

JAPAN | 2024

aws SUMMIT



AWS-05

# AWS コスト管理の最前線

**石王 愛**

アマゾン ウェブ サービス ジャパン合同会社  
技術支援本部  
シニアテクニカルアカウントマネージャー



# 自己紹介



石王 愛 (いしおう あい)

シニアテクニカルアカウントマネージャー

- エンタープライズのお客様のAWS 活用支援をサポート
- コスト最適化推進
- 好きなAWSのサービス：  
AWS Cost and Usage Reports

# このセッションでお話しすること

## • 話すこと

- AWSのコスト増加の原因と担当者を調べ、コストを最適化する方法

## • 想定する参加者

- これからコスト管理・最適化をはじめたいと思っている CCoE、FinOps チーム、アプリケーションチーム、インフラチームの方

## • 話さないこと

- 各サービスにおける具体的なコスト最適化方法

# リリースしたら満足してませんか？

開発時

開発を始めるときは・・・  
コスト試算は慎重に  
やります！

運用時

運用に入ったら・・・  
あとは知りません



アプリケーション担当

# こんなことになっていませんか？

利用料金が上昇してます  
原因調査・コスト最適化  
お願いします

どこをコスト最適化  
できるか確認する体力が  
ありません

うちのプロジェクトの  
コストどうやって見るん  
ですか？



経理



アプリケーション担当

# Agenda

1. AWS コスト管理のベストプラクティス
2. ユースケースで考えるコスト最適化
3. まとめ

# AWS コスト管理の ベストプラクティス



# クラウドの料金の特性とは

必要な時に、必要なだけ、IT リソースを提供



光熱費のお支払いと同じ感覚で**利用した分だけお支払い**

# Well-Architected Framework コスト最適化の柱

5つの設計原則に従うことでコスト最適化を達成することが可能



## 5つの設計原則

1. クラウド財務管理（CFM）を実装する

2. 不要な「使用」を削減する

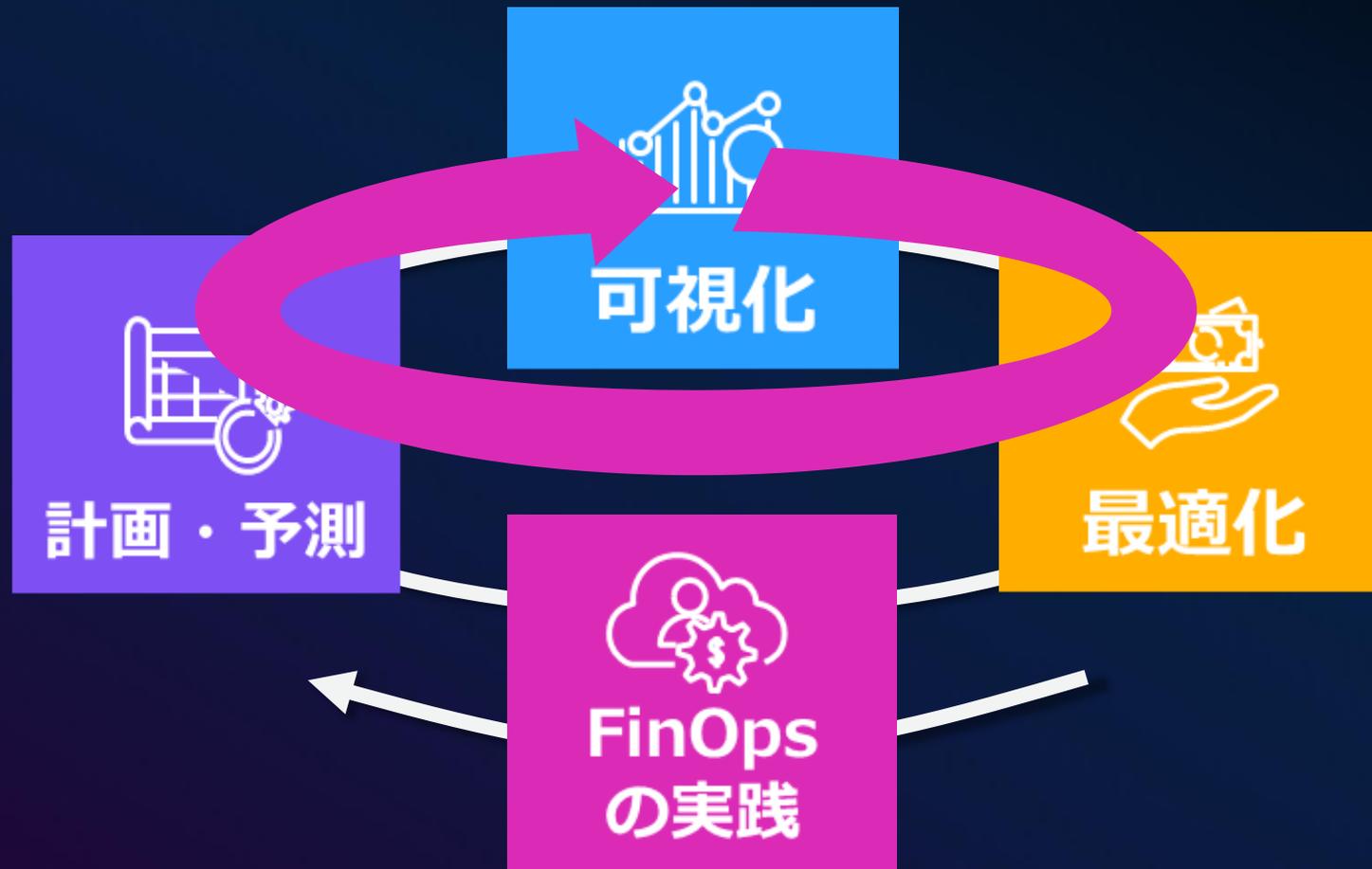
3. ビジネス成果の測定値とデリバリコストを使ってコスト効率性を評価する

4. 差別化につながらない高負荷の作業に費用をかけるのをやめる

5. 費用を分析し帰属関係を明らかにする

# Cloud Financial Management (CFM) の全体像

CFM には、「可視化」、「最適化」、「計画・予測」、「FinOps の実践」の4つの柱があります。



# ユースケースで考える コスト最適化

# 6月のとある日・・・

5月分の請求書を見たところ  
利用料金が上昇してました  
原因調査とコスト最適化  
お願いします！



経理  
Aさん

了解！



クラウド管理者  
Bさん

# コスト最適化何からはじめる？

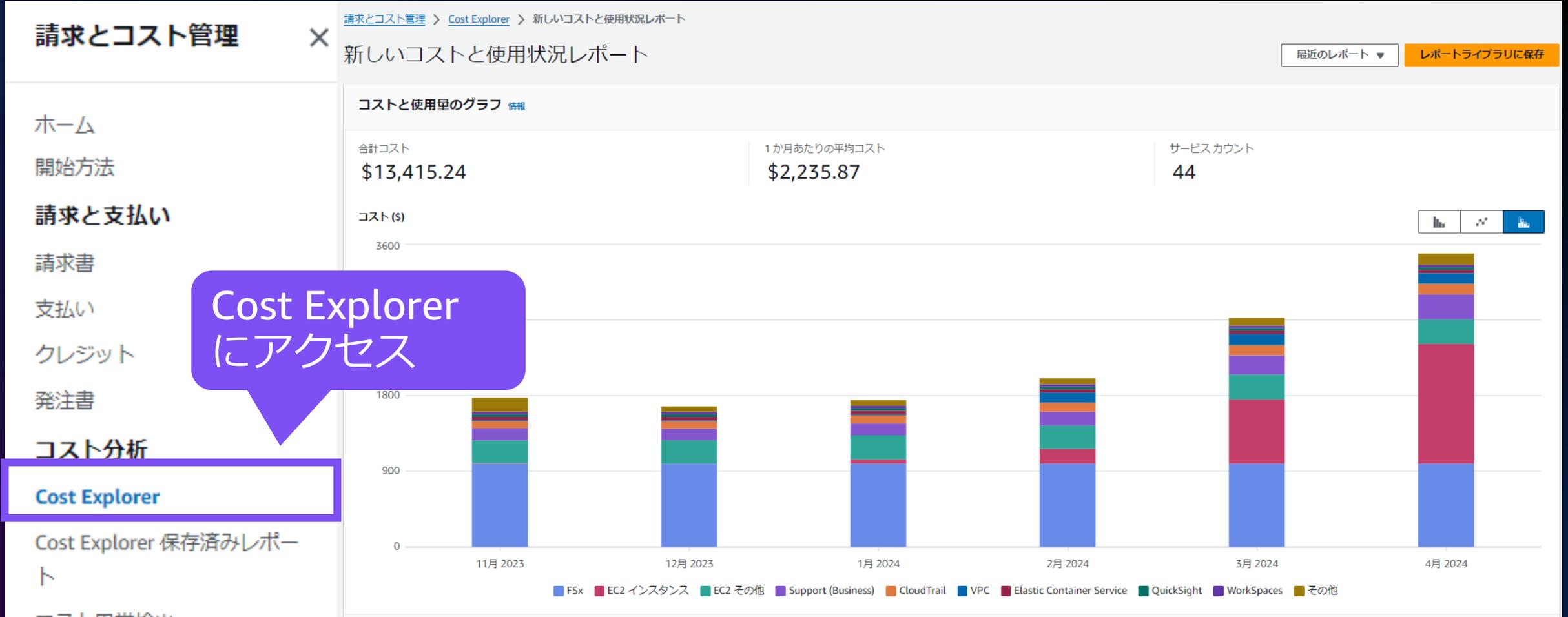
CFMの「可視化」から始めましょう！



# [ミッション1 コストの可視化]

## パパッと確認

# パパッとコストを確認したいときは・・・ AWS Cost Explorer ( CE )



# CE のレポートパラメータ

ミッション 1  
可視化

▼ 時刻

日付範囲 - 新機能

📅 2023-10-01 — 2024-03-31

表示中 過去 6 か月間

粒度

月別 ▼

▼ グループ化の条件

ディメンション - 新機能 クリア

サービス ▼

▶ フィルター - 新機能 情報

▼ 詳細オプション

次で集計したコスト: 情報

非ブレンドコスト ▼

期間・時間粒度

グルーピング

フィルタリング

集計コスト  
(コストの表示方法)

# 5月にどのサービスが増えたか確認しましょう

ミッション1  
可視化

▼ 時刻

日付範囲 - 新機能

📅 2024-04-01 — 2024-05-31

粒度

月別 ▼

▼ グループ化の条件

ディメンション - 新機能 クリア

サービス ▼

▶ フィルター - 新機能 情報

▼ 詳細オプション

次で集計したコスト: 情報

非ブレンドコスト ▼

期間・時間粒度

4月と5月を比較したいので  
日付：2024/4/1～2024/5/31  
粒度：月

グルーピング

どのサービスが伸びているか  
見たいので「サービス」を選択

フィルタリング

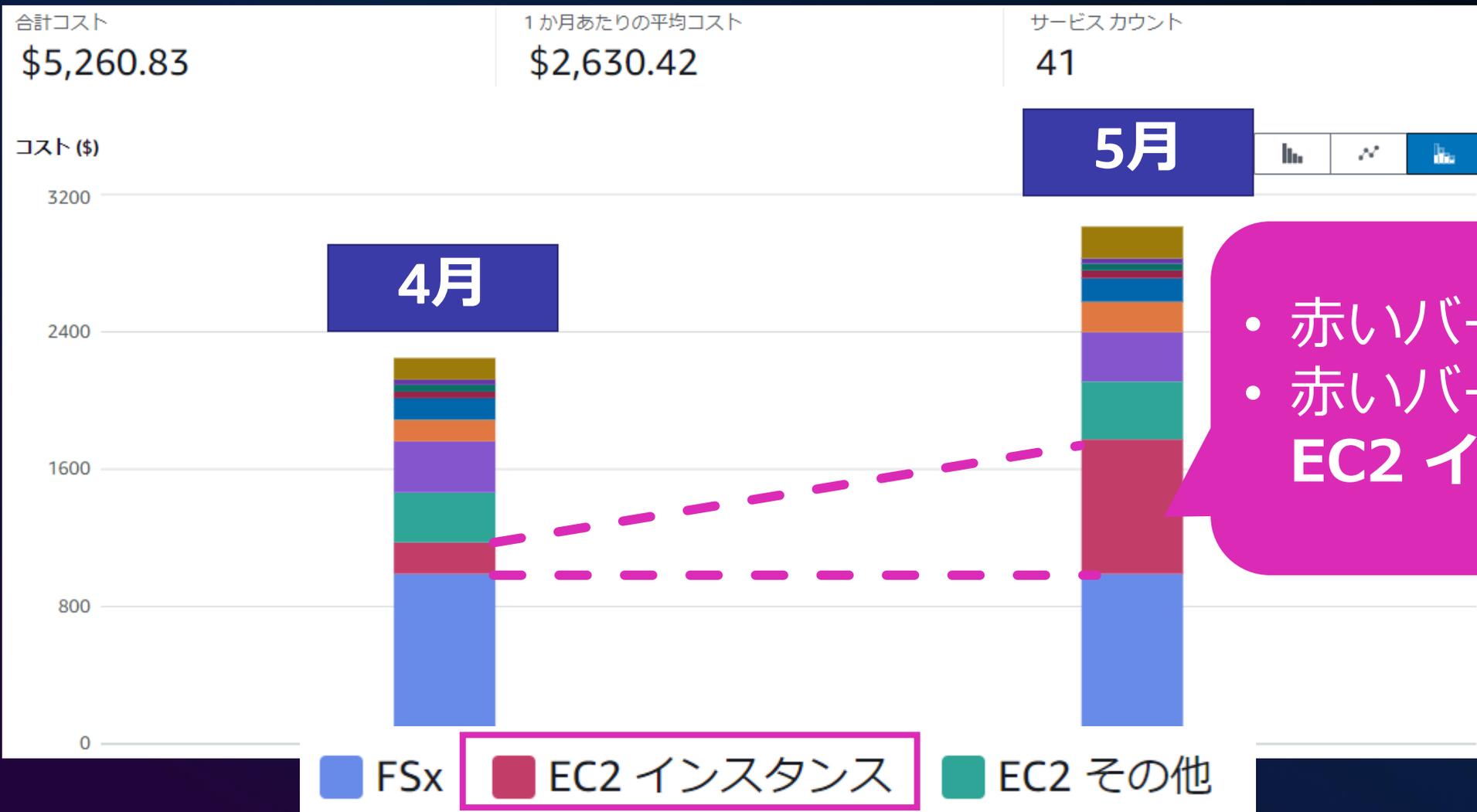
なし

集計コスト

請求書と同じ金額が見たいので  
「非ブレンドコスト」を選択

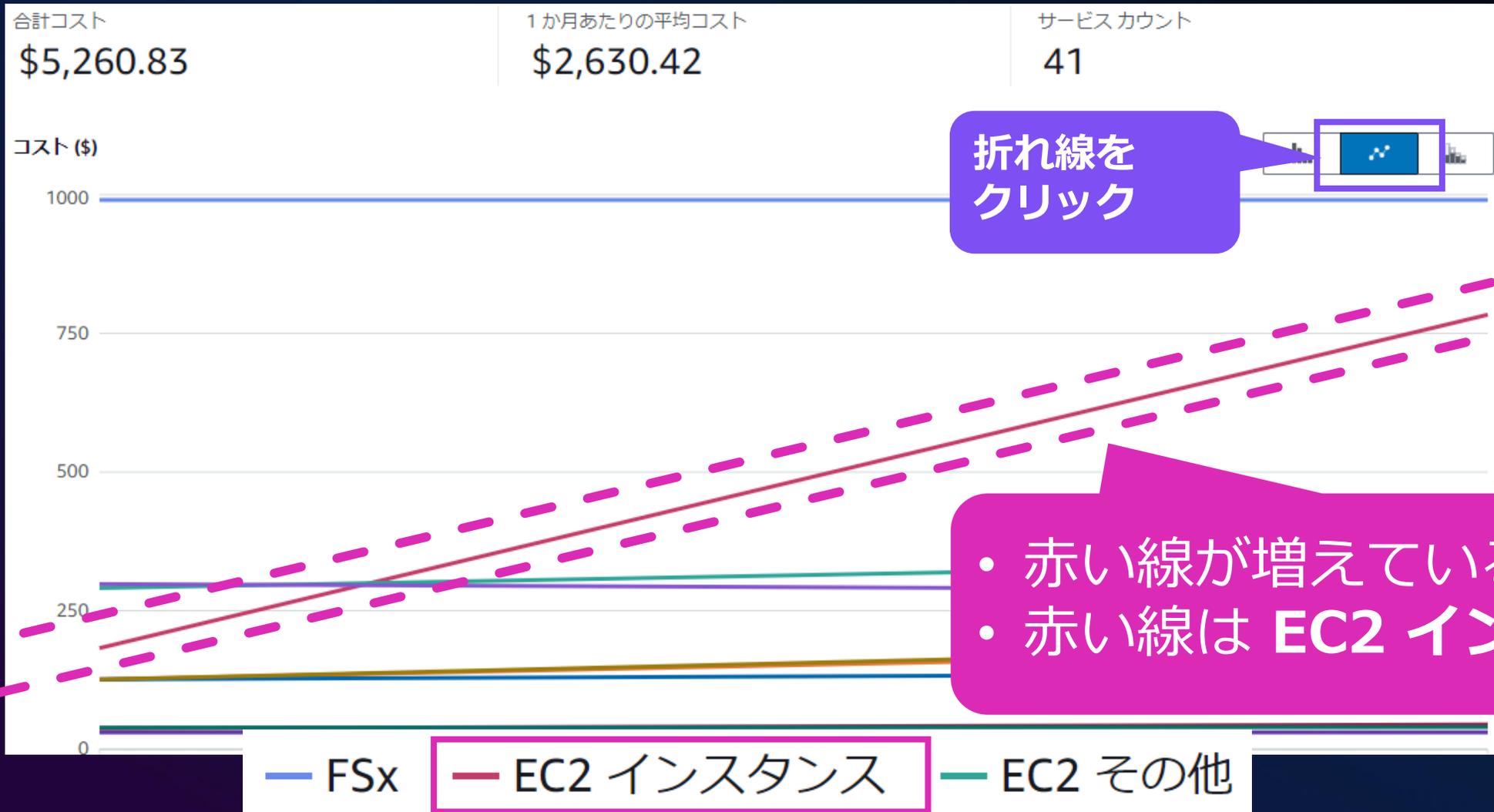
# 5月にどのサービスが増えたか確認しましょう

ミッション1  
可視化



- 赤いバーが増えている
- 赤いバーは EC2 インスタンス

# 折れ線なら変化がわかりやすい



# 生成 AI に聞くことも

# Amazon Q Developer can now help you understand your costs (preview)

4月と5月を比較して最もコストが高いAWS サービスとその分析をしてください

Amazon Q

Compare the costs from 2024/4/1 to 2024/4/30 with the costs from 2024/5/1 to 2024/5/31 and display a list of AWS services that have grown the most. Provide details and analysis.

To compare the AWS service costs from April 2024 to May 2024 and list the services with the highest cost increases, I will:

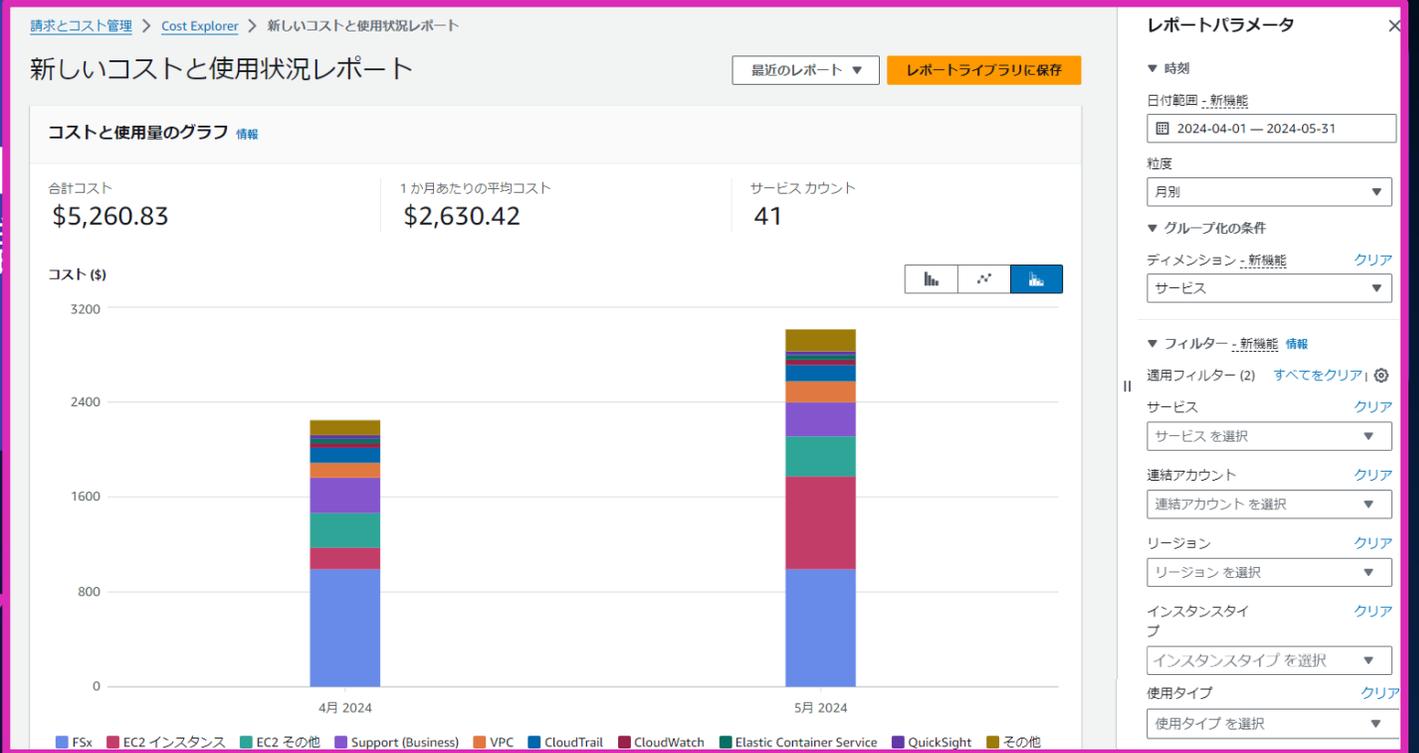
1. Identify the services that had an increase in cost from April to May.

CE の URL が  
自動で生成される

Sources

- [1] Visualize this data in AWS Cost Explorer
- [2] AWS Cost Explorer GetCostAndUsage API Reference
- [3] Modifying your chart - AWS Cost Management

4  
増



# 5月に EC2 のどのインスタンスタイプが増えたか確認しましょう

▼ 時刻

日付範囲 - 新機能

2024-05-01 — 2024-05-31

粒度

日別

▼ グループ化の条件

ディメンション - 新機能 [クリア](#)

インスタンスタイプ

▼ フィルター - 新機能 [情報](#)

適用フィルター (1) [すべてをクリア](#) [設定](#)

サービス [クリア](#)

サービスを含む (1)

EC2 インスタンス (Elastic Compute Cloud - Compute) [×](#)

▼ 詳細オプション

次で集計したコスト: [情報](#)

非ブレンドコスト

期間・時間粒度

5月の詳細を見たいので  
日付：2024/5/1～2024/5/31  
粒度：日

グルーピング

どのインスタンスタイプが伸びているか見たいので  
「インスタンスタイプ」を選択

フィルタリング

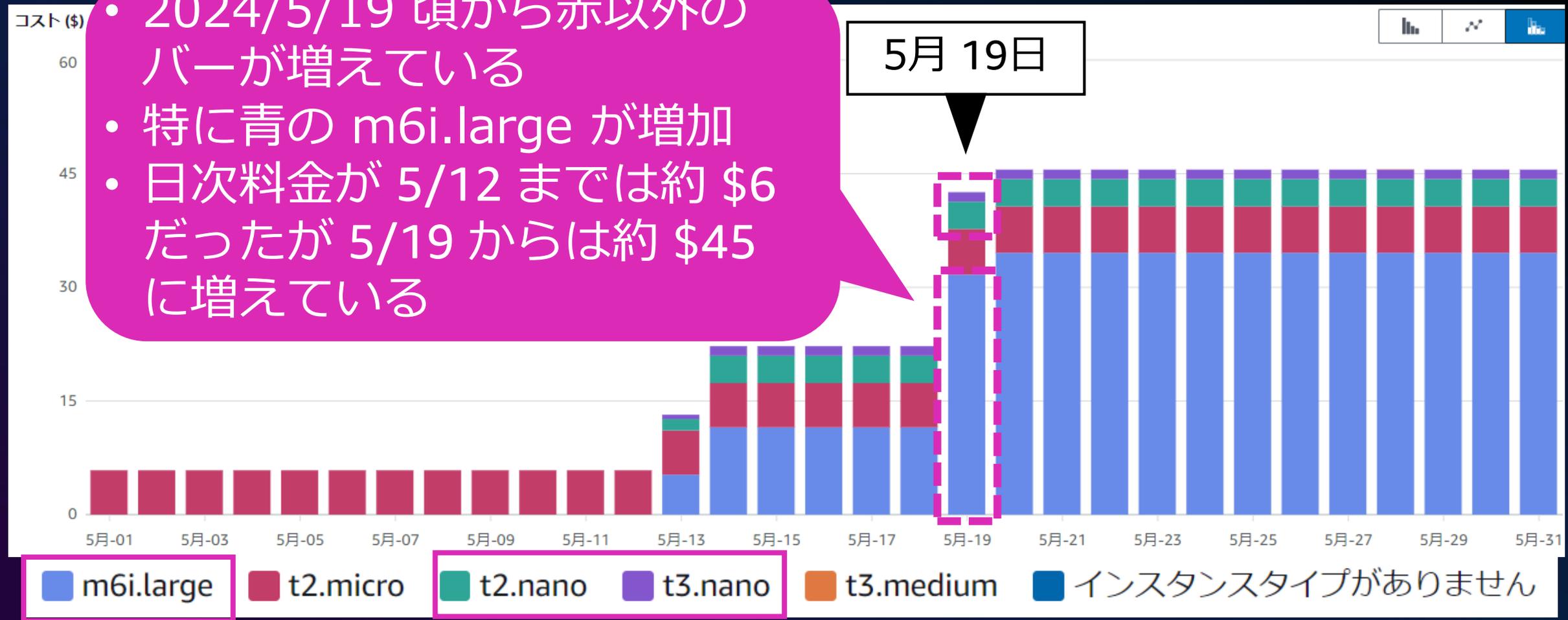
EC2 に絞ってみたいので  
「サービス：EC2 インスタンス」を選択

集計コスト

請求書と同じ金額が見たいので「非ブレンドコスト」を選択

# 5月に EC2 のどのインスタンスタイプが増えたか確認しましょう

- 2024/5/19 頃から赤以外のバーが増えている
- 特に青の m6i.large が増加
- 日次料金が 5/12 までは約 \$6 だったが 5/19 からは約 \$45 に増えている



# 調査の結果を報告したところ・・・

5月 19日から m6i.large、t2.nano、  
t3.nanoの料金が一日当たり  
約 \$40 増えていることが  
料金上昇の原因です！



クラウド管理者  
B さん



経理  
A さん

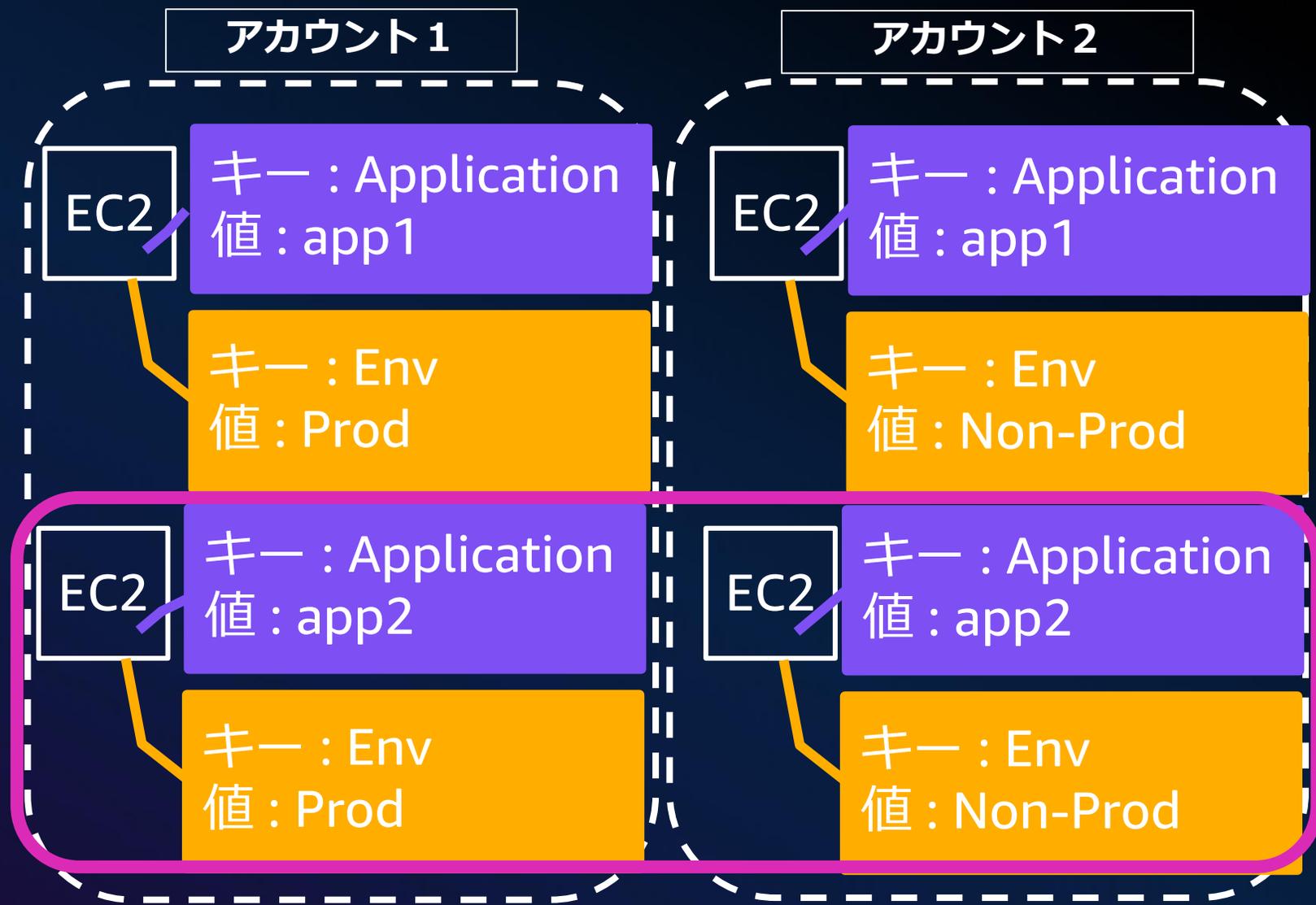
おお、素晴らしい！  
その EC2 を管理しているのは  
誰ですか？  
コスト最適化できないか確認  
してもらいましょう

# タグを使ってリソースの管理者を識別する

タグはAWSリソースにつける「名前」や「ラベル」のようなもの

タグを使ってリソースに様々な情報を付加できる

- 組み合わせは**キー**と**値**
- タグの単位例：  
プロジェクト・環境・アプリケーション・部門 等



# タグをコスト関連のサービスで見るとするには

請求とコスト管理 ×

ホーム  
開始方法  
請求と支払い  
請求書  
支払い  
クレジット  
発注書  
コスト分析

請求とコスト管理 > コスト配分タグ

## コスト配分タグ 情報

3個のコスト配分タグがアクティブ化されました

ユーザー定義のコスト配分タグ | AWS 生成コスト配分タグ

### ユーザー定義のコスト配分タグ (64) 情報

🔍 コスト配分タグを検索

<input type="checkbox"/>	タグキー	▼	ステータス	▼
<input type="checkbox"/>	ApplicationName		🟢 アクティブ	
<input type="checkbox"/>	Name		🟢 アクティブ	
<input type="checkbox"/>	testtasktag		🔴 非アクティブ	
<input type="checkbox"/>	EnableAWSServiceCatalogAppRegistry		🔴 非アクティブ	
<input type="checkbox"/>	user:Application		🔴 非アクティブ	

コスト配分タグに  
アクセス

Payer アカウントで  
タグをコスト配分タグと  
して有効化することが必要

# コスト配分タグのバックフィル機能を利用し、過去まで遡ってコスト関連のサービスでタグを見ましょう(最大 12か月間)

EC2	2023年												2024年			
	5	6	7	8	9	10	11	12	1	2	3	4	5	6		
EC2①	★															
EC2②																

バックフィル機能によりタグ確認可能

バックフィルはできない

2023年 5月 タグ付与  
タグキー: ApplicationName  
タグ値: app2

2024年 3月 タグ付与  
タグキー: ApplicationName  
タグ値: app2

ApplicationName  
コスト配分タグ  
有効化



# 5月に誰の EC2 が増えたか確認しましょう

ミッション 1  
可視化

▼ 時刻

日付範囲 - 新機能  
2024-05-01 — 2024-05-31

粒度  
月別 ▼

▼ グループ化の条件

ディメンション - 新機能 [クリア](#)

Tag ▼

Tag  
ApplicationName ▼

▼ フィルター - 新機能 [情報](#)

適用フィルター (1) [すべてをクリア](#) [設定](#)

サービス [クリア](#)

サービスを含む (1) ▼

EC2 インスタンス (Elastic Compute Cloud - Compute) ✕

▼ 詳細オプション

次で集計したコスト: [情報](#)

非ブレンドコスト ▼

期間・時間粒度

5月の詳細を見たいので  
日付：2024/5/1～2024/5/31  
粒度：日

グルーピング

どのチームの EC2 が伸びているか  
見たいので「タグ」  
タグキー：ApplicationName  
を選択

フィルタリング

EC2 に絞ってみたいので  
「EC2 インスタンス」を選択

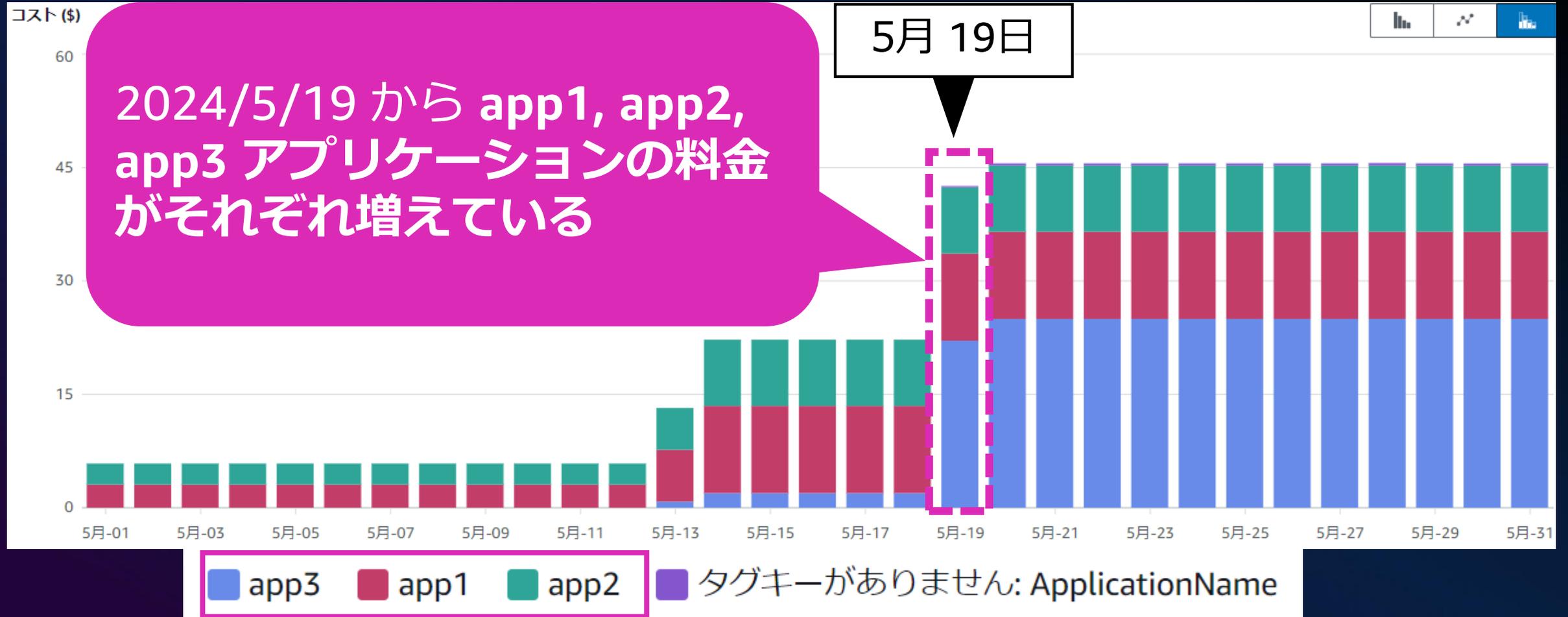
集計コスト

請求書と同じ金額が見たいので  
「非ブレンドコスト」を選択



# 5月に誰の EC2 が増えたか確認しましょう

ミッション1  
可視化



# でもタグの付与って漏れちゃうこと ありませんか？

## 予防的

適切なタグが付与されるようガードレールを設定



- Service control policies
- IAM Policies

## 能動的

開発時にタグを付与する



- AWS CloudFormation
- AWS Service Catalog

## 受動的

タグが付いていないリソースの検知/検索・タグを付与する



- AWS Resource Explorer
- AWS Tag Editor
- AWS Config
- AWS Security Hub

# タグが付いていないリソースを発見 できる様々なサービス

種別	サービス	クロスアカウント検索	料金
検索	AWS Resource Explorer	○	無料
	AWS Tag Editor	×	無料
検知・通知	AWS Config Rule (required-tags)	○	有料
	AWS Security Hub	○	有料

# AWS Resource Explorerで 指定したタグが付いていないリソースを検索

AWS Resource Explorer > リソースの検索

## リソースの検索

Search for resources

表示: all-resources [ビューを変更]

クエリ: Supports Capability = tags ApplicationName タグが無いものに絞る

1000 リソースを CSV にエクスポートする

Organizations 内を横断で確認できる

識別子	リソースタイプ	リージョン	AWS アカウント	タグ: Name
f4b...29-86	ssm:automation-execution	カナダ (中部) ca-central-1	このアカウント	(タグ付けなし)
72...2fc-bf	ssm:automation-execution	米国東部 (バージニア北部) us-east-1	このアカウント	(タグ付けなし)

# 各アカウントの AWS Tag Editor で タグが付いていないリソースを検索

**Tag Editor**

**Find resources to tag**  
You can search for resources that you want to tag across regions. Then, you can add, remove, or edit tags for resources in your search results. [Learn more](#)

**Regions**  
Select regions ▼  
All regions X

**Resource types**  
Select resource types ▼  
All supported resource types X

**Tags - Optional**  
Tag key:   
Optional tag value:  Add  
Type the tag key and optional values shared by the resources you want to search for, and then choose Add or press Enter.  
ApplicationName: (not tagged) X

リージョン

リソースタイプ

検索したいタグキー

※複数指定可  
※タグキー & タグの  
値の組合せでも  
検索可能

# 各アカウントの AWS Tag Editor で タグを付与！

リソースの検索結果 ( リソースのロード中)

999 resources を CSV にエクスポート ▼

選択したリソースのタグを管理する

タグを編集するリソースを最大 500 まで選択します。

⚠ タグエディタを使用して、ネストされた CloudFormation スタックにタグを付けたリタグ付け解除したりしないことをお勧めします。代わりに、ネストされたスタックにタグ付けまたはタグ付け解除します。  
ネストされた CloudFormation スタックにタグ付けまたはタグ解除を行う場合は、親スタックにのみ変更を適用することを強くお勧めします。

一括でタグ付けが可能

Q 最初の 999 のリソースをフィルタリングする

<input checked="" type="checkbox"/>	識別子	サービス	タイプ	リージョン	タグ	
<input checked="" type="checkbox"/>	Exa...		Fleet	us-east-1	(タグ付けされていませ ん).....	-
<input checked="" type="checkbox"/>	Exa...		Stack	us-east-1	(タグ付けされていませ ん).....	-
<input checked="" type="checkbox"/>	C9-...		Stack	us-east-1	(タグ付けされていませ ん).....	-
<input checked="" type="checkbox"/>	ec2...	CloudFormation	Stack	us-east-1	(タグ付けされていませ ん).....	1
<input checked="" type="checkbox"/>	vpc...	CloudFormation	Stack	us-east-1	(タグ付けされていませ ん).....	1

リソースを  
選択

# 補足 : AWS Config 高度なクエリ 自然言語クエリプロセッサ ( preview )

AWS Config 高度なクエリでもタグがついていないリソースを Organizations 内を横断で検索できます。

自然言語クエリプロセッサを利用すれば簡単な英語で尋ねるだけで、ほしい情報を検索する SQL を生成できます。

[AWS Config](#) > [高度なクエリ](#) > [クエリエディタ](#)

クエリエディタ

```
List EC2 which do not have tag key =  
ApplicationName
```

```
SELECT  
  resourceId,  
  resourceType  
WHERE  
  resourceType = 'AWS::EC2::Instance'  
  AND NOT tags.key = 'ApplicationName'
```

# 補足: タグの付与にはこんな裏技も AWS Service Catalog AppRegistry

あとから  
CloudFormation( CFN )  
スタックのリソースに  
まとめてタグ付けするとき  
に便利

- 自動で **AWSApplication** タグがリソースに付与される
- 自動でコスト配分タグ **AWSApplication** が有効化される

The screenshot shows the AWS Service Catalog 'Create Application' page. The left sidebar contains navigation options like 'Home', 'Provisioning', 'AppRegistry', and 'Marketplace'. The main content area is titled 'アプリケーションを作成' (Create Application). It includes a description, a form for 'アプリケーションの名前と説明' (Application Name and Description), and a section for 'リソース' (Resources). Under 'リソース', there are three options: 'プロビジョニング済み製品を追加' (Add provisioned products), 'CloudFormation (CFN) スタックに基づいて選択' (Select based on CloudFormation (CFN) stacks), and 'アプリケーションのタグ付けを有効化' (Enable application tagging). A blue callout box with white text points to the 'CloudFormation (CFN) スタックに基づいて選択' option, stating 'CloudFormation ( CFN ) スタックを指定することも可能' (It is also possible to specify CloudFormation (CFN) stacks). Below this, there is a dropdown menu for selecting stacks and a 'すべて閲覧' (View all) button.

# ここまでのまとめ

- **AWS Cost Explorer** でパパッとコストを確認しよう！
  - まずは AWS Cost Explorer を使って、いつ、どのインスタンスタイプが増えたのかを絞り込みましょう。
  - 次にタグを使って、どのチームのリソースが増えたのかを絞り込む。コスト配分タグの有効化も忘れずに。
  - タグが無いものは AWS Resource Explorer で検索し AWS Tag editor を使って付与しましょう。

# よし、担当者がわかった！

増えた原因は  
app1, app2, app3 アプリケーション  
のリソースだとわかりました！



クラウド管理者  
Bさん



経理  
Aさん

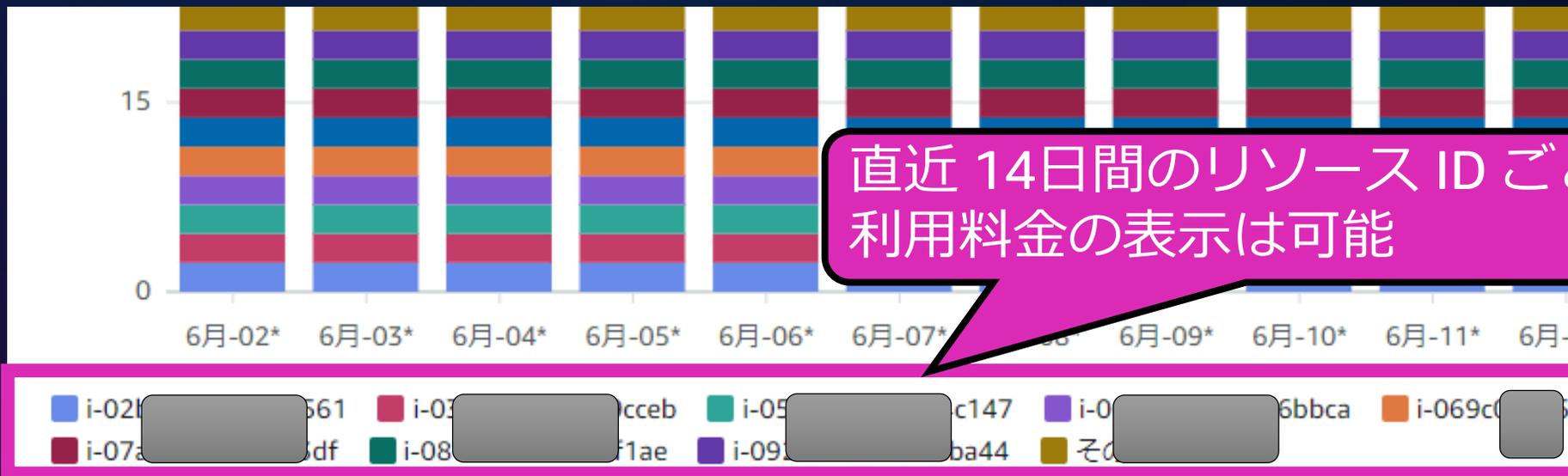
ありがとうございます！  
では各アプリケーションの  
EC2 一覧を抽出して担当者に  
内容を確認してもらいましょう！

# [ミッション1 コストの可視化]

細かく確認

# アプリケーションチームごとの EC2 一覧を 作成する

AWS Cost Explorer では直近 14日間のリソース ID を確認できる(\*)。  
しかし、グルーピング条件は一つしか指定できないため、  
アプリケーションチームごとのリソース ID 一覧は作成できない。



\* 請求とコスト管理 > コスト管理の設定 > 履歴データおよび詳細データの設定  
で「日単位の詳細度のリソースレベルのデータ」有効化が必要（無料）

# AWS Cost and Usage Reports ( CUR ) ならアプリケーションチームごとの EC2 一覧を作成 できる

line_item_usage_id	line_item_usage_start_time	line_item_usage_end_time	line_item_usage_type	line_item_product_code	product_id	line_item_resource_id	line_item_usage_amount	line_item_usage_unit	resource_tags
123456789123	2024/5/1 0:00	2024/5/1 1:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 1:00	2024/5/1 2:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 2:00	2024/5/1 3:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 3:00	2024/5/1 4:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 4:00	2024/5/1 5:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 5:00	2024/5/1 6:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}
123456789123	2024/5/1 6:00	2024/5/1 7:00	Usage:t2.micro	AmazonEC2	t2.micro	i-03958	0.0116	Hour	{user_application_name=app1, user_aws_application=arn:aws:resource-groups:us-east-1:123456789123:group/dep1/00x1hqj2xxkbye9z1rcj1jsco8}

1 時間単位の  
コストデータが見れる

リソース ID も  
わかる

各リソース ID のタグ  
情報もわかる

# AWS Cost and Usage Reports ( CUR )

AWS の利用状況と  
利用料金情報を最も細かく  
包括的に提供する  
コストデータ

- 格納先 S3 を指定して作成
- CUR を作成するときに時間・日・月単位などの時間粒度やリソース ID を含めるかといったことが設定時に指定可能

作成方法は[こちら](#)



# CUR は Amazon Athena で SQL を使って 分析するのがおすすめ！



CUR を作成すると  
Amazon S3 に  
格納される



Amazon Athena で  
SQL を  
使った分析も可能

CUR 作成から Amazon Athena で分析するための設定は[こちら](#)

# CUR を使えば . . . アプリケーション毎の EC2 一覧の抽出も簡単！

BillingPeriod	tag_name	tag_value	AccountName	InstanceType	ResourceId	Hours	Cost
May-24	user_application_name	app1	Prod	t2.micro	i-02222222222222221	744	8.6304
May-24	user_application_name	app1	Prod	t2.micro	i-03333333333333339	743	8.6188
May-24	user_application_name	app1	Non-Prod	t2.nano	i-0666666666666666a	442	2.5636
May-24	user_application_name	app1	Non-Prod	t2.nano	i-099999999999999953	442	2.5636
May-24	user_application_name	app2	Prod	t2.nano	i-0111111111111111fc	442	2.5636
May-24	user_application_name	app2	Prod	t2.nano	i-05555555555555553	442	2.5636
May-24	user_application_name	app2	Non-Prod	t2.nano	i-08888888888888881	442	2.5636
May-24	user_application_name	app3	Prod	t2.nano	i-04444444444444447	442	2.5636

# CUR を使えば . . . EC2 ではなくコンテナだったとしても大丈夫！

追加の設定で Amazon ECS, Amazon EKS の各 Task や Pod 単位の利用状況に基づきコストを配分したデータを取得することができます。

- コスト管理の設定で「コスト配分データを分割」を有効化
- CUR を作成するときに「コスト配分データの分割(SCAD)」を有効化

lineItem/ ResourceID	lineItem/ lineItemType	lineItem/ UsageType	lineItem/ UnblendedCost	lineItem/ NetUnblendedCost	savingsPlan/ SavingsPlanEffectiveCost	savingsPlan/ NetSavingsPlanEffectiveCost	splitLineItem/ ParentResourceid	splitLineItem/ SplitUsage	splitLineItem/ SplitCost	splitLineItem/ NetSplitCost	splitLineItem/ UnusedCost	splitLineItem/ NetUnusedCost
i-12345	SavingsPlan CoveredUsage	BoxUsage: m5.xlarge	1.5	1.4	1	0.8						
EC2-Pod1	Usage	EKS-EC2-vCPU- Hours						1	0.14	0.11	0	0
EC2-Pod1	Usage	EKS-EC2-GB- Hours						4	0.08	0.06	0.01	0.01
EC2-Pod2	Usage	EKS-EC2-vCPU- Hours						1.9	0.27	0.21	0	0
EC2-Pod2	Usage	EKS-EC2-GB- Hours						6	0.12	0.09	0.02	0.01
EC2-Pod3	Usage	EKS-EC2-vCPU- Hours							0.14	0.11	0	0
EC2-Pod3	Usage	EKS-EC2-GB- Hours					i-12345	2	0.04	0.03	0.01	0
EC2-Pod4	Usage	EKS-EC2-vCPU- Hours					i-12345	1	0.14	0.11	0	0
EC2-Pod4	Usage	EKS-EC2-GB- Hours					i-12345	2	0.04	0.03	0.01	0

CPU,メモリ使用率に  
基づいて算出した Pod  
ごとのコスト

# 補足：改善された CUR 2.0 を使いましょう！

昨年秋に CUR2.0 が発表され、今までの CUR は「レガシー CUR」に。  
今後は**改善された CUR2.0** をご利用頂くことをお勧めします！

  
レガシー CUR  
のご利用者

Reserved Instance とか  
Savings Plans 購入したら  
CUR の列が増えた🌀

コスト配分タグを  
追加したら  
CUR の列が増えた🌀

「支払アカウント ID」  
「使用アカウント ID」しか  
ないから名称は自分で紐づけ🌀



  
CUR2.0  
のご利用者

**CUR の列が変わらない**♡  
**含める列を自分で選べる** ♡  
から不要な列は除外できる♡

コスト配分タグを追加しても  
**resource\_tags 列の値に**  
**まとめて表示**される♡

**「支払アカウント名」**  
**「使用アカウント名」**列がある♡

# お勧め：簡単にコストダッシュボードを作成 AWS Cost and Usage Dashboard ( CUD )

請求とコスト管理

- ホーム
- 開始方法
- 請求と支払い
- 請求書
- 支払い

Cost Explorer

- Cost Explorer 保存済みレポート
- コスト異常検出
- 無料利用枠
- データエクスポート**
- コスト組織
- Cost Categories
- コスト配分タグ
- 請求コンダクター

エクスポート名	ステータス	エクスポートタイプ	データテーブル	作成日
<a href="#">mycostreport</a>	🟢ヘルシー	コストと使用状況ダッシュボード	コストと使用状況ダッシュボード	2024年1月10日 13:31 GMT+09:00
<a href="#">newcur</a>	🟢ヘルシー	標準データ		2024年1月10日 13:31 GMT+09:00
<a href="#">cidpayer</a>	🟢ヘルシー	レガシー		2024年1月10日 13:31 GMT+09:00
<a href="#">curhourly2</a>	🟢ヘルシー	レガシー		2024年1月10日 13:31 GMT+09:00

コストと使用状況ダッシュボードに  
アクセスして**数クリック**で作成可能

データエクスポートに  
アクセス

Powered by **QuickSight**



**コストと使用状況ダッシュボード**

QuickSight でコストと使用状況のダッシュボードをデプロイして設定すると、インサイトを見つけてビジネスチームと共有できます。

**顧客の二酸化炭素排出量ツール**

顧客の二酸化炭素排出量ツールにアクセスし、AWS 製品およびサービスに関連する二酸化炭素排出量の見積もりをご覧ください。

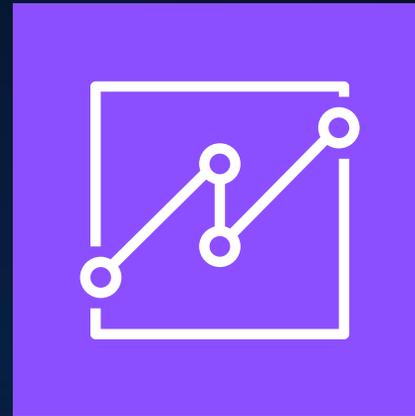
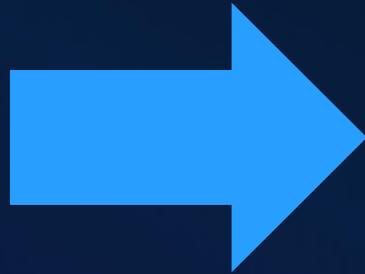
**AWS 使用状況レポート**

単一のサービスをカバーする動的に生成された AWS 使用レポートを設定し、ダウンロードします。

# 数クリックだけで自動で CUR の作成 から Amazon QuickSight の設定まで行います



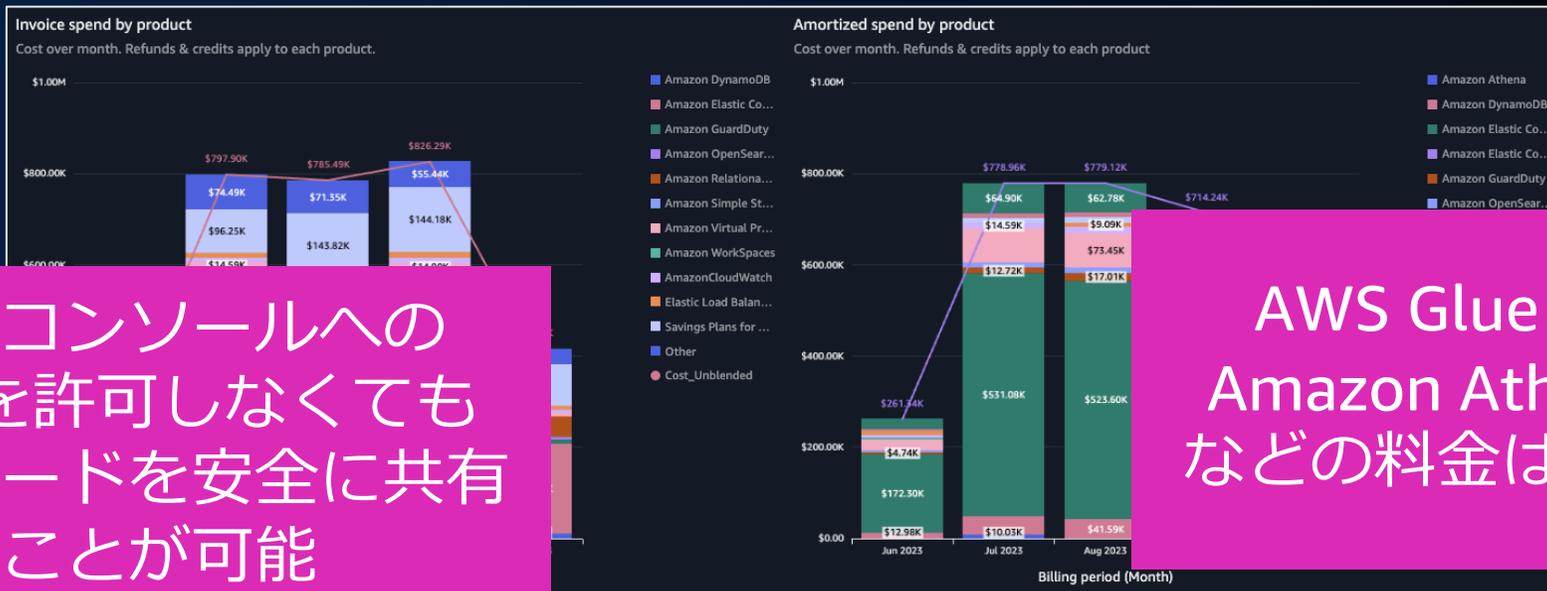
CUR を作成して  
Amazon S3 に  
格納



Amazon QuickSight  
を使って可視化

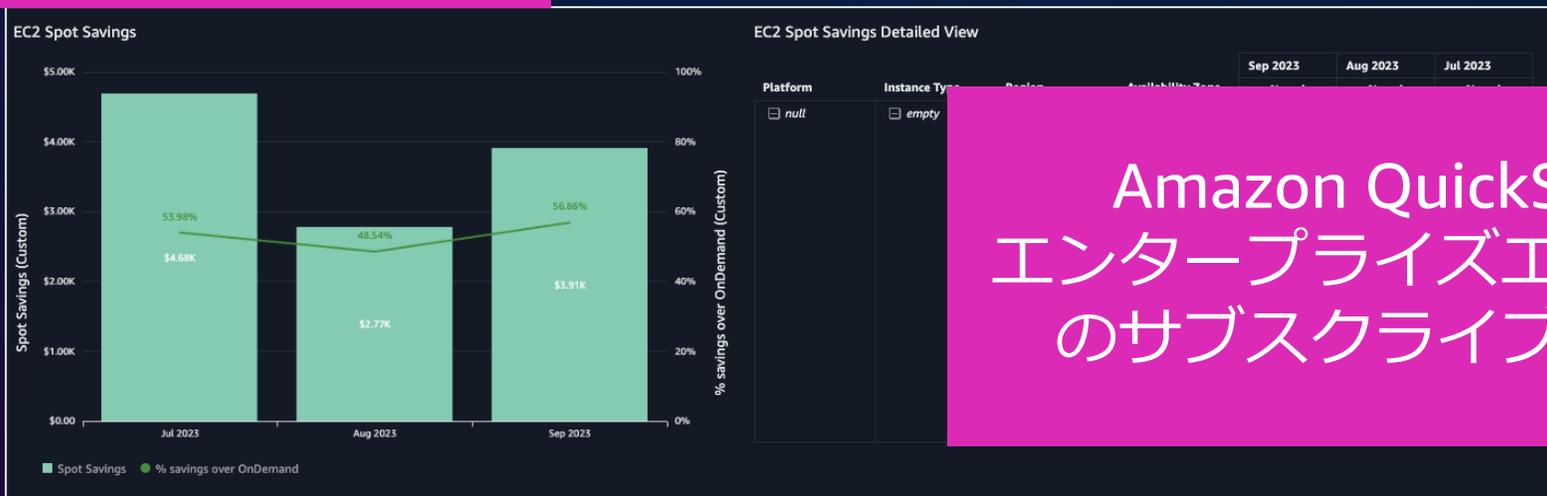
# AWS Cost and Usage Dashboard ( CUD )

ミッション1  
可視化



閲覧者にコンソールへのアクセスを許可しなくてもダッシュボードを安全に共有することが可能

AWS Glue や Amazon Athena などの料金は不要



Amazon QuickSight のエンタープライズエディションのサブスクライブ料金のみ



# ここまでのまとめ

- **AWS Cost and Usage Reports ( CUR )** でアプリケーションチーム毎の EC2 一覧を作成しよう！
- CUR を使えばリソース単位や時間単位のコストを細かく分析できる。
- 簡単に **AWS Cost and Usage Dashboard ( CUD )** でコストダッシュボードを作成しよう！

# アプリケーションチームに連絡！

EC2 一覧を送付します  
コスト最適化の余地がないか  
確認してください！



クラウド管理者  
Bさん



app1, app2, app3  
アプリケーション  
担当者

了解！

# コスト最適化何からはじめる？

CFMの「最適化」に取り組みましょう！



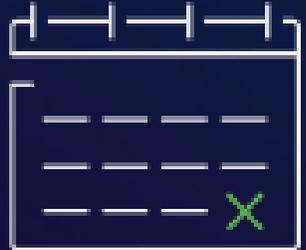
# [ミッション2 コスト最適化]

## クイックウィン最適化

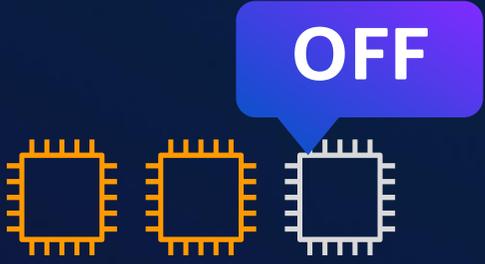
# コスト最適化は何からはじめたらいいの？

## クイックウィン最適化

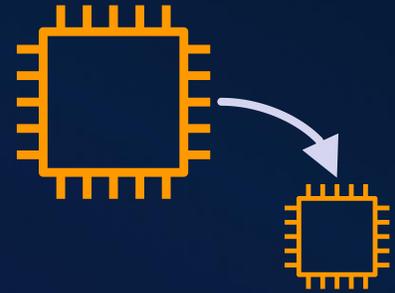
① 需要に応じたスケジューリング



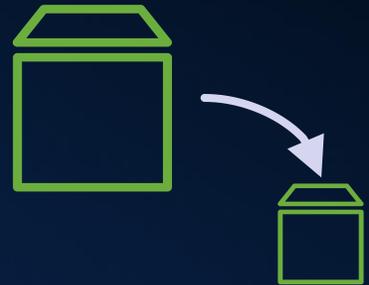
② 未使用リソース停止



③ 適切なインスタンスタイプの見直し



④ 適切なストレージの見直し



⑤ Savings Plans の購入



インスタンス

EBS



# ① 需要に応じたスケジューリング

まずは開発環境などの  
常時稼働不要なサーバを  
夜間・土日停止しましょう

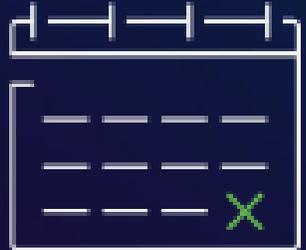
時間	月	火	水	木	金	土	日
8:00-20:00	○	○	○	○	○	×	×
20:00-8:00	×	×	×	×	×	×	×

サービス	特徴
Amazon EventBridge スケジューラ と AWS Lambda	<ul style="list-style-type: none"> <li>一番シンプルな実装方法</li> </ul>
AWS Systems Manager Resource Scheduler	<ul style="list-style-type: none"> <li>使いやすい UI</li> <li>マルチアカウント制御可能</li> </ul>
AWS Instance Scheduler	<ul style="list-style-type: none"> <li>AWS ソリューション</li> <li>マルチアカウントの制御が可能</li> <li>柔軟なスケジュール設定</li> </ul>

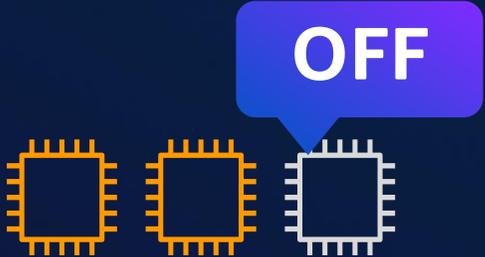
# コスト最適化は何からはじめたらいいの？

## クイックウィン最適化

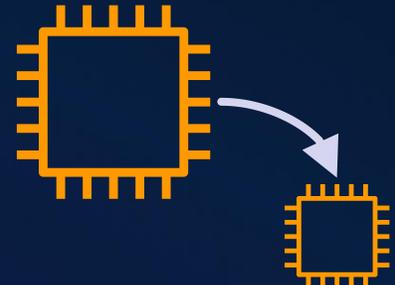
① 需要に応じたスケジューリング



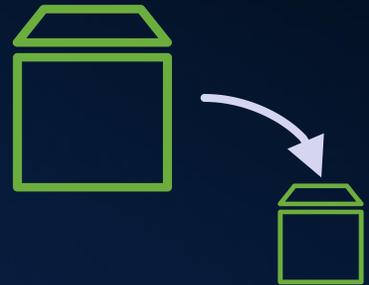
② 未使用リソース停止



③ 適切なインスタンスタイプの見直し



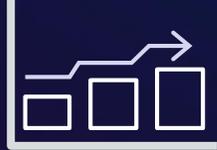
④ 適切なストレージの見直し



⑤ Savings Plans の購入

全てのリコメンデーションを一度に確認することが可能

インスタンス



EBS



# 一度にリコメンデーションを確認できる！ AWS Cost Optimization Hub

The screenshot shows the AWS Cost Optimization Hub interface. On the left is a navigation menu with categories like '請求とコスト管理', 'コスト分析', and 'Savings とコミットメント'. The 'コスト最適化ハブ' link is highlighted. The main content area shows 'コスト最適化ハブ 情報' and a '推奨アクション' dropdown menu. A donut chart displays '\$1,193 1か月あたり' with callouts for 'アップグレード', '適切なサイズ設定', and 'Graviton に移行'. A '機会を見る' button is at the bottom right.

請求とコスト管理 ×

ホーム  
開始方法  
請求と支払い  
請求書  
支払い  
クレジット  
発注書  
コスト分析  
Cost Explorer  
Cost Explorer 保存済みレポート

請求とコスト管理 > コスト最適化ハブ

コスト最適化ハブ 情報  
コスト最適化の機会を絞り込み、推奨削減額を集計します。リソースごとのコスト最適化に関する推奨アクションが表示されます。

推奨削減額を次の条件で集計:  
推奨アクション ▼

フィルター

アップグレード  
適切なサイズ設定  
Graviton に移行

\$1,193  
1か月あたり

アイドル状態または未使用のリソースを停止

削減額推定モードを割引前に設定した場合、Cost Optimization Hub はリソースベースの推奨アクションとリザーブドインスタンス/Savings Plans の購入の間で、推奨削減額の最大値を計算します。毎月の推奨削減額は、過去 30 日間の AWS 支出よりも高くなる場合があります。

機会を見る

コスト最適化の  
リコメンデーションが表示される

コスト最適化ハブ  
にアクセス

リソース単位の  
リコメンデーション  
を見てみましょう！

# リコメンデーションの詳細

まずリコメンデーションの対象を、見たい EC2 に絞ります

The screenshot shows the AWS Cost Explorer interface for '削減額機会' (Cost Savings Opportunities). It is divided into two panels. The left panel shows the '推定削減額があるリソース' (Resources with estimated savings) section, where the 'リソースタイプ' (Resource Type) filter is set to 'EC2 インスタンス' (EC2 Instance). The right panel shows the 'タグ' (Tags) filter section, where the 'ApplicationName' tag is set to 'app2'. A table of resources is partially visible on the right side of the right panel.

**EC2 に絞る**

**app2 アプリケーションのリソースに絞る**

現在のリソース
i-0e527ce724
i-0e09f17f073
i-0a4d3f9221
i-087babbcec
i-0a728604fe
i-07affcc1743

# リコメンデーションの詳細

ミッション2  
最適化

各 EC2 にどのアクションをとるかこの画面だけで決めることができる！

一つのインスタンスについて複数のコスト最適化リコメンデーションと推定削減額が表示される

推定削減額があるリソース

検索: テキスト、プロパティまたは値で分布をフィルタリング

リソースタイプ = EC2 インスタンス × タグ = ApplicationName フィルター

毎月の推定削減額	リソースタイプ	リソース ID	推奨アクション	現在のリソースの概要	推奨リソースの概要	推定削減率	推定月額コスト	実装作業
\$13.87	EC2 インスタンス	[REDACTED]	Graviton に移行	m6i.large	m6g.large	20%	\$70.08	非常に高い
\$0.18	EC2 インスタンス	[REDACTED]	アップグレード	m6i.large	m7i-flex.large	1%	\$70.08	中
\$70.08	EC2 インスタンス	[REDACTED]	Stop	i-08[REDACTED]	-	100%	\$70.08	低

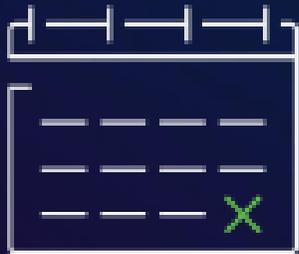
  

リソースの再起動は必要ですか	ロールバックは可能ですか	アカウント ID	リージョン	タグ	日数単位のルックバック期間
はい	はい	19[REDACTED]	米国東部 (バージニア北部)	ApplicationName:app2	32
はい	はい	19[REDACTED]	米国東部 (バージニア北部)	ApplicationName:app2	32
はい	はい	19[REDACTED]	米国東部 (バージニア北部)	ApplicationName:app2	32

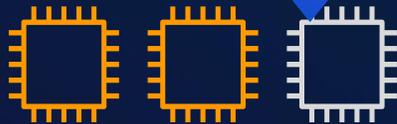
# コスト最適化は何かからはじめたらいいの？

## クイックウィン最適化

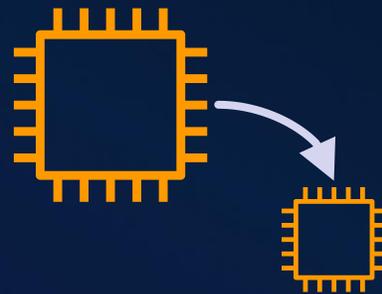
① 需要に応じた  
スケジューリング



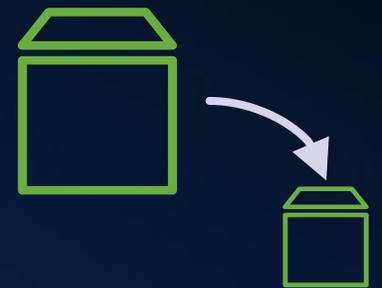
② 未使用  
リソース停止



③ 適切な  
インスタンスタイプ  
の見直し



④ 適切な  
ストレージの  
見直し



⑤ Savings Plansの購入



適切な稼働、キャパシティへの変更が完了したら  
コミットメントの購入検討を

インスタンス

EBS



# ①~③ のアクションが完了したら ⑤ Savings Plansの購入を検討しましょう

推定削減月額

## Compute Savings Plans :

### Recommended actions:

- Purchase Savings Plans | \$40.033/hour with three years term (AllUpfr
- Purchase Savings Plans | \$40.173/hour with three years term (Partia
- Purchase Savings Plans | \$40.422/hour with three years term (NoUpfront) in AWS accou
- Purchase Savings Plans | \$37.003/hour with one year term (AllUpfr
- Purchase Savings Plans | \$36.841/hour with one year term (Partiall
- Purchase Savings Plans | \$36.879/hour with one year term (NoUpfr

**\$8,874.55/month**  
Estimated savings

[View recommendations](#)

Consider purchasing recommended Compute Sav  
You can purchase Compute Savings Plans or view  
the Billing and Cost Management console.

オプション違いの  
Compute Savings Plans  
リコメンデーション

# 補足 : Amazon CloudWatch natural language query generation

Amazon CloudWatch Logs /  
Amazon CloudWatch メトリクスに  
ついて、簡単な英語で尋ねる  
だけでほしい情報を検索する  
SQL を生成できます。

この例では  
Amazon CloudWatch Logs Insights  
で請求額が最も高い Lambda  
呼び出しの上位 100 件を検索する  
SQL を生成しています。

```
Return the top 100 highest billed  
lambda invocations
```

```
filter @type = "REPORT"  
| fields @requestId, @billedDuration  
| sort by @billedDuration desc  
| limit 100
```

新しいクエリを生成

クエリを更新

# ここまでのまとめ

- コスト最適化は**クイックウィン最適化**から
  - まずは開発インスタンスなどの夜間休日停止から始めましょう
  - 次に Cost Optimization Hub で低稼働率のインスタンスの停止やインスタンスタイプの変更のそれぞれの最適化効果を比較し、最も適切な施策を実行しましょう
  - それが終わったら Savings Plans などのコミットメントの購入による最適化効果を Cost Optimization Hub で確認し、購入を検討しましょう

# その後 . . .

PoC インスタンスの削除で約 \$100  
検証環境の夜間休日停止で約 \$200  
削減できました！  
本番稼働したら

Compute Savings Plans も購入予定です！



app1, app2, app3  
アプリケーション  
担当者

ありがとう  
ございます！



経理  
A さん



クラウド管理者  
B さん

# [ミッション3 計画・予測]

## 更なる改善

# 振り返って・・・



経理  
Aさん

今回のような利用料金の上昇は  
請求書のタイミングじゃなく、  
上がったときに気づいたら  
もっと早くコスト最適化  
できましたね

確かに！



クラウド管理者  
Bさん

# コスト最適化何からはじめる？

CFMの「計画・予測」に取り組みましょう！



# 予算と予算アラートを設定しましょう

## AWS Budgets

予算を超過する前にアラートを通知したり、アラートが発生したらインスタンスを自動停止する等の自動オペレーションを実行できます。

### 予算

期間  
日次予算は、予測アラート、または日次予算計画の有効化をサポートしません。

月

予算更新タイプ

定期予算  
定期予算は、毎月の請求期間の初日に更新されます。

期限切れになる予算  
月次予算の有効期限が切

開始月

6月 2024

予算設定方法 情報

固定  
月次予算額に照らして追跡する予算を作成します。

予算額 (\$) を入力してください  
先月のコスト: \$1,907.06

2500.00

ひと月 \$2,500 の  
予算を設定

### 予算アラート

アラートのしきい値を設定

しきい値  
このアラートはいつトリガーされる必要がありますか?

80 予算額の %

トリガー  
このアラートは何を使用してトリガーされる必要がありますか?

実際

要約: 実際のコストが予算額 (\$2,000.00) の 80.00% (\$1,600.00) を超えると、アラートのしきい値を超過します。

予算額の 80% で  
アラート通知

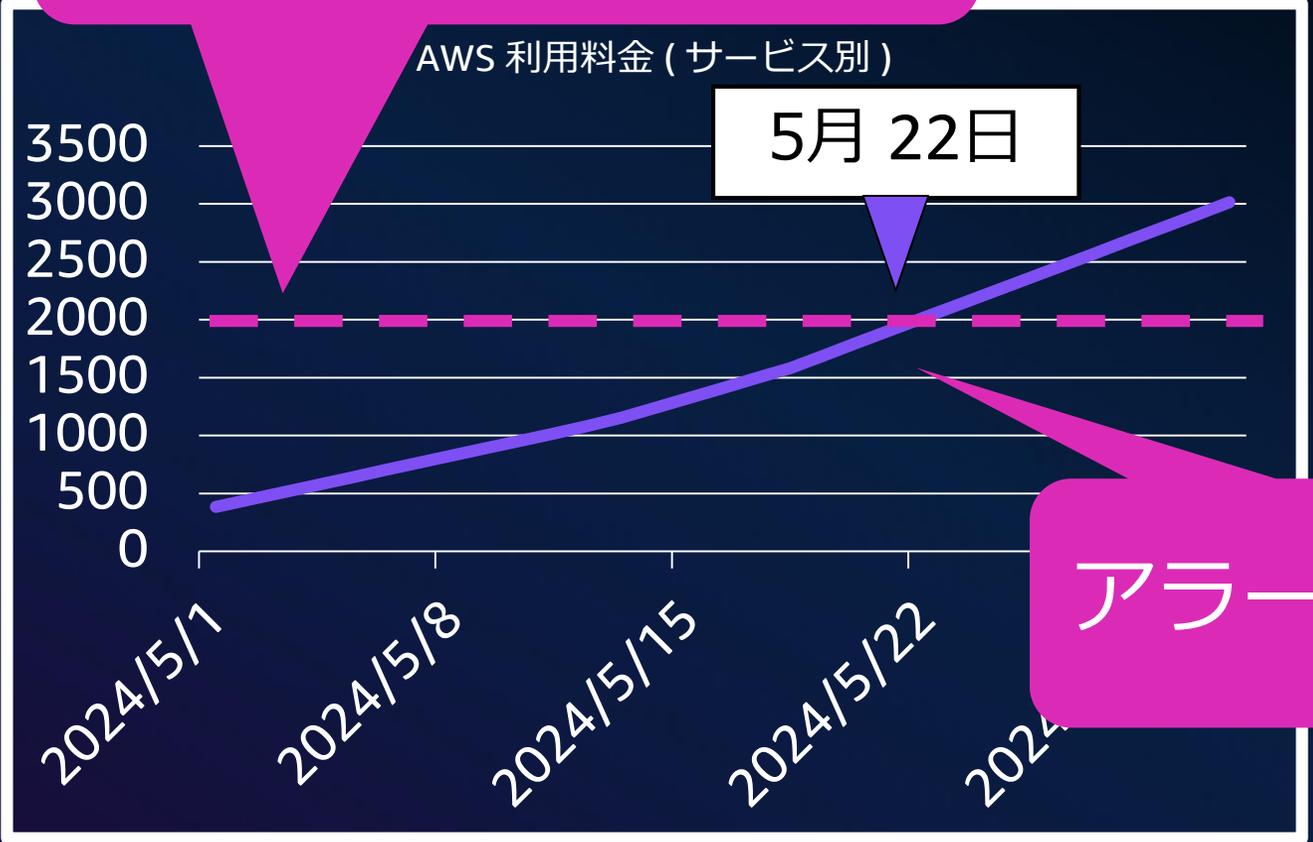
# 予算・予算アラートを設定すると こうなります

予算 : \$2,500  
予算アラート : \$2,000

これなら  
安心!



経理  
Aさん



# 異常検知アラートも設定しましょう

## AWS Cost Anomaly Detection

AWS サービスの利用状況を機械学習モデルによって学習して、ベースラインを大きく逸脱した値を検知することができます。

### モニターの設定

AWS コスト管理 > Cost Anomaly Detection > モニターを作成

ステップ 1  
Choose monitor type

ステップ 2  
Configure alert subscriptions

#### Choose monitor type 情報

#### モニタータイプ

支出をモニタリングする方法

- AWS のサービス - 推奨事項**  
このモニターは、使用する各サービスを個別に評価し、小さな異常を検出できるようにします。異常しい値は、サービス使用履歴パターンに基づいて自動的に調整されます。
- 連結アカウント**  
このモニターは、個々の連結アカウントの総支出を評価します。このモニターは、組織が連結アカウント別にチーム (または製品、サービス、環境) を定義する場合に役立ちます。

モニタータイプは  
AWS サービスが  
おすすめ！

# 異常検知アラートを設定すると こうなります



サービス別の  
異常がわかる  
んですね！



経理  
Aさん

\* AWS サービスの利用状況を機械学習モデルによって学習し、ベースラインを大きく逸脱した値を検知します。

# ここまでのまとめ

- **予算アラートと異常検知アラートを設定しよう！**
- AWS Budgets を使って予算と予算アラートを作成し、利用料金全体や、プロジェクトごとの利用料金が計画と異なる推移をしたらすぐ検知できるように監視しましょう
- AWS Cost Anomaly Detection を使って、自動でサービス毎のコストの異常検知をしましょう

# まとめ

# まとめ

このセッションではユースケースを通してCFMのフレームワークを使ったAWSのコスト最適化の進め方をご紹介します。

# まとめ

## 1. 可視化



- AWS Cost Explorer / AWS Cost and Usage Reports で分析
- タグを使ったリソースの管理、タグ無しリソースの検索・是正方法
- データエクスポートと CUR2.0 で簡単にダッシュボードを作る

## 2. 最適化



- Cost Optimization Hub でコスト最適化の機会を知る

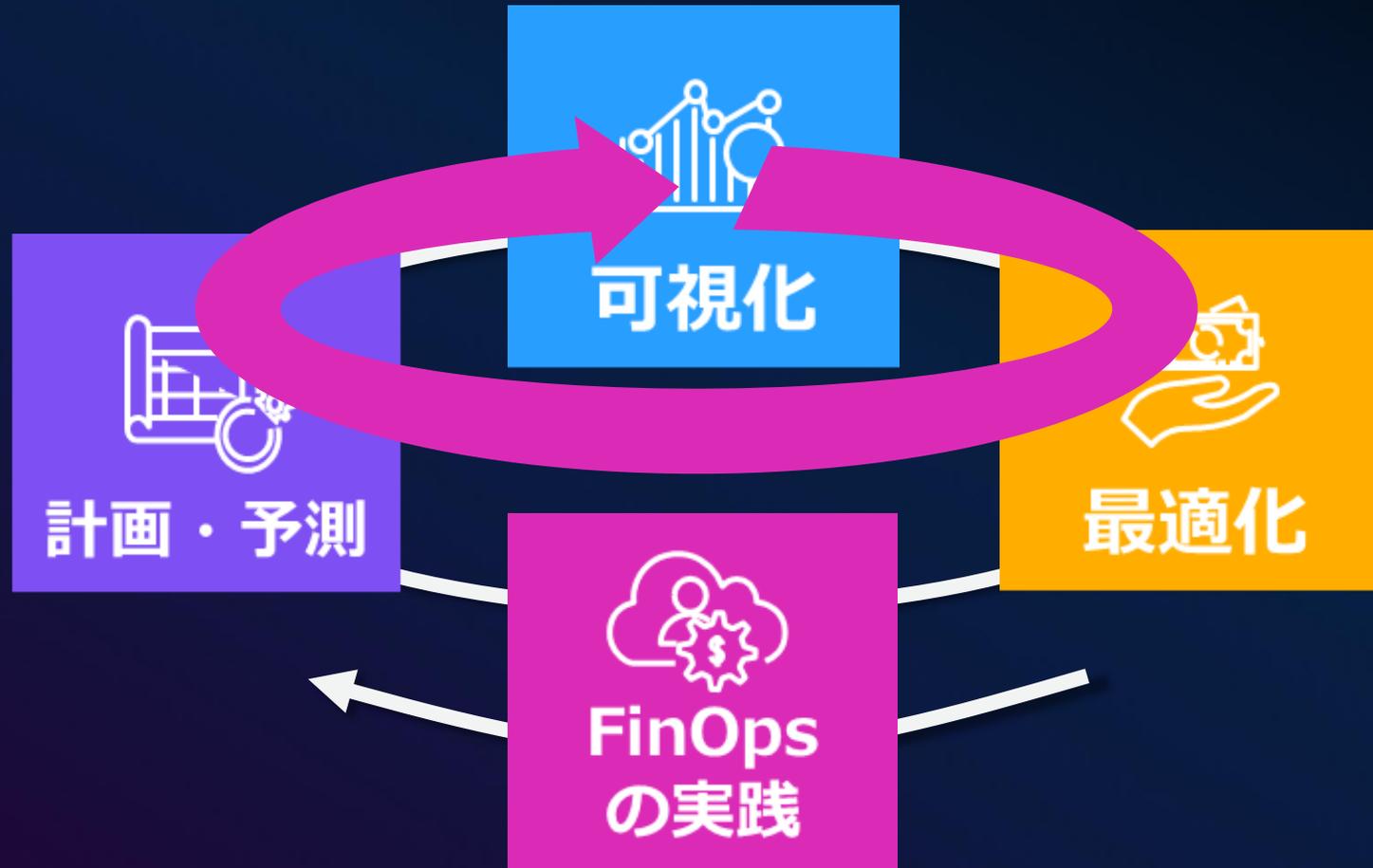
## 3. 計画・予測



- AWS Budgets を使って予算を立てる/アラートを設定する
- AWS Cost Anomaly Detection を使ってコストの異常を検知する

# CFMの「FinOpsの実践」に取り組みましょう！

可視化、最適化、計画・予測に継続的に取り組みましょう。



# まとめ

コストを見るところから始めてみてはいかがでしょうか？  
まずは AWS Cost Explorer にアクセスしてみてください！



# Thank you!

石王 愛

