Google Cloud

進化する BigQuery

～ エンタープライズ DWH における 9 つのポイント ～

Tetsuki Takamura

Data Analytics Specialist - Customer Engineer

Google Cloud Japan

スピーカー自己紹介

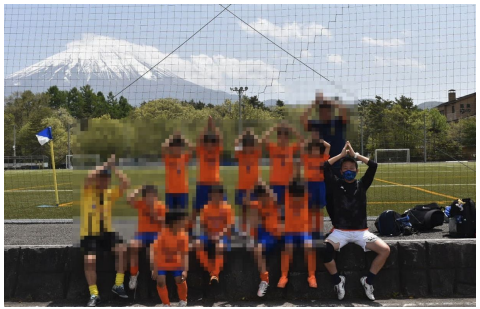
Tetsuki Takamura
(tetsuki@)

Google Cloud Japan
Solution & Technology
Data Analytics Specialist

日系 SIer 企業を経て、2021 年より現職

Google Cloud の Data Analytics 領域の Specialist として、
お客様のデータ活用を技術観点から支援

週末はサッカーコーチ (小学 3 年生担当)



BigQuery に
どんなイメージを
お持ちでしょうか？

BigQuery って、データサイエンティストが
アドホックに分析する OLAP 基盤としては
いいと思うんだけど、

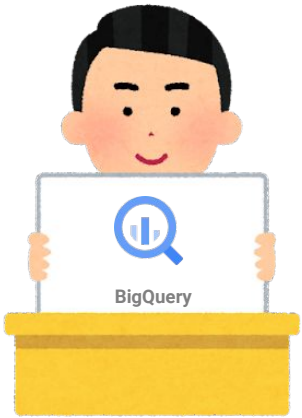
エンタープライズ用途の DWH として
使うのは向いてないんじゃないの？

本セッションの目的

BigQuery の近年のリリースにより強化された機能を

踏まえて、**エンタープライズ DWH としての価値**を

お客様やパートナーの皆様にお伝えするためのセッションです



「へー、BigQuery って、それできるようになったんだ！」

「DWH / Datalake の更改の際に候補にしてみよう」

※本日はハイライトのみお伝えします。各機能の詳細は、公式　HP や Blog 等をご覧ください

エンタープライズ DWH の 9 つのポイント

エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2.
3.
4.
5.
6.
7.
8.
9.

低レイテンシ・大量同時接続への対応

懸念点

- BI レポートは**低レイテンシ・大量同時接続数**のケースが多い
  (例 : 月曜日にダッシュボードの参照で 2,000-3,000 位のアクセスあり)

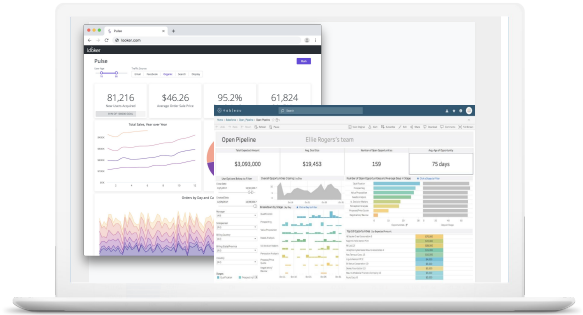
- OLTP データベースなら処理できるけど、**BigQuery は苦手**な分野じゃないの？

低レイテンシ・大量同時接続への対応

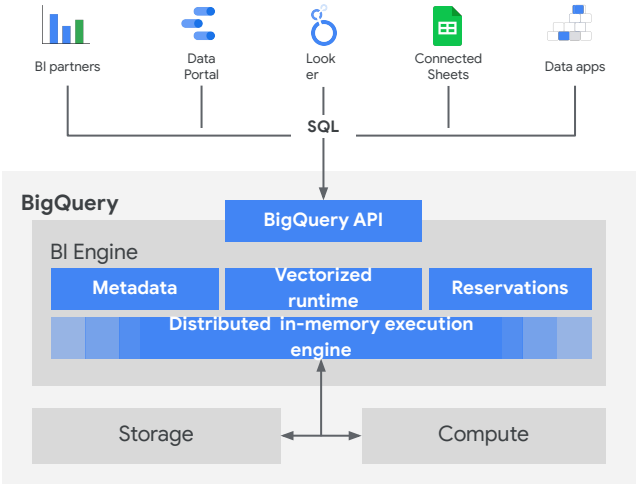
BigQuery BI Engine SQL Interface

GA (2021-12-14)

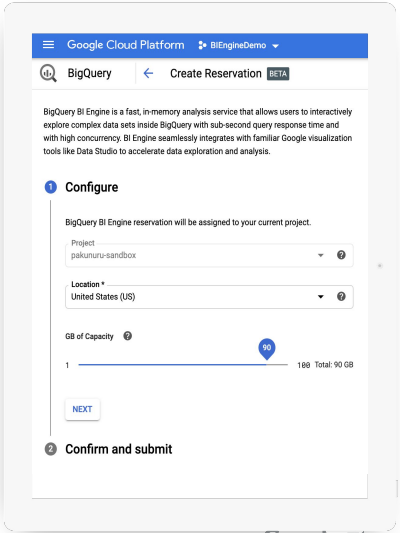
Sub-second queries



Simplified architecture



Smart tuning



エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3.
4.
5.
6.
7.
8.
9.

パフォーマンスチューニングの選択肢

懸念点

- 性能問題が発生した時、BigQuery でパフォーマンスチューニングに関して何ができるか？
- Enterprise な移行案件だと、**スキーマの変更やクエリの書き換えはそんなに簡単にできない**んだけど
- いざって時にアクセラレートできる方法があれば、性能試験で問題が発生しても安心できる

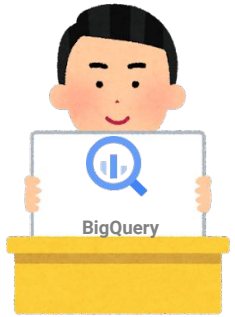
パフォーマンスチューニングの選択肢

BigQuery のチューニングバリエーション

- SQL チューニング
- テーブル・スキーマ最適化
- (進化) Materialized View (V2)
- Partitioning
- Clustering
- (進化) BI Engine SQL Interface
- (進化) Reservations (Flex slots)

query 書き換え必要

query 書き換え不要



エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4.
5.
6.
7.
8.
9.

懸念点

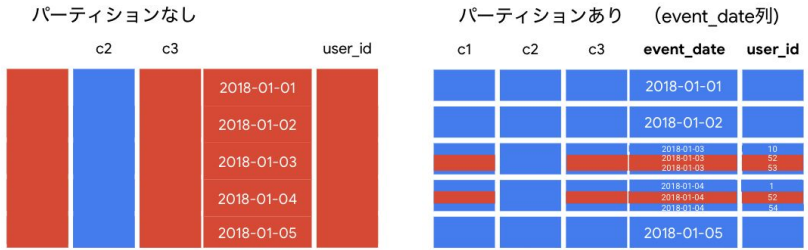
- 必要なデータだけにアクセスしてコストとパフォーマンスを最適化したいんだけど、**BigQueryって全表走査 (フルスキャン) してるから効率悪い**んじゃないの？

コスト効率の良いクエリ処理

これまでの BigQuery

Partitioning と Clustering

- BigQuery では、**Partitioning** 及び **Clustering** により読み取りデータ量を最小化した上で、独自の分散処理技術 (Borg / Dremel / Colossus / Jupiter) を用いて高速処理を実現します
- 結果、分散キーの設計や管理を行う必要がなく、高いパフォーマンスを実現しています



さらに進化

Native JSON

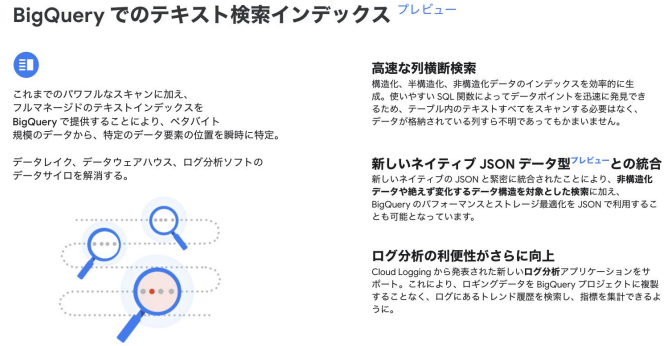
Preview (2022-01-06)

- Native JSON対応により、非構造化データや絶えず変化するデータ構造を対象とした検索が可能に

テキスト検索インデックス

Preview

- 全文検索とインデックス機能 (String 列限定 / SEARCH 関数で利用)



エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5.
6.
7.
8.
9.

ワークロード管理

懸念点

- BigQuery は、多くの利用者が同時にアクセスしても
  ストレスなく使える反面、ワークロードの**制御**が難しい

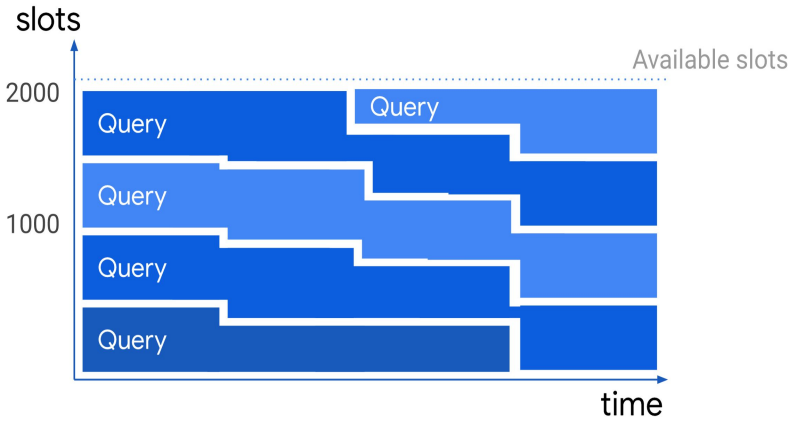
- 例えば、重要なワークロードに優先的にリソース
  割り振って**コスト効果を高めたい**んだけど、
  BigQuery って対応できる？

ワークロード管理

これまでの BigQuery

ダイナミック クエリ プラン & フェア スケジューラ

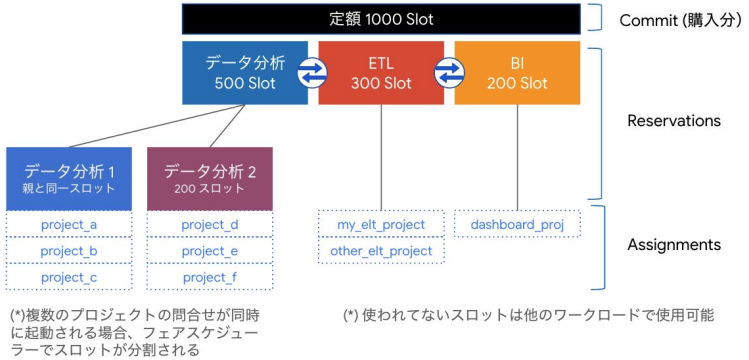
- 分析者にストレスを感じさせない
- 動的にクエリプランを生成し、実行中も最適化
- 有効に使えるリソース (Slot) を常に最大限活用



さらに進化

Reservation (Slot) による ワークロード管理

- プロジェクトレベルで Slot の割当量をコントロール
- 割り当てするジョブタイプ
  (QUERY, PIPELINE, ML_EXTERNAL) も指定可能



Query Queues

Preview

- ジョブキューとリソース割り当てを可能に

エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5. 柔軟なバックアップ・コピー
6.
7.
8.
9.

懸念点

- **BigQuery の Time Travel は便利だけど、**
  7 日までしかできないんだよなぁ

- それ以前の時点のデータのスナップ ショットを
  取ったり、テーブルを複製する良い方法はないの？

柔軟なバックアップ・コピー

これまでの BigQuery

データセットやテーブルのCopy

GCS へのデータExport

Time travel (FOR SYSTEM_TIME AS OF)

- 自動で 7 days 以内の任意のデータが復元可能（無料）

Time travel

Read data from any time within the last 7 days.

```
SELECT x, y
FROM dataset.table
FOR SYSTEM_TIME AS OF
 TIMESTAMP_SUB(CURRENT_TIMESTAMP(), INTERVAL 3 DAY)
```

＋

さらに進化

Table Snapshot

GA (2021-10-28)

- 特定ポイントのデータの Snapshot を保持 (PITR)

```
CREATE SNAPSHOT TABLE
  myproject.library_backup.books
  CLONE myproject.library.books
  OPTIONS(expiration_timestamp = TIMESTAMP "2022-04-27 12:30:00.00-08:00")
```

Table Clone

Preview (2022-02-15)

- Snapshot の可変バージョン
- 本番環境の変更テストなど柔軟に

スナップショット

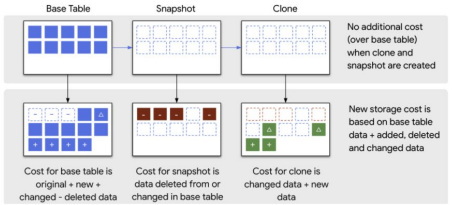
ベーステーブルの、変更不可能な読み取り専用バージョン。タイムトラベルの時間枠を超える論理バックアップまたはクエリに有用です。

クローン

ベーステーブルの可変バージョン。読み取り/書き込み/スキーマを進化させることができ、本番環境の変更テストに有用です。

低コスト

ベーステーブルが変更または削除されるか (一意のバイトに対して課金)、テーブルクローンが変更されるまで、テーブルスナップショットおよびクローンに追加のストレージコストは発生しません。



エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5. 柔軟なバックアップ
6. 環境の複製・クローン
7.
8.
9.

懸念点

- 大規模開発で、同時に複数チームで
  開発・テストするから、環境のクローンを作りたい

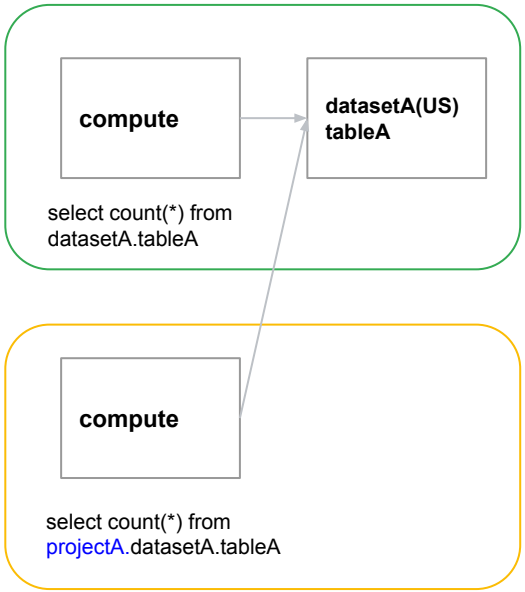
- 他の DB だと新しいインスタンスを作成して
  環境作れるけど、**BigQuery だと開発環境に容易に**
  クローンを作れないんだよな

BigQuery 環境構成パターン

: project A　: project B　: project C

(A) データー元化

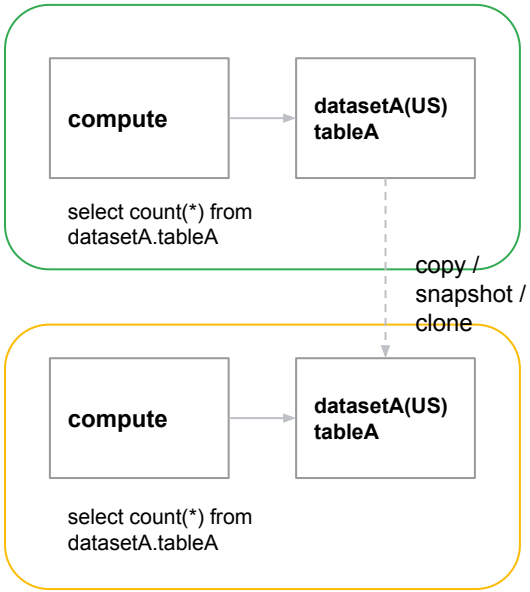
(ex : 別組織にデータを参照させる)



データの複製コストなく
共有が可能

(B) 環境クローン

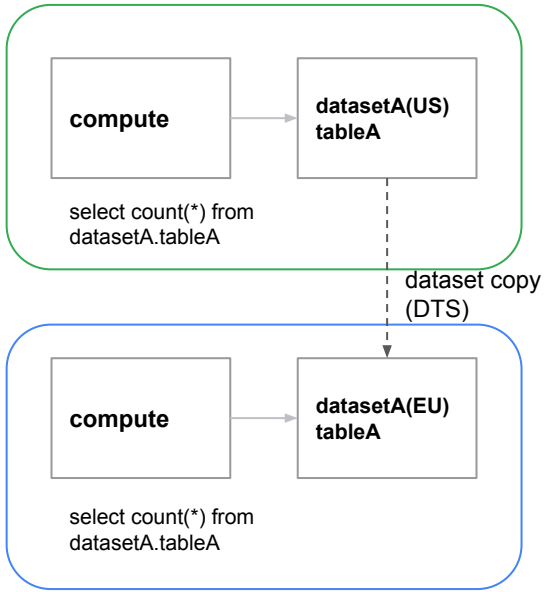
(ex : 開発環境を複製し、複数チームで同時開発)



同一の dataset 名・table 名で
query 実行可能

(C) リージョンクローン

(ex : 別リージョンで同じ dataset名を利用)



同一の dataset 名・table 名で
query 実行可能

エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5. 柔軟なバックアップ
6. 環境の複製・クローン
7. トランザクション対応
8.
9.

懸念点

- エンタープライズ用途だと、BI レポートで参照中の
  データを更新するケースもあるので、**複数の更新処理をまとめて Commit or Rollback** するって普通に必要に
  なるんだけど・・・　BigQuery って、
  トランザクションは未対応ですよね？

トランザクション対応

これまでの BigQuery

- DML ステートメントは暗黙の
  トランザクション
  （終了時点で自動コミット）
- 複数ステートメントの
  トランザクション対応は非対応

＋

さらに進化

- **マルチ ステートメント トランザクション**

Preview (2021-07-09)

  - BEGIN、COMMIT、ROLLBACK を使って
    複数のステートメントにまたがる
    トランザクションを制御可能に
  - 分離レベル = **Snapshot Isolation**

```
BEGIN

  BEGIN TRANSACTION;
  INSERT INTO mydataset.NewArrivals
    VALUES ('top load washer', 100, 'warehouse #1');
  -- Trigger an error.
  SELECT 1/0;
  COMMIT TRANSACTION;

EXCEPTION WHEN ERROR THEN
  -- Roll back the transaction inside the exception handler.
  SELECT @@error.message;
  ROLLBACK TRANSACTION;
END;
```

Multi-statement transactions | BigQuery

エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5. 柔軟なバックアップ
6. 環境の複製・クローン
7. トランザクション対応
8. デプロイ・バージョン管理
9.

懸念点

- インフラ、テーブルスキーマ、データ変換スクリプト、BI レポート向けのメトリック・・・
  色んな管理対象があるが、これらの構成管理、どうすれば良いのか・・・

デプロイ・バージョン管理

	管理対象 (例)	ツール (例)
Semantic	• Dimension、Measure、フィールド定義	LookML
Application	• バッチ処理等の変換 SQL スクリプト • ETL の Transformation のパイプライン	Dataform, dbt
Data	• BigQuery のテーブル DDL • BigQuery の各種メタデータ情報	INFORMATION_SCHEMA
Infrastructure	• Google Cloud サービスの構成、コンフィギュレーション	Terraform

エンタープライズ DWH 9 つのポイント

1. 低レイテンシ・大量同時接続への対応
2. パフォーマンスチューニングの選択肢
3. コスト効率の良いクエリ処理
4. 高度なワークロード管理
5. 柔軟なバックアップ
6. 環境の複製、クローン
7. トランザクション対応
8. デプロイ・バージョン管理
9. 監視・ログ分析

懸念点

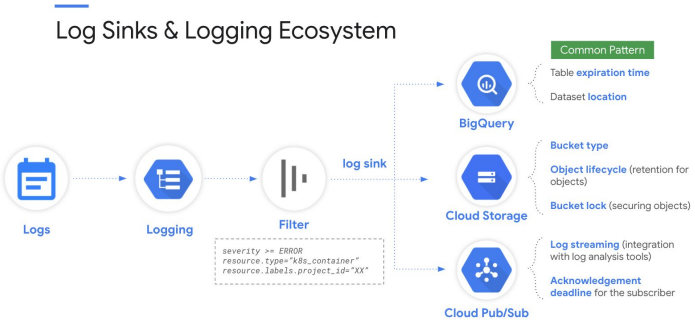
- Cloud Logging の UI がいまいち慣れなくて、BigQuery にデータ連携してログ分析してるんだけど、Log Sink しなきゃいけないし、コストもかかるのが気になる

監視・ログ分析

これまでの BigQuery

Log Sinks & Logging Ecosystem

- Cloud Logging によるモニタリングでは、**Log sink** を使用することで、フィルタをかけながら GCS, BigQuery などの別の Product にログデータを転送することが可能

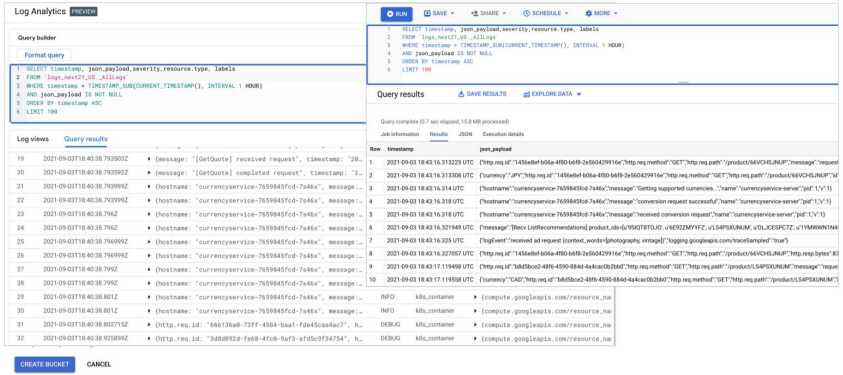


＋

さらに進化

Log Analytics　Preview

- Cloud Logging のログを、低コスト& 簡単に BigQueryに連携しアドホックなログ分析が可能に



本日のまとめ

本日お伝えしたこと

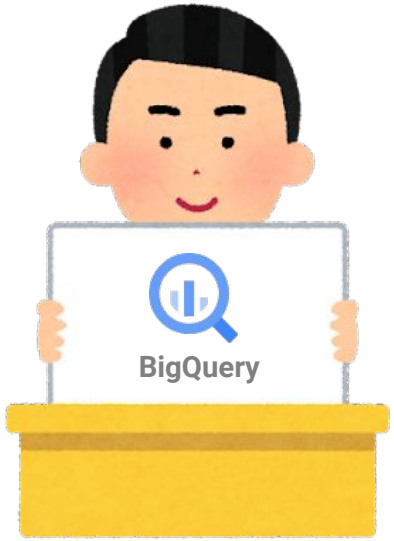
お客様が 現在・将来 にわたり実現したいこと

BigQuery の強み・特性

- On-demand・従量課金
- フェアスケジューラ
- ペタバイトクラスのデータ分析
- Compute と Storage の分離
- Dremel, Colossus, MindMeld, Jupiter
- ML, GIS, Analytics Hub

エンタープライズ DWH 対応

- 低レイテンシ・大量同時接続
- パフォーマンスチューニングの選択肢
- コスト効率の良いクエリ処理
- 高度なワークロード管理
- 柔軟なバックアップ
- 環境の複製・クローン
- トランザクション対応
- デプロイ・バージョン管理
- 監視・ログ分析



BigQuery のこれまでの強み + エンタープライズ DWH 対応

不確実な状況の中、

現在及び将来の多様なワークロードに対応するために、

貴重なデータをどこにおき、活用していくべきか？

DWH modernization を進める企業にとって、
シンプルかつ最適なソリューションを決定することは非常に難しく、
また後から変更できない重要な選択

進化する BigQuery と
Google Cloud エコシステムにより
お客様の Data-To-Value (データの価値化) を最大化します

Thank you.

Google Cloud