$27,$28,$29,$30,$31,$32,$33,$
34,$35,$36,$37,$38,$39,$40,$41,$42,$43,$44,$45,$46,$47,$48,$49,$50,$51,$52,$53,$54,$55,$56,$57,$58,$
```

```
59,$60,$61,$62,$63,$64,$65,$66,$67,$68,$69,$70,$71,$72,$73,$74,$75,$76,$77,$78,$79,$80,$81,$82,$83,$
84,$85,$86,$87,$88,$89,$90,$91,$92,$93,$94,$95,$96,$97,$98,$99,100,101,102,103,104,105,106,107,108,1
09,110,111,112,113,114,115,116,117,118,119,120,121,122,123,124,125,126,127,128,129,130,131,ast(curre
nt_timestamp() as timestamp_ntz(9))AS upload_at
          FROM
    @raw.snowplow.gitlab_events)FILE_FORMAT =
    (FORMAT_NAME = 'snowplow_tsv') ON_ERROR =
    'skip_file';
```

これは、**JSON**がより良いフォーマットであることを強調しています。これはインフラの将来のイテレーションにな

るかもしれません。バッドイベントパイプは以下のように作成されます。

```
CREATE OR REPLACE PIPE raw.snowplow.gitlab_bad_event_pipe auto_ingest=TRUE
AS COPY INTO raw.snowplow.gitlab_bad_events (jsontext)
    FROM @raw.snowplow.gitlab_bad_events
    FILE_FORMAT = (TYPE = 'JSON');
```

パイプを表示するには

```
raw.snowplowでパイプを表示します。
```

パイプの説明に

```
DESCRIBE PIPE raw.snowplow.gitlab_good_event_pipe;
```

走っているパイプを一時停止すること。

```
ALTER PIPE raw.snowplow.gitlab_good_event_pipe SET PIPE_EXECUTION_PAUSED = TRUE;
```

パイプを強制的に再開すること。

```
SELECT system$pipe_force_resume('raw.snowplow.gitlab_good_event_pipe')。
```

パイプの状態を確認するには

```
SELECT system$pipe_status('raw.snowplow.gitlab_good_event_pipe')です。
```

ステージを強制的に更新し、**snowpipe**が古いイベントを拾うようにします。

```
ALTER PIPE gitlab_good_event_pipe refresh;
```

dbt

RAWデータベースから**PROD**にデータを実体化して問い合わせを行うために、**dbt**内にパーティショニング戦略を実装しました。デフォルトでは、除雪機モデルと**Fishtown**除雪機パッケージは、**PREP**データベースの現在の月にスコープされたスキーマに書き込みます。**2019**年**7**月の場合、スキーマは`snowplow_2019_07`となります。

各月のパーティション内では、基本モデルとパッケージで生成されたモデルのすべてが、パーティションの日付に一致する派生タイムスタンプを持つすべてのイベントに対して書き込まれます。実行時にdbtに変数を渡すことで、異なる月別パーティションを生成することができます。

```
--vars '{"year":"2019", "month":"01", "part":"2019_01"}'
```

バックフィルはAirflowで行います。dbt_snowplow_backfillDAGは、2018年7月から当月までの各月のタスクを生成します。

Do Not Track

当社の除雪機のトラッキング設定および特定の実装では、ユーザーのブラウザにDNT（Do Not Track）ヘッダーが存在する場合は常にそれを尊重します。

layout: handbook-page-toc title:"SQLスタイルガイド" 説明"私たちにはリンターがいないので、この SQL スタイルガイドを施行するのは私たちの集団の責任です。"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- TOC {:toc .toc-list-icons .hidden-md .hidden-lg}。

SQLスタイルガイド

私たちにはリンターがいないので、このスタイルガイドを実行するのは*私たちの責任*です。

フィールドの名称と参照方法

- フィールド名はすべて小文字にしてください。
- ID、名前、またはtypeのような一般的に曖昧な値は、常にそれが何を識別または命名しているかを前置する必要があります。

```
グッド
セレク
ト
       idAS account_id
  、name AS
  account_name、type AS
  account_type。
  ...

バッド
セレク
ト
  id、
  name
  、
  type
  。
  ...
```

- 異なるソースからのデータに結合する場合、フィールドの前にデータソースを付ける必要があります。
  曖昧さを避けるために、sfdc_account_id

```
グッド
セレク
ト
  sfdc_account.account_     idAS
  sfdc_account_id, zuora_account.account_id AS
  zuora_account_id
FROM sfdc_account
LEFT JOIN zuora_account ON ...
```

```
-- 悪い
```

```
SELECT
  sfdc_account.account_id,
  zuora_account.account_id AS zuora_id
FROM sfdc_account
LEFT JOIN zuora_account ON ...
```

- テーブルを結合して両方のカラムを参照する場合、エイリアスではなく完全なテーブル名を参照することを強く推奨します。テーブル名が長い（〜20）場合は、可能であればCTEの名前を変更し、最後に何かわかりやすい名前にエイリアスすることを検討します。

- すべてのフィールド名は蛇足である
  グッドセ

```
グッド
セレク
ト
  dvcecreatedtstamp AS device_created_timestamp
  date_details.fiscal_quarter_name,
  cost_category.cost_category_level_1,
  cost_category.cost_category_level_2 があ
  ります。
FROM budget_forecast_cogs_opex
LEFT JOIN date_details
 ON date_details.first_day_of_month = budget_forecast_cogs_opex.accounting_period LEFT
JOIN cost_category
 ON budget_forecast_cogs_opex.unique_account_name = cost_category.unique_account_name

-- OKだが、好ましくない。エイリアシングの代わりにCTEの名称変更を検討する
SELECT
  bfcopex.account_id,
  -- さらに15個のカラム
  date_details.fiscal_year,
  date_details.fiscal_quarter,
  date_details.fiscal_quarter_name,
  cost_category.cost_category_level_1,
  cost_category.cost_category_level_2
FROM budget_forecast_cogs_opex bfcopex LEFT
JOIN date_details
 ON date_details.first_day_of_month = bfcopex.accounting_period LEFT
JOIN cost_category
 ON bfcopex.unique_account_name = cost_category.unique_account_name

バッド
セレク
ト
  a.*,
  b.fiscal_year,
  b.fiscal_quarter,
  b.fiscal_quarter_name,
  c.cost_category_level_1,
  c.cost_category_level_2 の15列
  を追加。
FROM budget_forecast_cogs_opex a
LEFT JOIN date_details b
 ON b.first_day_of_month = a.accounting_period LEFT
JOIN cost_category c
 ON b.unique_account_name = c.unique_account_name
```

```
FROMテーブル

バッド
セレク
ト
  dvcecreatedtstamp AS DeviceCreatedTimestamp
FROM table
```

- ブール型のフィールド名は、has_、is_、does_ のいずれかで始まります。

```
グッドセ
レクト
  deleted AS is_deleted,
  slaAS has_sla
FROMテーブル

バッド
セレク
ト
  削除、スラ
FROMテーブル
```

- ソースデータを変換する際、大文字小文字を区別する列や、「$」や「_」とは異なる特殊文字を含む列を識別するために二重引用符を使用します。大文字のフィールド名に二重引用符必要ありませこれはSnowflake識別子が内部的に処理ため

```
--良い
SELECT "First_Name_&_" AS first_name,

-- 悪い
SELECT "FIRST_NAME" AS first_name,
```

日付

- タイムスタンプは_atで終わります。例：deal_closed_at、常にUTCでなければなり
- ません。 日付は_dateで終わります。例：deal_closed_date
- 月はそのように表示し、常に日付形式に切り詰める必要があります（例：deal_closed_month）。
- 列名に「日付」や「月」などのキーワードを使わないようにする。
- DAYOFWEEK(created_at) >」のように、date_partよりも明示的な日付関数を優先し、extractよりもdate_partを優先する。DATE_PART(dayofweek, 'created_at') > EXTRACT(dow FROM created_at)
  - date_trunc('month', created_at)ではカレンダーの月（'2019-01-01'なら'2019-01-25'）が生成され、SELECT date_part('month', '2019-01-25'::date)では数字の1が生成されることに注意してください。

DATEDIFFを使うと、直感的でない結果になることが多いので注意が必要です。

- 例えば、SELECT DATEDIFF('days', '2001-12-01 23:59:59.999', '2001-12-02 00:00:00.000')では、次のように返されます。
  タイムスタンプが1ミリ秒違っていても、1になります。
- 同様に、SELECT DATEDIFF('days', '2001-12-01 00:00:00.001', '2001-12-01 23:59:59.999') return 0
  タイムスタンプがほぼ丸一日離れているにもかかわらず。
- DATEDIFF関数で適切な間隔を使用することで、正しい結果を得ることができます。例えば、DATEDIFF('days', '2001-12-01 23:59:59.999', '2001-12-02 00:00:00.000')では、1日分の
  間隔で、DATEDIFF('ms', '2001-12-01 23:59:59.999', '2001-12-02 00:00:00.000')では1
  ミリ秒の間隔で

サブクエリではなくCTE（Common Table Expressions）の使用

- CTEは、SQLをより読みやすくし、パフォーマンスを向上させます。

- 他のテーブルを参照するには、CTE を使用します。CTEはクエリの先頭
- に配置する必要があります（インポートステートメントと考えてくださ

  い）。
- 性能が許す限り、CTEは単一の論理的な作業単位を実行すること CTEの名称は、明
- 確であると同時にできる限り簡潔であること
  - replace_sfdc_account_id_with_master_record_id のような長い名前は避け、CTE にコメントを入れた短い名前を使うようにしましょう。これにより、結合時のテーブルのエイリアスを避けることができます。
- 紛らわしいロジックや注目すべきロジックを持つCTEは、ファイル内でコメントし、dbtドキュメ
- ントで文書化すべき モデル間で重複しているCTEは、それぞれのモデルに引き抜くべき
- クエリステートメントの上下に空の行を残す CTEは以下の
- ようにフォーマットしてください。

```
WITH events AS ( -- これらのselect文をimport文と考えてください。

  ...

), filtered_events AS ( -- CTEのコメントはここへ)

  ...

)

SELECT * -- 最後のモデルについては、常に「select * from final」を目指すべきです
FROM filtered_events
```

一般

- CTE内では、SQL文全体をスペース4個分インデントする必要があります。

```
--良い
WITH my_data AS (

    SELECT *
    FROM prod.my_data
    WHERE filter = 'my_filter'

)

-- 悪い
WITH my_data AS (

  SELECT *
  FROM prod.my_data
  WHERE filter = 'my_filter'

)
```

- クエリ内のインデント（例：列、JOIN句、複数行のGROUP BYなど）は、スペース2個分のインデントが必要です。

```
グッド
セレク
ト
  column_name1,
  column_name2,
```

```
  カラム名3 FROM
table_1 JOIN
table_2
  ON table_1.id =
table_2.id WHERE clouds =
true
  AND gem =
true GROUP BY
1,2,3
HAVING カラムネーム1 >
  0 AND カラムネーム2 >
  0

バッド
セレク
ト
    カラム名1、カ
    ラム名2、カラ
    ム名3
FROM
table_1
JOIN
table_2
    ON table_1.id =
table_2.id WHERE clouds =
true
    AND gem =
true GROUP BY
1,2,3
HAVING カラムネーム1 > 0
    AND カラムネーム2 >
    0
```

- タブは使用せず、スペースのみを使用してください。編集者は、タブをスペースに変換するように設定する必要があります。
- **SQL**の行は**80**文字以内にしてください。
- カンマは、WHEREの一時的なフィルターを除いて、行末（**EOL**）に右のカンマを入れてください。
節で特定の値を指定します。

```
--良い
SELECT
  削除され      AS is_deleted, -- EOL右コンマ
  た            AS account_id
  accountId
FROMテーブル
WHERE is_deleted = false
AND account_id NOT IN (
                        '232'
                        234」 -- 左コンマ
                        , '425'
                       )

-- 悪い
SELECT
  削除され      AS is_deleted, -- EOL右コンマ
  た            AS account_id
  accountId
FROMテーブル
WHERE is_deleted = false
AND account_id NOT IN ('232', '234',
  '425')
```

- `SELECT`の際には、常に各列を独立した行にします。ただし、`SELECT *`の場合は**1**つの行になります。
- `DISTINCT`は、`SELECT`と同じ行に含まれている必要があります。
- フィールド名やテーブル名を投影する際には、`AS`キーワードを使用する必要があります。
- `AS`を使用してエイリアシングを行う場合は、元のカラム名を**1**本の縦線に、`AS`キーワードを別の縦線に配置するようにしてください。

- フィールドは集計／窓関数の前に記載すること 数値による順序付けと
- グループ化（例：GROUP BY 1, 2）が望ましい
    - dbtモデルで3つ以上のカラムでグループ化する場合、dbt-utils group_byマクロを使用してく
- ださい。 WHEREとHAVINGのどちらかで十分な場合はPreferしてください。
- 例えば、data_by_row['id']::bigint as id_valueのように、ブラケット構文を使用してJSONにアクセスすることができます。
- 結合でUSINGを使うとSnowflakeで不正確な結果が出るので絶対に使わないでください。このトピックに関するフォーラムのディスカッションを表示するには、アカウントを作成してください。
- UNIONよりもUNION ALLを優先してください。これは、UNIONが上流のデータインテグリティの問題を示している可能性があり、他の場所で解決した方が良いからです。
- <> よりも != を優先してください。これは、他のプログラミング言語では != の方が一般的であり、「not equal」のように読めるため、私たちはこのような言い方をすることが多いからです。
- パフォーマンスを考慮する。LIKEとILIKE、ISと=、NOTと！と<>の違いを理解する。適切に使用する
- LOWER(column) LIKE '%match%' よりも、column ILIKE '%Match%' を優先します。これにより、誤った大文字が予期せぬ結果を引き起こす可能性が低くなります。
- DRYプリンシパルに慣れ親しむ。dbtではCTE、jinja、マクロを、Sisenseではスニペットを活用する。同じ行を2回入力した場合、2箇所で管理する必要がある
- は、コードの行数を減らすために最適化することはありません。改行は安い。ブレインタイムは高価

データタイプ

- デフォルトのデータ型を使用し、エイリアスは使用しないでください。詳細は、Snowflake のデータタイプの概要をご覧ください。デフォルトは
    - DECIMAL、NUMERIC、INTEGER、BIGINTなどの代わりにNUMBER。
    - DOUBLE、REALなどの代わりにFLOAT。
    - STRING、TEXTなどの代わりに
    - VARCHARDATETIMEではなくTIMESTAMP

ただし、タイムスタンプの場合は例外です。TIMEよりもTIMESTAMPを優先します。なお、TIMESTAMPのデフォルトはTIMESTAMP_NTZです。のように、タイムゾーンが含まれていません。

機能

- 関数名とキーワードはすべて大文字にする IFNULL TO
- NVLを優先する
- 1行のCASE文よりもIFFを優先する
- is_less_than_tenとしてbooleanステートメント(amount < 10)を選択するよりもIFFを優先する
- 繰り返しの多いCASE文を可能な限り簡略化してください。

```
OK
CASE
  WHEN Field_id = 1 THEN 'date'
  WHEN Field_id = 2 THEN 'integer'
  WHEN Field_id = 3 THEN
  'currency' WHEN Field_id = 4
  THEN 'boolean' WHEN Field_id = 5
  THEN 'variant' WHEN Field_id = 6
  THEN 'text'
END AS field_type
```

```
より良いケース
field_id
  WHEN 1 THEN 'date'
  WHEN 2 THEN 'integer'
  WHEN 3 THEN
  'currency' WHEN 4
  THEN 'boolean' WHEN 5
  THEN 'variant' WHEN 6
  THEN 'text'
END AS field_type
```

JOINs

- 例えば、JOINの代わりにLEFT JOINを使うなど、結合の際には明示的にしてください。(デ
- フォルトの結合はINNERです) 結合時にテーブル名をカラムの前に付け、そうでなけれ

  ば省略します。
- **FROM**テーブルを先に、**JOIN**テーブルを後にして結合の順番を指定します。

```
良いFROMソ
ース
LEFT JOIN other_source
  ON source.id =
other_source.id WHERE ...

-- 悪い
FROMソース
LEFT JOIN other_source
  ON other_source.id =
source.id WHERE ...
```

コード例

- まとめてみました。

```
WITH my_data AS

    ( SELECT * )
    FROM prod.my_data
    WHERE filter = 'my_filter'

), some_cte AS (

    SELECT DISTINCT
      id,
      other_field_1,
      other_field_2
    FROM prod.my_other_data

), final AS

    ( SELECT
      data_by_row['id']::NUMBER AS id_field,
      field_1                   AS detailed_field_1です。
      field_2                   AS detailed_field_2,
      detailed_field_3,
      CASE
        WHEN cancellation_date IS NULL AND expiration_date IS NOT
          NULL THEN expiration_date
        WHEN cancellation_date IS
          NULL THEN start_date + 7
```

```
          ELSE cancellation_date
        ENDAS cancellation_date, LAG(detailed_field_3)
        OVER (
          パーティション・バイ
            id_field,
            detailed_field_1
          ORDER BY cancellation_date
        )                          AS previous_detailed_field_3,
        SUM(field_4)               AS field_4_sum,
        MAX(field_5)               AS
      field_5_max FROM my_data
      LEFT JOIN some_cte
        ON my_data.id =
      some_cte.id WHERE field_1 =
      'abc'
        AND (field_2 = 'def' OR field_2 =
      'ghi') GROUP BY 1, 2, 3, 4, 5
      count(*) > 1 order
      by 4 desc
  )

  SELECT *
  FROM final
```

コメント

- モデル内で一行コメントを作成する場合は `--` 構文を使用します。 モデル
- 内で複数行コメントを作成する場合は `/* */` 構文を使用します。
- コメントを書くときは、**1**行の制限を守ってください。コメントが長すぎる場合は、新しい行に移動するか、モデルのドキュメントに移動してください。
- **dbt**モデルのコメントは、モデルのドキュメントに記載されるべきです。
- **SQL**で計算する場合は、何が起こっているのかを簡単に説明し、その指標を定義したハンドブック（およびその計算方法）へのリンクを表示する必要があります。
- `TODO`コメントを残すのではなく、改善のために新しい課題を作る

その他のSQLスタイルガイド

- **Brooklyn Data Co**
- **FishtownAnalytics**
- **Matt Mazur**
- **Kickstarter**

---

layout: handbook-page-toc title:"Data & Analytics Team Operating Principles" description:"GitLab データ＆アナリティクスチーム運営原則ハンドブック"

このページについて

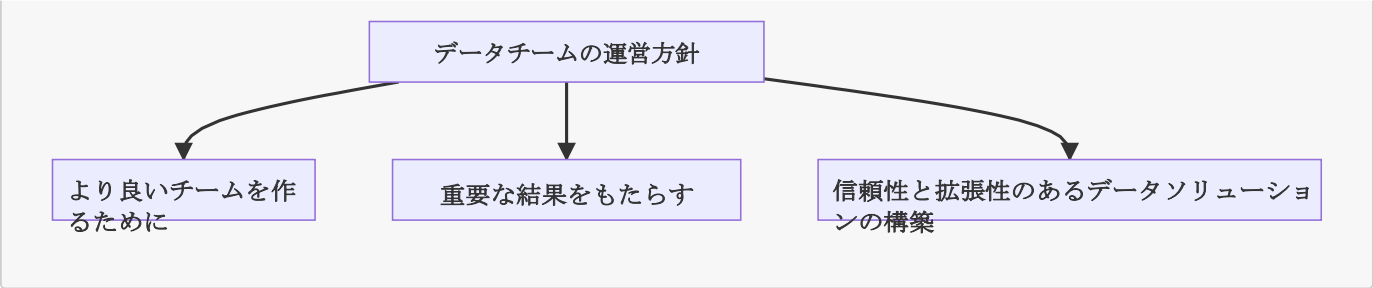
{:.no_toc .hidden-md .hidden-lg}。

{::options parse_block_html="true" /}。

データ＆アナリティクスの動作原理

私たちは、世界で最も革新的な企業のデータ＆アナリティクスのセンター・オブ・エクセレンスであることを誇りに思うと同時に、謙虚に受け止めています。

データチームには、3つの運営方針があります。



より良いチームを作るために

- **CREDIT**
- 私たちは、すべての人を尊重しています。
- 私たちは、**GitLab**の成功に向けて最善の意思を持
- っています。 私たちは、自分自身とお互いに正直
  です。
- 私たちは意見を異にすることを許可されていますが、コミットメ
- ントに向けて行動しなければなりません 難しい会話から逃げな
  い
- 私たちはチームとして一緒にいる
- チームでミスに取り組み、改善を目指す

重要な結果をもたらす

- 第一にデータドリブン、第二にデータ・インフォームド、第三に原則と
- ストーリー 私たちは、**Built to Last, not built too fast**を信条として
  います。
- 私たちは成長思考で行動し、常に水準を高めていきます 私たちは活動と
- 進歩を混同しません
- 自分の仕事がどのように戦略的イニシアティブと連携し、ビジネスパートナーがインパクトのある結果を出すために
  どのように役立つかを知っています。
- 私たちは、文脈、事実、洞察を提供するために余分な努力をします。
- 前もって期待値を定義します。範囲、時間、成功基準
- 重要な取り組みを定期的に進めるために、ビジネスパートナーと積極的に関わっています。

信頼性と拡張性のあるデータソリューションの構築

- データセキュリティとデータ品質に対する責任があります。 十
- 分にテストされた正確なデータソリューションでリードします
  。
- 私たちが作成したプロダクションソリューションを運用・保守す
- る 私たちは、**GitLab +2**年のためにデザインし、今日の**GitLab**の
  ために提供する
- ベンダー・パートナーの能力とサービスを最大限に活用する
- 私たちは、複雑な結果を調整するための最良の方法は、それを描くことだと信じています。
- 自動化されたテストは最高のテストであり、私たちはデータ提供プロセスのすべてのステップでテストを実施します。

データチームが持つその他の信念は以下の通りです。

- すべてがコードで定義され、バージョン管理されるべきである。
- データの実装には、**DevOps**のベストプラクティスをワークフローに統合する必要があります。
- 高品質で保守可能なコードベースを持ちながら、各ビジネス機能と提携します。 データのセキュリ
- ティとデータの機密性を考慮しながら、コードをオープンソース化します。
- 企業内のすべての分析的な質問に対する真実のソースは単一であり、最も徹底したデータの洞察を提供するために、
  複数の視点を表現することも重要である
- 糊付け作業はチームの健全性を保つために重要であり、その価値に応じて個別に評価されます。私たちは全員が貢献す
  ることを期待しています。
- 限られたリソースを、データが最大の効果を発揮する場所に集中させる
- すべてのビジネスユーザーは、簡単な統計の解釈と計算方法を学ぶことができます。

layout: handbook-page-toc title:"Data Team Programs" 説明"データプログラム"

このページについて

{:.no_toc .hidden-md .hidden-lg}。

- **TOC {:toc .hidden-md .hidden-lg}。**

データプログラムページへようこそ

データプログラムのページでは、オンボーディングから日々のオペレーションに至るまで、私たちがサポートする様々なプログラムに関する情報が掲載されています。

データプログラム

| プログラム名 | タイプ | 目的 | :--- | :--- | | 財務向けデータ | 運用 | 財務アナリストを支援するための情報 | | プロダクトマネージャー向けデータ | 運用 | プロダクトマネージャーを支援するための情報 | | マーケティングアナリスト向けデータ | マーケティングアナリストを支援するための情報 | | セールスアナリスト向けデータ | セールスアナリストを支援するための情報 | | データオンボーディング | **1-time/ad-hoc** | 中央データチームの内外を問わず、データエンジニア、アナリスト、または開発者のトレーニング、プロビジョニング、およびイネーブルメントを行います。**|** データ**Slack**チャンネル | 運用 | 全社のチームメンバーの質問に対応するための**#data**チャンネルの監視。**|** データトリアージ | 運用 | データプラットフォームが分析のために利用可能であることを保証するための日々のプロセス。**|**

データオンボーディング

GitLabにオンボーディングし、データプログラムでエンジニア、アナリスト、デベロッパーとして働く場合は、以下の手順に従ってください。

1. **GitLabDataAnalytics**で`Data Onboarding`テンプレートを使って新しい課題を開きます。
2. **課題にわかりやすい名前をつける：**`Your Name - Data Onboarding`
3. マネージャーに課題を割り当て、関連するコンテンツを追加/削除する