



# FOCUS

## FinOps Open Cost and Usage Specification

### Version

Publication version 1.1

Copyright © 2024 - FinOps Open Cost and Usage Specification (FOCUS) a Series of the Joint Development Foundation Projects, LLC. Linux Foundation [trademark](#), and document use rules apply.

### Status of This Document


This section describes the status of this document at the time of its publication.

This is a published release of the FinOps Open Cost and Usage Specification.

This document was produced by a group operating under the Joint Development Foundation Projects agreement. FOCUS maintains a public list of any patent disclosures made in connection with the deliverables of the group; [that page](#) also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information.

## Document Use License

Copyright (c) Joint Development Foundation Projects, LLC, FinOps Open Cost and Usage Specification (FOCUS) Series and its contributors. The materials in this repository are made available under the Creative Commons Attribution 4.0 International license (CC-BY-4.0), available at [\[https://creativecommons.org/licenses/by/4.0/legalcode\]\(https://creativecommons.org/licenses/by/4.0/legalcode\)](https://creativecommons.org/licenses/by/4.0/legalcode).

Shield:  CC BY 4.0

This work is made available under: [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/legalcode).



THESE MATERIALS ARE PROVIDED "AS IS." The parties expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to the materials. The entire risk as to implementing or otherwise using the materials is assumed by the implementer and user. IN NO EVENT WILL THE PARTIES BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS DELIVERABLE OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER MEMBER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This document is governed by the Patent Policy Option 4: W3C Mode: See [project charter](#).

## Abstract

FOCUS is an open-source specification for billing data. It defines a common schema for billing data, aligns terminology with the FinOps Framework and defines a minimum set of requirements for billing data. The specification provides clear guideline for billing data generators to produce FinOps-serviceable data. The specification enables FinOps practitioners to perform common FinOps capabilities such as chargeback, cost allocation, budgeting and forecasting etc. using a generic set of instructions, regardless of the origin of the FOCUS compatible dataset.

# Working Group

## Maintainers

Thanks to the following FOCUS Maintainers for their leadership and contributions to the FOCUS Release **v1.1** specification.

- Alex Hullah (Goldman Sachs)
- Christopher Harris (Datadog)
- Irena Jurica (Neos)
- Joaquin Prado (FinOps Foundation)
- Karl Kraft (Walmart)
- Larry Advey (Twilio)
- Michael Flanakin (Microsoft)
- Riley Jenkins (Domo)
- Shawn Alpay (Envisor / FinOps Foundation)
- Udam Dewaraja (FinOps Foundation)
- Zach Erdman (Amazon Web Services)

## Contributors

Thanks to the following FOCUS members for their contributions to the FOCUS Release **v1.1** specification.

- Adam Schwartz (Ernst & Young)
- Andrew Qu (Everest)
- Arun Ramakrishnan (Oracle)
- Erik Peterson (CloudZero)
- George Parker (Salesforce)
- Graham Murphy (Australian Retirement Trust)
- Janine Pickard-Green (MagicOrange Group Limited)
- John Grubb (Platform.sh)
- Joseph John (Microsoft)
- Marc Perreaut (Amadeus)
- Rob Martin (FinOps Foundation)
- Rupa Patel (Google)
- Sanjna Srivatsa (VMWare)
- Sonal Garg (Kyndryl)
- Tim Wright (Google)

## Steering Committee Members

Thanks to the following FOCUS Steering Committee members for their leadership on the FOCUS specification.

- Anne Johnston (Capital One)
- Michael Flanakin (Microsoft)
- Mike Fuller (FinOps Foundation)
- Roy Wolman (Amazon Web Services)
- Sarah McMullin (Google)
- Tim O'Brien (Walmart)

# Table of Contents

## 1. Introduction

- 1.1. Background and History
- 1.2. Intended Audience
- 1.3. Scope
- 1.4. Design Principles
- 1.5. Design Notes
- 1.6. Typographic Conventions
- 1.7. FOCUS Feature level
- 1.8. Conformance Checkers and Validators

## 2. Columns

- 2.1. Availability Zone
- 2.2. Billed Cost
- 2.3. Billing Account ID
- 2.4. Billing Account Name
- 2.5. Billing Currency
- 2.6. Billing Period End
- 2.7. Billing Period Start
- 2.8. Capacity Reservation ID
- 2.9. Capacity Reservation Status
- 2.10. Charge Category
- 2.11. Charge Class
- 2.12. Charge Description
- 2.13. Charge Frequency
- 2.14. Charge Period End
- 2.15. Charge Period Start
- 2.16. Commitment Discount Category
- 2.17. Commitment Discount ID
- 2.18. Commitment Discount Name
- 2.19. Commitment Discount Quantity
- 2.20. Commitment Discount Status
- 2.21. Commitment Discount Type
- 2.22. Commitment Discount Unit
- 2.23. Consumed Quantity
- 2.24. Consumed Unit
- 2.25. Contracted Cost
- 2.26. Contracted Unit Price
- 2.27. Effective Cost
- 2.28. Invoice Issuer
- 2.29. List Cost
- 2.30. List Unit Price
- 2.31. Pricing Category
- 2.32. Pricing Quantity
- 2.33. Pricing Unit
- 2.34. Provider
- 2.35. Publisher
- 2.36. Region ID
- 2.37. Region Name
- 2.38. Resource ID
- 2.39. Resource Name
- 2.40. Resource Type
- 2.41. Service Category
- 2.42. Service Name

- 2.43. Service Subcategory
- 2.44. SKU ID
- 2.45. SKU Meter
- 2.46. SKU Price Details
- 2.47. SKU Price ID
- 2.48. Sub Account ID
- 2.49. Sub Account Name
- 2.50. Tags

### 3. Attributes

- 3.1. Column Naming and Ordering
- 3.2. Currency Code Format
- 3.3. Date/Time Format
- 3.4. Discount Handling
- 3.5. Key-Value Format
- 3.6. Null Handling
- 3.7. Numeric Format
- 3.8. String Handling
- 3.9. Unit Format

### 4. Metadata

- 4.1. Data Generator
- 4.2. Schema

### 5. Use Case Library

### 6. Glossary

### 7. Appendix

- 7.1. Commitment Discounts
- 7.2. Grouping constructs for resources or services
- 7.3. Origination of Cost Data
- 7.4. Examples

# 1. Introduction

*This section is non-normative.*

FOCUS aims to establish a community-driven specification for consumption-based billing data. Due to the lack of a broadly adopted specification, infrastructure and services [providers](#) have resorted to proprietary billing schemas and terminology. The lack of conformance amongst the billing data generators has forced FinOps practitioners to employ disparate, best-effort schemes which each *practitioner* must develop individually for each *provider* to perform essential FinOps capabilities such as chargeback, cost allocation, budgeting and forecasting.

The FOCUS specification's schema definition and FinOps-aligned terminology provide a clear guide for producing FinOps-serviceable billing datasets. Datasets conforming to FOCUS enable FinOps practitioners to perform common FinOps capabilities, like the ones mentioned above, using a generic set of instructions, regardless of the origin of the dataset.

## 1.1. Background and History

This project is supported by the [FinOps Foundation](#). This work initially started under the Open Billing working group under the FinOps Foundation. The decision was made in Jan 2023 to begin to migrate the work to a newly formed project under the Linux Foundation called the FinOps Open Cost and Usage Specification (FOCUS) to better support the creation of a specification.

## 1.2. Intended Audience

This specification is designed to be used by three major groups:

- Billing data generators: Infrastructure and services *providers* that bill based on consumption, such as (but not limited to):
  - [Cloud Service Providers \(CSPs\)](#)
  - Software as a Service (SaaS) platforms
  - [Managed Service Providers \(MSPs\)](#)
  - Internal infrastructure and service platforms
- FinOps tool *providers*: Organizations that provide tools to assist with FinOps
- FinOps practitioners: Organizations and individuals consuming billing data for doing FinOps

## 1.3. Scope

The FOCUS working group will develop an open-source specification for billing data. The schema will define data [dimensions](#), [metrics](#), a set of attributes about billing data, and a common lexicon for describing billing data.

## 1.4. Design Principles

The following principles were considered while building the specification.

#### **1.4.1. FOCUS is an iterative, living specification**

- Incremental iterations of the specification released regularly will provide higher value to practitioners and allow feedback as the specification develops. The goal is not to get to a complete, finished specification in one pass.

#### **1.4.2. Working backward with ease of adoption**

- Aim to work backward from essential FinOps capabilities that practitioners need to perform to prioritize the dimensions, metrics and attributes of the cost and usage data that should be defined in the specification to fulfill that capability.
- Be FinOps scenario-driven. Define columns that answer scenario questions; don't look for scenarios to fit a column, each column must have a use case.
- Don't add dimensions or metrics to the specification just because it can be added.
- When defining the specification, consideration should be made to existing data already in the major providers' (AWS, GCP, Azure, OCI) datasets.
- As long as it solves the FinOps use case, there should be a preference to align with data that is already present in a majority of the major providers.
- Strive for simplicity. However, prioritize accuracy, clarity, and consistency.
- Strive to build columns that serve a single purpose, with clear and concise names and values.
- The specification should allow data to be presented free from jargon, using simple understandable terms, and be approachable.
- Naming and terms used should be carefully considered to avoid using terms for which the definition could be confused by the reader. If a term must be used which has either an unclear or multiple definitions, it should be clarified in the [glossary](#).
- The specification should provide all of the data elements necessary for the [Capabilities](#).

#### **1.4.3. Provider-neutral approach by default**

- While the schema, naming, terminology, and attributes of many providers are reviewed during development, this specification aims to be provider-neutral.
- Contributors must take care to ensure the specification examines how each decision relates to each of the major cloud providers and SaaS vendors, not favoring any single one.
- In some cases, the approach may closely resemble one or more provider's implementations, while in other cases, the approach might be new. In all cases, the FOCUS group (community composed of FinOps practitioners, Cloud and SaaS providers and FinOps vendors) will attempt to prioritize enabling FinOps [Capabilities](#) and alignment with the FinOps [Framework](#).

#### **1.4.4. Extensibility**

- The initial specification aims to introduce a common schema and terminology for billing datasets produced by Cloud Service Providers (CSPs).
- The specification, however, aims to be extensible to SaaS products and other types of cost datasets.
- Future versions of the specification will look to expand the content to support a broader set of prioritized FinOps capabilities.



## 1.5. Design Notes

### 1.5.1. Optimize for data analysis

- Optimize columns for data analysis at scale and avoid the requirement of splitting or parsing values.
- Avoid complex JSON structures when an alternative columnar structure is possible.
- Facilitate the inclusion of data necessary for a system of record for cost and usage data to consume.

### 1.5.2. Consistency helps with clarity

- Where possible, use consistent names that will naturally create associations between related columns in the specification.
- Column naming must strictly follow the [column naming conventions](#).
- Use established standards (e.g., ISO8601 for dates, ISO4217 for currency).

## 1.6. Typographic Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [BCP14](#) [[RFC2119](#)][[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

## 1.7. FOCUS Feature level

Under each column defined in the FOCUS specification, there exists a 'Feature level' designation that describes the column as 'Mandatory', 'Conditional', or 'Optional'. Feature level is designated based on the following criteria described in the normative requirements in each column definition:

- If the existence of a column is described with MUST with no conditions of when it applies, then the feature level is designated as 'Mandatory'.
- If the existence of a column is described as MUST with conditions of when it applies, then the feature level is designated as 'Conditional'.
- If the existence of a column is described as RECOMMENDED, then the feature level is designated as 'Recommended'.
- If the existence of a column is described as MAY, then the feature level is designated as 'Optional'.

## 1.8. Conformance Checkers and Validators

There are no current resources available to test for specification conformance or validators to run on sample data. When one becomes available, this section of the specification will be updated with details.

## 2. Columns

The FOCUS specification defines a group of columns that provide qualitative values (such as dates, resource, and provider information) categorized as "dimensions" and quantitative values (numeric values) categorized as "metrics" that can be used for performing various [FinOps capabilities](#). Metrics are commonly used for aggregations (sum, multiplication, averaging etc.) and statistical operations within the dataset. Dimensions are commonly used to categorize, filter, and reveal details in your data when combined with metrics. The Columns are presented in Alphabetical order.

### 2.1. Availability Zone

An [availability zone](#) is a provider-assigned identifier for a physically separated and isolated area within a Region that provides high availability and fault tolerance. Availability Zone is commonly used for scenarios like analyzing cross-zone data transfer usage and the corresponding cost based on where [resources](#) are deployed.

The AvailabilityZone column is RECOMMENDED to be present in a [FOCUS dataset](#) when the provider supports deploying resources or services within an *availability zone*. This column MUST be of type String and MAY contain null values when a charge is not specific to an *availability zone*.

#### 2.1.1. Column ID

AvailabilityZone

#### 2.1.2. Display Name

Availability Zone

#### 2.1.3. Description

A provider-assigned identifier for a physically separated and isolated area within a Region that provides high availability and fault tolerance.

#### 2.1.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Recommended
Allows nulls	True
Data type	String

Constraint	Value
Value format	<not specified>

### 2.1.5. Introduced (version)

0.5

## 2.2. Billed Cost

The [billed cost](#) represents a charge serving as the basis for invoicing, inclusive of the impacts of all reduced rates and discounts while excluding the [amortization](#) of relevant purchases (one-time or recurring) paid to cover future eligible charges. This cost is denominated in the [Billing Currency](#). The Billed Cost is commonly used to perform FinOps capabilities that require cash-basis accounting such as cost allocation, budgeting, and invoice reconciliation.

The BilledCost column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column MUST be of type Decimal, MUST conform to [Numeric Format](#), and be denominated in the BillingCurrency. The sum of the BilledCost for [rows](#) in a given [billing period](#) MUST match the sum of the invoices received for that *billing period* for a [billing account](#).

### 2.2.1. Column ID

BilledCost

### 2.2.2. Display Name

Billed Cost

### 2.2.3. Description

A charge serving as the basis for invoicing, inclusive of all reduced rates and discounts while excluding the *amortization* of upfront charges (one-time or recurring).

### 2.2.4. Content constraints

Constraint	Value
Column type	Metric
Feature level	Mandatory
Allows nulls	False
Data type	Decimal
Value format	<a href="#">Numeric Format</a>

Constraint	Value
Number range	Any valid decimal value

### 2.2.5. Introduced (version)

0.5

## 2.3. Billing Account ID

A Billing Account ID is a provider-assigned identifier for a [billing account](#). *Billing accounts* are commonly used for scenarios like grouping based on organizational constructs, invoice reconciliation and cost allocation strategies.

The BillingAccountId column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST NOT contain null values. BillingAccountId MUST be a globally unique identifier within a provider.

See [Appendix: Grouping constructs for resources or services](#) for details and examples of the different grouping constructs supported by FOCUS.

### 2.3.1. Column ID

BillingAccountId

### 2.3.2. Display Name

Billing Account ID

### 2.3.3. Description

The identifier assigned to a *billing account* by the provider.

### 2.3.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

### 2.3.5. Introduced (version)

0.5

## 2.4. Billing Account Name

A Billing Account Name is a display name assigned to a [billing account](#). *Billing accounts* are commonly used for scenarios like grouping based on organizational constructs, invoice reconciliation and cost allocation strategies.

The BillingAccountName column MUST be present in a [FOCUS dataset](#) and MUST NOT be null when the provider supports assigning a display name for the *billing account*. This column MUST be of type String. BillingAccountName MUST be unique within a customer when a customer has more than one *billing account*.

See [Appendix: Grouping constructs for resources or services](#) for details and examples of the different grouping constructs supported by FOCUS.

### 2.4.1. Column ID

BillingAccountName

### 2.4.2. Display Name

Billing Account Name

### 2.4.3. Description

The display name assigned to a *billing account*.

### 2.4.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.4.5. Introduced (version)

0.5

## 2.5. Billing Currency

[Billing currency](#) is an identifier that represents the currency that a charge for [resources](#) or [services](#) was billed in. Billing Currency is commonly used in scenarios where costs need to be grouped or aggregated.

The BillingCurrency column MUST be present in a [FOCUS dataset](#). BillingCurrency MUST match the currency used in the invoice generated by the invoice issuer. This column MUST be of type String and MUST NOT contain null values. BillingCurrency MUST conform to [Currency Code Format](#) requirements.

### 2.5.1. Column ID

BillingCurrency

### 2.5.2. Display Name

Billing Currency

### 2.5.3. Description

Represents the currency that a charge was billed in.

### 2.5.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<a href="#">Currency Code Format</a>

### 2.5.5. Introduced (version)

0.5

## 2.6. Billing Period End

Billing Period End represents the [exclusive](#) end date and time of a [billing period](#). For example, a time period where [BillingPeriodStart](#) is '2024-01-01T00:00:00Z' and BillingPeriodEnd is '2024-02-01T00:00:00Z' includes charges for January, since BillingPeriodStart is [inclusive](#), but does not include charges for February since BillingPeriodEnd is *exclusive*.

The BillingPeriodEnd column MUST be present in a [FOCUS dataset](#). This column MUST be of type [Date/Time](#)

[Format](#), MUST be an *exclusive* value, and MUST NOT contain null values. The sum of the [BilledCost](#) column for [rows](#) in a given *billing period* MUST match the sum of the invoices received for that *billing period* for a [billing account](#).

### 2.6.1. Column ID

BillingPeriodEnd

### 2.6.2. Display Name

Billing Period End

### 2.6.3. Description

The [exclusive](#) end date and time of a [billing period](#).

### 2.6.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	Date/Time
Value format	<a href="#">Date/Time Format</a>

### 2.6.5. Introduced (version)

0.5

## 2.7. Billing Period Start

Billing Period Start represents the [inclusive](#) start date and time of a [billing period](#). For example, a time period where BillingPeriodStart is '2024-01-01T00:00:00Z' and [BillingPeriodEnd](#) is '2024-02-01T00:00:00Z' includes charges for January, since BillingPeriodStart is inclusive, but does not include charges for February since BillingPeriodEnd is [exclusive](#).

The BillingPeriodStart column MUST be present in a [FOCUS dataset](#), MUST be of type [Date/Time Format](#), MUST be an *inclusive* value, and MUST NOT contain null values. The sum of the [BilledCost](#) metric for [rows](#) in a given *billing period* MUST match the sum of the invoices received for that *billing period* for a [billing account](#).

### 2.7.1. Column ID

BillingPeriodStart

### 2.7.2. Display Name

Billing Period Start

### 2.7.3. Description

The [inclusive](#) start date and time of a [billing period](#).

### 2.7.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	Date/Time
Value format	<a href="#">Date/Time Format</a>

### 2.7.5. Introduced (version)

0.5

## 2.8. Capacity Reservation ID

A Capacity Reservation ID is the identifier assigned to a [capacity reservation](#) by the provider. Capacity Reservation ID is commonly used for scenarios to allocate charges for capacity reservation usage.

The CapacityReservationId column adheres to the following requirements:

- CapacityReservationId MUST be present in a [FOCUS dataset](#) when the provider supports *capacity reservations* and MUST be of type String.
- CapacityReservationId SHOULD NOT be null when a charge is related to a capacity reservation.
- CapacityReservationId MUST NOT be null when a charge represents the unused portion of a *capacity reservation*.
- CapacityReservationId MUST be null when a charge is not related to a *capacity reservation*.
- CapacityReservationId MUST ensure global uniqueness within the provider and SHOULD be a fully-qualified identifier.

### 2.8.1. Column ID



CapacityReservationId

## 2.8.2. Display Name

Capacity Reservation ID

## 2.8.3. Description

The identifier assigned to a *capacity reservation* by the provider.

## 2.8.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

## 2.8.5. Introduced (version)

1.1

## 2.9. Capacity Reservation Status

Capacity Reservation Status indicates whether the charge represents either the consumption of the [capacity reservation](#) identified in the CapacityReservationId column or when the *capacity reservation* is unused.

The CapacityReservationStatus column adheres to the following requirements:

- CapacityReservationStatus MUST be present in a [FOCUS dataset](#) when the provider supports *capacity reservations* and MUST be of type String.
- CapacityReservationStatus MUST be null when CapacityReservationId is null.
- CapacityReservationStatus MUST NOT be null when CapacityReservationId is not null and [ChargeCategory](#) is "Usage".
- CapacityReservationStatus MUST be one of the allowed values.
- CapacityReservationStatus MUST label all unused *capacity reservation* charges and MUST label used *capacity reservation* charges if the provider supports it.

### 2.9.1. Column ID

CapacityReservationStatus

## 2.9.2. Display Name

Capacity Reservation Status

## 2.9.3. Description

Indicates whether the charge represents either the consumption of a *capacity reservation* or when a *capacity reservation* is unused.

## 2.9.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	Allowed Values

Allowed values:

Value	Description
Used	Charges that utilized a specific amount of a capacity reservation.
Unused	Charges that represent the unused portion of a capacity reservation.

## 2.9.5. Introduced (version)

1.1

## 2.10. Charge Category

Charge Category represents the highest-level classification of a charge based on the nature of how it is billed. Charge Category is commonly used to identify and distinguish between types of charges that may require different handling.

The ChargeCategory column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column is of type String and MUST be one of the allowed values.

### 2.10.1. Column ID

## 2.10.2. Display Name

Charge Category

## 2.10.3. Description

Represents the highest-level classification of a charge based on the nature of how it is billed.

## 2.10.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	Allowed values

Allowed values:

Value	Description
Usage	Positive or negative charges based on the quantity of a service or resource that was consumed over a given period of time including refunds.
Purchase	Positive or negative charges for the acquisition of a service or resource bought upfront or on a recurring basis including refunds.
Tax	Positive or negative applicable taxes that are levied by the relevant authorities including refunds. Tax charges may vary depending on factors such as the location, jurisdiction, and local or federal regulations.
Credit	Positive or negative charges granted by the provider for various scenarios e.g promotional credits or corrections to promotional credits.
Adjustment	Positive or negative charges the provider applies that do not fall into other category values.

## 2.10.5. Introduced (version)

0.5

## 2.11. Charge Class

Charge Class indicates whether the row represents a correction to a previously invoiced [billing period](#). Charge Class is commonly used to differentiate corrections from regularly incurred charges.

The ChargeClass column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST be "Correction" when the row represents a correction to a previously invoiced *billing period*. ChargeClass MUST be null when it is not a correction or when it is a correction within the current *billing period*.

### 2.11.1. Column ID

ChargeClass

### 2.11.2. Display Name

Charge Class

### 2.11.3. Description

Indicates whether the row represents a correction to a previously invoiced *billing period*.

### 2.11.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	True
Data type	String
Value format	Allowed values

Allowed values:

Value	Description
Correction	Correction to a previously invoiced <i>billing period</i> (e.g., refunds and credit modifications).

### 2.11.5. Introduced (version)

1.0

## 2.12. Charge Description

A Charge Description provides a high-level context of a [row](#) without requiring additional discovery. This column is a self-contained summary of the charge's purpose and price. It typically covers a select group of corresponding details across a billing dataset or provides information not otherwise available.

The ChargeDescription column MUST be present in a [FOCUS dataset](#), MUST be of type String, and SHOULD NOT be null. Providers SHOULD specify the length of this column in their publicly available documentation.

### 2.12.1. Column ID

ChargeDescription

### 2.12.2. Display Name

Charge Description

### 2.12.3. Description

Self-contained summary of the charge's purpose and price.

### 2.12.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.12.5. Introduced (version)

1.0-preview

## 2.13. Charge Frequency

Charge Frequency indicates how often a charge will occur. Along with the [charge period](#) related columns, the Charge Frequency is commonly used to understand recurrence periods (e.g., monthly, yearly), forecast upcoming charges, and differentiate between one-time and recurring fees for purchases.

The ChargeFrequency column is RECOMMENDED be present in a [FOCUS dataset](#) and MUST NOT be null. This column is of type String and MUST be one of the allowed values. When [ChargeCategory](#) is "Purchase", ChargeFrequency MUST NOT be "Usage-Based".

### 2.13.1. Column ID

ChargeFrequency

2.13.2. Display Name

Charge Frequency

2.13.3. Description

Indicates how often a charge will occur.

2.13.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Recommended
Allows nulls	False
Data type	String
Value format	Allowed values

Allowed values:

Value	Description
One-Time	Charges that only happen once and will not repeat. One-time charges are typically recorded on the hour or day when the cost was incurred.
Recurring	Charges that repeat on a periodic cadence (e.g., weekly, monthly) regardless of whether the product or service was used. Recurring charges typically happen on the same day or point within every period. The charge date does not change based on how or when the service is used.
Usage-Based	Charges that repeat every time the service is used. Usage-based charges are typically recorded hourly or daily, based on the granularity of the cost data for the period when the service was used (referred to as charge period). Usage-based charges are not recorded when the service is not used.

2.13.5. Introduced (version)

1.0-preview

2.14. Charge Period End

Charge Period End represents the [exclusive](#) end date and time of a [charge period](#). For example, a time period where [ChargePeriodStart](#) is '2024-01-01T00:00:00Z' and [ChargePeriodEnd](#) is '2024-01-02T00:00:00Z' includes charges for January 1, since [ChargePeriodStart](#) is [inclusive](#), but does not include charges for January 2 since [ChargePeriodEnd](#) is [exclusive](#).

ChargePeriodEnd MUST be present in a [FOCUS dataset](#), MUST be of type Date/Time, MUST be an *exclusive* value, and MUST NOT contain null values. ChargePeriodEnd MUST match the ending date and time boundary of the effective period of the charge.

2.14.1. Column ID

ChargePeriodEnd

2.14.2. Display Name

Charge Period End

2.14.3. Description

The [exclusive](#) end date and time of a charge period.

2.14.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	Date/Time
Value format	<a href="#">Date/Time Format</a>

2.14.5. Introduced (version)

0.5

2.15. Charge Period Start

Charge Period Start represents the [inclusive](#) start date and time within a [charge period](#). For example, a time period where ChargePeriodStart is '2024-01-01T00:00:00Z' and [ChargePeriodEnd](#) is '2024-01-02T00:00:00Z' includes charges for January 1, since ChargePeriodStart is *inclusive*, but does not include charges for January 2 since ChargePeriodEnd is [exclusive](#).

ChargePeriodStart MUST be present in a [FOCUS dataset](#), MUST be of type Date/Time, MUST be an *inclusive* value, and MUST NOT contain null values. ChargePeriodStart MUST match the beginning date and time boundary of the effective period of the charge.

### 2.15.1. Column ID

ChargePeriodStart

### 2.15.2. Display Name

Charge Period Start

### 2.15.3. Description

The [inclusive](#) start date and time within a [charge period](#).

### 2.15.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	Date/Time
Value format	<a href="#">Date/Time Format</a>

### 2.15.5. Introduced (version)

0.5

## 2.16. Commitment Discount Category

Commitment Discount Category indicates whether the [commitment discount](#) identified in the CommitmentDiscountId column is based on usage quantity or cost (aka "spend"). The CommitmentDiscountCategory column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountCategory column MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*. This column MUST be of type String, MUST be null when [CommitmentDiscountId](#) is null, and MUST NOT be null when CommitmentDiscountId is not null. The CommitmentDiscountCategory MUST be one of the allowed values.

### 2.16.1. Column ID

CommitmentDiscountCategory



## 2.16.2. Display Name

Commitment Discount Category

## 2.16.3. Description

Indicates whether the *commitment discount* identified in the CommitmentDiscountId column is based on usage quantity or cost (aka "spend").

## 2.16.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	Allowed Values

Allowed values:

Value	Description
Spend	Commitment discounts that require a predetermined amount of spend.
Usage	Commitment discounts that require a predetermined amount of usage.

## 2.16.5. Introduced (version)

1.0-preview

# 2.17. Commitment Discount ID

A Commitment Discount ID is the identifier assigned to a [commitment discount](#) by the provider. Commitment Discount ID is commonly used for scenarios like chargeback for *commitments* and savings per *commitment discount*. The CommitmentDiscountId column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountId column MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*. This column MUST be of type String and MUST NOT contain null values when a charge is related to a *commitment discount*. When a charge is not associated with a *commitment discount*, the column MUST be null. CommitmentDiscountId MUST ensure global uniqueness within the provider and SHOULD be a fully-qualified identifier.

## 2.17.1. Column ID

CommitmentDiscountId

### 2.17.2. Display Name

Commitment Discount ID

### 2.17.3. Description

The identifier assigned to a *commitment discount* by the provider.

### 2.17.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.17.5. Introduced (version)

1.0-preview

## 2.18. Commitment Discount Name

A Commitment Discount Name is the display name assigned to a [commitment discount](#). The CommitmentDiscountName column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountName column MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*. This column MUST be of type String. The CommitmentDiscountName value MUST be null if the charge is not related to a *commitment discount* and MAY be null if a display name cannot be assigned to a *commitment discount*. CommitmentDiscountName MUST NOT be null if a display name can be assigned to a *commitment discount*.

### 2.18.1. Column ID

CommitmentDiscountName

### 2.18.2. Display Name

### 2.18.3. Description

The display name assigned to a *commitment discount*.

### 2.18.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.18.5. Introduced (version)

1.0-preview

## 2.19. Commitment Discount Quantity

Commitment Discount Quantity is the amount of a [commitment discount](#) purchased or accounted for in *commitment discount* related [rows](#) that is denominated in [Commitment Discount Units](#). The aggregated Commitment Discount Quantity across purchase records, pertaining to a particular [Commitment Discount ID](#) during its [term](#), represents the total Commitment Discount Units acquired with that commitment discount. For committed usage, the Commitment Discount Quantity is either the number of Commitment Discount Units consumed by a *row* that is covered by a *commitment discount* or is the unused portion of a *commitment discount* over a *charge period*. Commitment Discount Quantity is commonly used in *commitment discount* analysis and optimization use cases and only applies to *commitment discounts*, not [negotiated discounts](#).

When [CommitmentDiscountCategory](#) is "Usage" (usage-based *commitment discounts*), the Commitment Discount Quantity reflects the predefined amount of usage purchased or consumed. If [commitment discount flexibility](#) is applicable, this value may be further transformed based on additional, provider-specific requirements. When [CommitmentDiscountCategory](#) is "Spend" (spend-based *commitment discounts*), the Commitment Discount Quantity reflects the predefined amount of spend purchased or consumed.

The [CommitmentDiscountQuantity](#) column adheres to the following requirements:

- [CommitmentDiscountQuantity](#) MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*.
- [CommitmentDiscountQuantity](#) MUST be of type Decimal and MUST conform to [Numeric Format](#) requirements.
- [CommitmentDiscountQuantity](#) MAY be null or any valid decimal value if [CommitmentDiscountId](#) is not null and [ChargeClass](#) is "Correction".

In cases where the [ChargeCategory](#) is "Purchase", [CommitmentDiscountId](#) is not null, and [ChargeClass](#) is not "Correction", the following applies:

- When [ChargeFrequency](#) is "One-Time", and CommitmentDiscountId is not null, CommitmentDiscountQuantity MUST be the positive quantity of CommitmentDiscountUnits, paid fully or partially upfront, that is eligible for consumption over the *commitment discount's term*.
- When ChargeFrequency is "Recurring", and CommitmentDiscountId is not null, CommitmentDiscountQuantity MUST be the positive quantity of CommitmentDiscountUnits that is eligible for consumption for each *charge period* that corresponds with the purchase.

In cases where the ChargeCategory is "Usage", CommitmentDiscountId is not null, and ChargeClass is not "Correction", the following applies:

- When [CommitmentDiscountStatus](#) is "Used", CommitmentDiscountQuantity MUST be the positive, metered quantity of CommitmentDiscountUnits that is consumed over the *row's charge period*.
- When CommitmentDiscountStatus is "Unused", CommitmentDiscountQuantity MUST be the remaining, positive, unused quantity of CommitmentDiscountUnits for the *row's charge period*.

CommitmentDiscountQuantity MUST be null in all other cases.

### 2.19.1. Column ID

CommitmentDiscountQuantity

### 2.19.2. Display Name

Commitment Discount Quantity

### 2.19.3. Description

The amount of a *commitment discount* purchased or accounted for in *commitment discount* related *rows* that is denominated in Commitment Discount Units.

### 2.19.4. Usability Constraints

**Aggregation:** When aggregating Commitment Discount Quantity for commitment utilization calculations, it's important to exclude *commitment discount* purchases (i.e. when ChargeCategory is "Purchase") that are paid to cover future eligible charges (e.g., *commitment discount*). Otherwise, when accounting for all upfront or accrued purchases, it's important to exclude *commitment discount* usage (i.e. when ChargeCategory is "Usage"). This exclusion helps prevent double counting of these quantities in the aggregation.

### 2.19.5. Content constraints

Constraint	Value
Column type	Metric
Feature level	Conditional
Allows nulls	True

Constraint	Value
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid decimal value

## 2.19.6. Introduced (version)

1.1

## 2.20. Commitment Discount Status

Commitment Discount Status indicates whether the charge corresponds with the consumption of a [commitment discount](#) identified in the CommitmentDiscountId column or the unused portion of the committed amount. The CommitmentDiscountStatus column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountStatus column MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*. This column MUST be of type String, MUST be null when [CommitmentDiscountId](#) is null, and MUST NOT be null when CommitmentDiscountId is not null and [Charge Category](#) is "Usage". CommitmentDiscountStatus MUST be one of the allowed values.

### 2.20.1. Column ID

CommitmentDiscountStatus

### 2.20.2. Display name

Commitment Discount Status

### 2.20.3. Description

Indicates whether the charge corresponds with the consumption of a *commitment discount* or the unused portion of the committed amount.

### 2.20.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True

Constraint	Value
Data type	String
Value format	Allowed Values

Allowed values:

Value	Description
Used	Charges that utilized a specific amount of a commitment discount.
Unused	Charges that represent the unused portion of the commitment discount.

## 2.20.5. Introduced (version)

1.0

## 2.21. Commitment Discount Type

Commitment Discount Type is a provider-assigned name to identify the type of [commitment discount](#) applied to the [row](#). The CommitmentDiscountType column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountType column MUST be present in a [FOCUS dataset](#) when the provider supports *commitment discounts*. This column MUST be of type String, MUST be null when [CommitmentDiscountId](#) is null, and MUST NOT be null when CommitmentDiscountId is not null.

### 2.21.1. Column ID

CommitmentDiscountType

### 2.21.2. Display Name

Commitment Discount Type

### 2.21.3. Description

A provider-assigned identifier for the type of *commitment discount* applied to the *row*.

### 2.21.4. Content constraints

Constraint	Value
Column type	Dimension

Constraint	Value
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.21.5. Introduced (version)

1.0-preview

## 2.22. Commitment Discount Unit

Commitment Discount Unit represents the provider-specified measurement unit indicating how a provider measures the [Commitment Discount Quantity](#) of a [commitment discount](#). The CommitmentDiscountUnit column is only applicable to *commitment discounts* and not [negotiated discounts](#).

The CommitmentDiscountUnit column adheres to the following requirements:

- CommitmentDiscountUnit MUST be present in a [FOCUS dataset](#) when the provider supports [commitment discounts](#).
- CommitmentDiscountUnit MUST be of type String, and the units of measure used in CommitmentDiscountUnit SHOULD adhere to the values and format requirements specified in the [UnitFormat](#) attribute.
- The CommitmentDiscountUnit MUST be the same across all *rows* where CommitmentDiscountQuantity has the same [CommitmentDiscountId](#).
- CommitmentDiscountUnit MAY be null if CommitmentDiscountId is not null and [ChargeClass](#) is "Correction".
- CommitmentDiscountUnit MUST NOT be null when CommitmentDiscountId is not null and ChargeClass is not "Correction".
- CommitmentDiscountUnit MUST be null in all other cases.

In cases where the CommitmentDiscountUnit is not null, the following applies:

- The CommitmentDiscountUnit MUST represent the unit used to measure the *commitment discount*.
- When accounting for [commitment discount flexibility](#), the CommitmentDiscountUnit value SHOULD reflect this consideration.

### 2.22.1. Column ID

CommitmentDiscountUnit

### 2.22.2. Display Name

Commitment Discount Unit

### 2.22.3. Description

The provider-specified measurement unit indicating how a provider measures the Commitment Discount Quantity of a *commitment discount*.

#### 2.22.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<a href="#">Unit</a> <a href="#">Format</a>

#### 2.22.5. Introduced (version)

1.1

### 2.23. Consumed Quantity

The Consumed Quantity represents the volume of a metered SKU associated with a [resource](#) or [service](#) used, based on the [Consumed Unit](#). Consumed Quantity is often derived at a finer granularity or over a different time interval when compared to the [Pricing Quantity](#) (complementary to [Pricing Unit](#)) and focuses on *resource* and *service* consumption, not pricing and cost.

The ConsumedQuantity column adheres to the following requirements:

- ConsumedQuantity MUST be present in a [FOCUS dataset](#) when the provider supports the measurement of usage.
- ConsumedQuantity MUST be of type Decimal and MUST conform to [Numeric Format](#) requirements.
- ConsumedQuantity MUST NOT be null and MUST be a valid positive decimal value if [ChargeCategory](#) is "Usage", [CommitmentDiscountStatus](#) is not "Unused", and [ChargeClass](#) is not "Correction".
- ConsumedQuantity MAY be null or any valid decimal value if ChargeCategory is "Usage", CommitmentDiscountStatus is not "Unused", and ChargeClass is "Correction".
- ConsumedQuantity MUST be null in all other cases.

#### 2.23.1. Column ID

ConsumedQuantity

#### 2.23.2. Display Name

Consumed Quantity



### 2.23.3. Description

The volume of a metered SKU associated with a *resource* or *service* used, based on the Consumed Unit.

### 2.23.4. Content constraints

Constraint	Value
Column type	Metric
Feature level	Conditional
Allows nulls	True
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid decimal value

### 2.23.5. Introduced (version)

1.0

## 2.24. Consumed Unit

The Consumed Unit represents a provider-specified measurement unit indicating how a provider measures usage of a metered SKU associated with a *resource* or *service*. Consumed Unit complements the [Consumed Quantity](#) metric. It is often listed at a finer granularity or over a different time interval when compared to [Pricing Unit](#) (complementary to [Pricing Quantity](#)), and focuses on *resource* and *service* consumption, not pricing and cost.

The ConsumedUnit column adheres to the following requirements:

- ConsumedUnit MUST be present in a [FOCUS dataset](#) when the provider supports the measurement of usage.
- ConsumedUnit MUST be of type String, and the units of measure used in ConsumedUnit SHOULD adhere to the values and format requirements specified in the [UnitFormat](#) attribute.
- ConsumedUnit MUST NOT be null if [ChargeCategory](#) is "Usage", [CommitmentDiscountStatus](#) is not "Unused", and [ChargeClass](#) is not "Correction".
- ConsumedUnit MAY be null if ChargeCategory is "Usage", CommitmentDiscountStatus is not "Unused", and ChargeClass is "Correction".
- ConsumedUnit MUST be null in all other cases.

### 2.24.1. Column ID

ConsumedUnit

### 2.24.2. Display Name

Consumed Unit

### 2.24.3. Description

Provider-specified measurement unit indicating how a provider measures usage of a metered SKU associated with a *resource* or *service*.

### 2.24.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<a href="#">Unit Format</a> recommended

### 2.24.5. Introduced (version)

1.0

## 2.25. Contracted Cost

Contracted Cost represents the cost calculated by multiplying [contracted unit price](#) and the corresponding [Pricing Quantity](#). Contracted Cost is denominated in the [Billing Currency](#) and is commonly used for calculating savings based on negotiation activities, by comparing it with [List Cost](#). If [negotiated discounts](#) are not applicable, the Contracted Cost defaults to the List Cost.

The ContractedCost column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column MUST be of type Decimal, MUST conform to [Numeric Format](#) requirements, and be denominated in the BillingCurrency. When [ContractedUnitPrice](#) is present and not null, multiplying the ContractedUnitPrice by PricingQuantity MUST produce the ContractedCost, except in cases of [ChargeClass](#) "Correction", which may address PricingQuantity or any cost discrepancies independently.

In cases where the ContractedUnitPrice is present and null, the following applies:

- The ContractedCost of a charge calculated based on other charges (e.g., when the [ChargeCategory](#) is "Tax") MUST be calculated based on the ContractedCost of those related charges.
- The ContractedCost of a charge unrelated to other charges (e.g., when the [ChargeCategory](#) is "Credit") MUST match the [BilledCost](#).

### 2.25.1. Column ID

ContractedCost

## 2.25.2. Display Name

Contracted Cost

## 2.25.3. Description

Cost calculated by multiplying *contracted unit price* and the corresponding Pricing Quantity.

## 2.25.4. Usability Constraints

**Aggregation:** When aggregating Contracted Cost for savings calculations, it's important to exclude either [Charge Category](#) "Purchase" charges (one-time and recurring) that are paid to cover future eligible charges (e.g., [Commitment Discount](#)) or the covered [Charge Category](#) "Usage" charges themselves. This exclusion helps prevent double counting of these charges in the aggregation. Which set of charges to exclude depends on whether cost are aggregated on a billed basis (exclude covered charges) or accrual basis (exclude Purchases for future charges). For instance, charges categorized as [Charge Category](#) "Purchase" and their related [Charge Category](#) "Tax" charges for a Commitment Discount might be excluded from an accrual basis cost aggregation of Contracted Cost. This is because the "Usage" and "Tax" charge records provided during the term of the commitment discount already specify the Contracted Cost. Purchase charges that cover future eligible charges can be identified by filtering for [Charge Category](#) "Purchase" records with a [Billed Cost](#) greater than 0 and an [Effective Cost](#) equal to 0.

## 2.25.5. Content Constraints

Constraint	Value
Column type	Metric
Feature level	Mandatory
Allows nulls	False
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid decimal value

## 2.25.6. Introduced (version)

1.0

## 2.26. Contracted Unit Price

The Contracted Unit Price represents the agreed-upon unit price for a single [Pricing Unit](#) of the associated SKU, inclusive of [negotiated discounts](#), if present, while excluding negotiated [commitment discounts](#) or any other discounts. This price is denominated in the [Billing Currency](#). The Contracted Unit Price is commonly

used for calculating savings based on negotiation activities. If negotiated discounts are not applicable, the Contracted Unit Price defaults to the [List Unit Price](#).

The ContractedUnitPrice column MUST be present in a [FOCUS dataset](#) when the provider supports negotiated pricing concepts. This column MUST be a Decimal within the range of non-negative decimal values, MUST conform to [Numeric Format](#) requirements, and be denominated in the BillingCurrency. It MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory. When ContractedUnitPrice is present and not null, multiplying ContractedUnitPrice by [PricingQuantity](#) MUST equal [ContractedCost](#), except in cases of ChargeClass "Correction", which may address PricingQuantity or any cost discrepancies independently.

### 2.26.1. Column ID

ContractedUnitPrice

### 2.26.2. Display Name

Contracted Unit Price

### 2.26.3. Description

The agreed-upon unit price for a single Pricing Unit of the associated SKU, inclusive of negotiated discounts, if present, while excluding negotiated commitment discounts or any other discounts.

### 2.26.4. Usability Constraints

**Aggregation:** Column values should only be viewed in the context of their row and not aggregated to produce a total.

### 2.26.5. Content Constraints

Constraint	Value
Column type	Metric
Feature level	Conditional
Allows nulls	True
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid non-negative decimal value

## 2.26.6. Introduced (version)

1.0

## 2.27. Effective Cost

Effective Cost represents the [amortized](#) cost of the [charge](#) after applying all reduced rates, discounts, and the applicable portion of relevant, prepaid purchases (one-time or recurring) that covered this charge. The *amortized* portion included should be proportional to the [Pricing Quantity](#) and the time granularity of the data. Since amortization breaks down and spreads the cost of a prepaid purchase, to subsequent eligible charges, the Effective Cost of the original prepaid charge is set to 0. Effective Cost does not mix or "blend" costs across multiple charges of the same service. This cost is denominated in the [Billing Currency](#). The Effective Cost is commonly utilized to track and analyze spending trends.

This column resolves two challenges that are faced by practitioners:

1. Practitioners need to *amortize* relevant purchases, such as upfront fees, throughout the *commitment* and distribute them to the appropriate reporting groups (e.g. [tags](#), [resources](#)).
2. Many [commitment discount](#) constructs include a recurring expense for the *commitment* for every [billing period](#) and must distribute this cost to the *resources* using the *commitment*. This forces reconciliation between the initial *commitment row* per period and the actual usage *rows*.

The EffectiveCost column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column MUST be of type Decimal, MUST conform to [Numeric Format](#) requirements, and be denominated in the BillingCurrency. EffectiveCost MUST be 0 when ChargeCategory is "Purchase" and the purchase is intended to cover future eligible charges. The aggregated EffectiveCost for a billing period may not match the charge received on the invoice for the same *billing period*.

In cases where the [ChargeCategory](#) is not "Usage" or "Purchase", the following applies:

- The EffectiveCost MUST be calculated based on the EffectiveCost of the related charges if the charge is calculated based on other charges (e.g. [ChargeCategory](#) is "Tax").
- The EffectiveCost MUST match the [BilledCost](#) if the charge is unrelated to other charges (e.g. [ChargeCategory](#) is "Credit").
- When CommitmentDiscountStatus is "Unused", the EffectiveCost MUST be the total committed cost consumed for the given charge period minus related usage charges.

### 2.27.1. Column ID

EffectiveCost

### 2.27.2. Display Name

Effective Cost

### 2.27.3. Description

The *amortized* cost of the *charge* after applying all reduced rates, discounts, and the applicable portion of relevant, prepaid purchases (one-time or recurring) that covered this charge.

### 2.27.3.1. Concerning Granularity and Distribution of Recurring Fee

Providers should distribute the *commitment* purchase amount instead of including a *row* at the beginning of a period so practitioners do not need to manually distribute the fee themselves.

### 2.27.3.2. Concerning Amortization Approaches

Eligible purchases should be *amortized* using a methodology determined by the provider that reflects the needs of their customer base and is proportional to the Pricing Quantity and the time granularity of the *row*. Should a practitioner desire to *amortize* relevant purchases using a different approach, the practitioner can do so using the [Billed Cost](#) for the line item representing the initial purchase.

## 2.27.4. Content constraints

Constraint	Value
Column type	Metric
Feature level	Mandatory
Allows nulls	False
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid decimal value

### 2.27.5. Introduced (version)

0.5

## 2.28. Invoice Issuer

An Invoice Issuer is an entity responsible for invoicing for the [resources](#) or [services](#) consumed. It is commonly used for cost analysis and reporting scenarios.

The InvoiceIssuer column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST NOT contain null values.

See [Appendix: Origination of cost data](#) section for examples of [Provider](#), [Publisher](#) and Invoice Issuer values that can be used for various purchasing scenarios.

### 2.28.1. Column ID

InvoiceIssuerName

## 2.28.2. Display Name

Invoice Issuer

## 2.28.3. Description

The name of the entity responsible for invoicing for the *resources* or *services* consumed.

## 2.28.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

## 2.28.5. Introduced (version)

0.5

## 2.29. List Cost

List Cost represents the cost calculated by multiplying the [list unit price](#) and the corresponding [Pricing Quantity](#). List Cost is denominated in the [Billing Currency](#) and is commonly used for calculating savings based on various rate optimization activities, by comparing it with [Contracted Cost](#), [Billed Cost](#) and [Effective Cost](#).

The ListCost column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column MUST be of type Decimal, MUST conform to [Numeric Format](#) requirements, and be denominated in the BillingCurrency. When [ListUnitPrice](#) is present and not null, multiplying the ListUnitPrice by PricingQuantity MUST produce the ListCost, except in cases of [ChargeClass](#) "Correction", which may address PricingQuantity or any cost discrepancies independently.

In cases where the ListUnitPrice is present and is null, the following applies:

- The ListCost of a charge calculated based on other charges (e.g., when the [ChargeCategory](#) is "Tax") MUST be calculated based on the ListCost of those related charges.
- The ListCost of a charge unrelated to other charges (e.g., when the [ChargeCategory](#) is "Credit") MUST match the [BilledCost](#).

### 2.29.1. Column ID

ListCost

## 2.29.2. Display Name

List Cost

## 2.29.3. Description

Cost calculated by multiplying List Unit Price and the corresponding Pricing Quantity.

## 2.29.4. Usability Constraints

**Aggregation:** When aggregating List Cost for savings calculations, it's important to exclude either [Charge Category](#) "Purchase" charges (one-time and recurring) that are paid to cover future eligible charges (e.g., [Commitment Discount](#)) or the covered [Charge Category](#) "Usage" charges themselves. This exclusion helps prevent double counting of these charges in the aggregation. Which set of charges to exclude depends on whether cost are aggregated on a billed basis (exclude covered charges) or accrual basis (exclude Purchases for future charges). For instance, charges categorized as [Charge Category](#) "Purchase" and their related [Charge Category](#) "Tax" charges for a Commitment Discount might be excluded from an accrual basis cost aggregation of List Cost. This is because the "Usage" and "Tax" charge records provided during the term of the commitment discount already specify the List Cost. Purchase charges that cover future eligible charges can be identified by filtering for [Charge Category](#) "Purchase" records with a [Billed Cost](#) greater than 0 and an [Effective Cost](#) equal to 0.

## 2.29.5. Content Constraints

Constraint	Value
Column type	Metric
Feature level	Mandatory
Allows nulls	False
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid decimal value

## 2.29.6. Introduced (version)

1.0-preview

## 2.30. List Unit Price

The List Unit Price represents the suggested provider-published unit price for a single [Pricing Unit](#) of the associated SKU, exclusive of any discounts. This price is denominated in the [Billing Currency](#). The List Unit Price is commonly used for calculating savings based on various rate optimization activities.



The ListUnitPrice column MUST be present in a [FOCUS dataset](#) when the provider publishes unit prices exclusive of discounts. This column MUST be a Decimal within the range of non-negative decimal values, MUST conform to [Numeric Format](#) requirements, and be denominated in the BillingCurrency. It MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory. When ListUnitPrice is present and is not null, multiplying ListUnitPrice by [PricingQuantity](#) MUST equal [ListCost](#), except in cases of ChargeClass "Correction", which may address PricingQuantity or any cost discrepancies independently.

### 2.30.1. Column ID

ListUnitPrice

### 2.30.2. Display Name

List Unit Price

### 2.30.3. Description

The suggested provider-published unit price for a single Pricing Unit of the associated SKU, exclusive of any discounts.

### 2.30.4. Usability Constraints

**Aggregation:** Column values should only be viewed in the context of their row and not aggregated to produce a total.

### 2.30.5. Content Constraints

Constraint	Value
Column type	Metric
Feature level	Conditional
Allows nulls	True
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number range	Any valid non-negative decimal value

### 2.30.6. Introduced (version)

## 2.31. Pricing Category

Pricing Category describes the pricing model used for a charge at the time of use or purchase. It can be useful for distinguishing between charges incurred at the [list unit price](#) or a reduced price and exposing optimization opportunities, like increasing [commitment discount](#) coverage.

The PricingCategory column adheres to the following requirements:

- PricingCategory MUST be present in a [FOCUS dataset](#) when the provider supports more than one pricing category across all SKUs and MUST be of type String.
- PricingCategory MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory.
- PricingCategory MUST be one of the allowed values.
- PricingCategory MUST be "Standard" when pricing is predetermined at the agreed upon rate for the [billing account](#).
- PricingCategory MUST be "Committed" when the charge is subject to an existing *commitment discount* and is not the purchase of the *commitment discount*.
- PricingCategory MUST be "Dynamic" when pricing is determined by the provider and may change over time, regardless of predetermined agreement pricing.
- PricingCategory MUST be "Other" when there is a pricing model but none of the allowed values apply.

### 2.31.1. Column ID

PricingCategory

### 2.31.2. Display Name

Pricing Category

### 2.31.3. Description

Describes the pricing model used for a charge at the time of use or purchase.

### 2.31.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	Allowed values

Allowed values:

Value	Description
Standard	Charges priced at the agreed upon rate for the billing account, including <a href="#">negotiated discounts</a> . This pricing includes any flat rate and volume/tiered pricing but does not include dynamic pricing or reduced pricing due to the application of a <i>commitment discount</i> . This does include the purchase of a commitment discount at agreed upon rates.
Dynamic	Charges priced at a variable rate determined by the provider. This includes any product or service with a unit price the provider can change without notice, like interruptible or low priority resources.
Committed	Charges with reduced pricing due to the application of the <i>commitment discount</i> specified by the Commitment Discount ID.
Other	Charges priced in a way not covered by another pricing category.

### 2.31.5. Introduced (version)

1.0-preview

## 2.32. Pricing Quantity

The Pricing Quantity represents the volume of a given SKU associated with a [resource](#) or [service](#) used or purchased, based on the [Pricing Unit](#). Distinct from [Consumed Quantity](#) (complementary to [Consumed Unit](#)), it focuses on pricing and cost, not *resource* and *service* consumption.

The PricingQuantity column MUST be present in a [FOCUS dataset](#). This column MUST be of type Decimal and MUST conform to [Numeric Format](#) requirements. The value MAY be negative in cases where [ChargeClass](#) is "Correction". This column MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory. When unit prices are not null, multiplying PricingQuantity by a unit price MUST produce a result equal to the corresponding cost metric, except in cases of ChargeClass "Correction", which may address PricingQuantity or any cost discrepancies independently.

### 2.32.1. Column ID

PricingQuantity

### 2.32.2. Display Name

Pricing Quantity

### 2.32.3. Description

The volume of a given SKU associated with a *resource* or *service* used or purchased, based on the Pricing Unit.

#### 2.32.4. Usability Constraints

**Aggregation:** When aggregating Pricing Quantity for commitment utilization calculations, it's important to exclude *commitment discount* purchases (i.e. when *ChargeCategory* is "Purchase") that are paid to cover future eligible charges (e.g., *Commitment Discount*). Otherwise, when accounting for all upfront or accrued purchases, it's important to exclude *commitment discount* usage (i.e. when *ChargeCategory* is "Usage"). This exclusion helps prevent double counting of these quantities in the aggregation.

#### 2.32.5. Content Constraints

Constraint	Value
Column type	Metric
Feature level	Mandatory
Allows nulls	True
Data type	Decimal
Value format	<a href="#">Numeric Format</a>
Number Range	Any valid decimal value

#### 2.32.6. Introduced (version)

1.0-preview

### 2.33. Pricing Unit

The Pricing Unit represents a provider-specified measurement unit for determining unit prices, indicating how the provider rates measured usage and purchase quantities after applying pricing rules like [block pricing](#). Common examples include the number of hours for compute appliance runtime (e.g. Hours), gigabyte-hours for a storage appliance (e.g., GB-Hours), or an accumulated count of requests for a network appliance or API service (e.g., 1000 Requests). Pricing Unit complements the [Pricing Quantity](#) metric. Distinct from the [Consumed Unit](#), it focuses on pricing and cost, not [resource](#) and [service](#) consumption, often at a coarser granularity.

The PricingUnit column MUST be present in a [FOCUS dataset](#). This column MUST be of type String. It MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory. Units of measure used in PricingUnit SHOULD adhere to the values and format requirements specified in the [UnitFormat](#) attribute.

The PricingUnit value MUST be semantically equal to the corresponding pricing measurement unit value provided in:

- The provider-published [price list](#)
- The invoice, when the invoice includes a pricing measurement unit

### 2.33.1. Column ID

PricingUnit

### 2.33.2. Display Name

Pricing Unit

### 2.33.3. Description

Provider-specified measurement unit for determining unit prices, indicating how the provider rates measured usage and purchase quantities after applying pricing rules like *block pricing*.

### 2.33.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	True
Data type	String
Value format	<a href="#">Unit</a> <a href="#">Format</a>

### 2.33.5. Introduced (version)

1.0-preview

## 2.34. Provider

A Provider is an entity that makes the [resources](#) or [services](#) available for purchase. It is commonly used for cost analysis and reporting scenarios.

The Provider column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST NOT contain null values.

See [Appendix: Origination of cost data](#) section for examples of Provider, Publisher and Invoice Issuer values that can be used for various purchasing scenarios.

### 2.34.1. Column ID

ProviderName

### 2.34.2. Display Name

Provider

### 2.34.3. Description

The name of the entity that made the *resources* or *services* available for purchase.

### 2.34.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

### 2.34.5. Introduced (version)

0.5

## 2.35. Publisher

A Publisher is an entity that produces the [resources](#) or [services](#) that were purchased. It is commonly used for cost analysis and reporting scenarios.

The Publisher column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST NOT contain null values.

See [Appendix: Origination of cost data](#) section for examples of [Provider](#), Publisher and [Invoice Issuer](#) values that can be used for various purchasing scenarios.

### 2.35.1. Column ID

PublisherName

### 2.35.2. Display Name

Publisher

### 2.35.3. Description

The name of the entity that produced the *resources* or *services* that were purchased.

#### 2.35.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

#### 2.35.5. Introduced (version)

0.5

### 2.36. Region ID

A Region ID is a provider-assigned identifier for an isolated geographic area where a [resource](#) is provisioned or a [service](#) is provided. The region is commonly used for scenarios like analyzing cost and unit prices based on where *resources* are deployed.

The RegionId column MUST be present in a [FOCUS dataset](#) when the provider supports deploying resources or services within a *region* and MUST be of type String. RegionId MUST NOT be null when a *resource* or *service* is operated in or managed from a distinct region by the Provider and MAY contain null values when a *resource* or *service* is not restricted to an isolated geographic area.

#### 2.36.1. Column ID

RegionId

#### 2.36.2. Display Name

Region ID

#### 2.36.3. Description

Provider-assigned identifier for an isolated geographic area where a *resource* is provisioned or a *service* is provided.

#### 2.36.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.36.5. Introduced (version)

1.0

## 2.37. Region Name

Region Name is a provider-assigned display name for an isolated geographic area where a [resource](#) is provisioned or a [service](#) is provided. Region Name is commonly used for scenarios like analyzing cost and unit prices based on where *resources* are deployed.

The RegionName column MUST be present in a [FOCUS dataset](#) when the provider supports deploying resources or services within a *region* and MUST be of type String. RegionName MUST NOT be null when a *resource* or *service* is operated in or managed from a distinct region by the Provider and MAY contain null values when a *resource* or *service* is not restricted to an isolated geographic area.

### 2.37.1. Column ID

RegionName

### 2.37.2. Display Name

Region Name

### 2.37.3. Description

The name of an isolated geographic area where a *resource* is provisioned or a *service* is provided.

### 2.37.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True



Constraint	Value
Data type	String
Value format	<not specified>

### 2.37.5. Introduced (version)

1.0

## 2.38. Resource ID

A Resource ID is an identifier assigned to a [resource](#) by the provider. The Resource ID is commonly used for cost reporting, analysis, and allocation scenarios.

The ResourceId column MUST be present in a [FOCUS dataset](#) when the provider supports billing based on provisioned resources. This column MUST be of type String. The ResourceId value MAY be a nullable column as some cost data [rows](#) may not be associated with a *resource*. ResourceId MUST appear in the cost data if an identifier is assigned to a *resource* by the provider. ResourceId SHOULD be a fully-qualified identifier that ensures global uniqueness within the provider.

### 2.38.1. Column ID

ResourceId

### 2.38.2. Display Name

Resource ID

### 2.38.3. Description

Identifier assigned to a *resource* by the provider.

### 2.38.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

## 2.38.5. Introduced (version)

0.5

## 2.39. Resource Name

The Resource Name is a display name assigned to a [resource](#). It is commonly used for cost analysis, reporting, and allocation scenarios.

The ResourceName column MUST be present in a [FOCUS dataset](#) when the provider supports billing based on provisioned resources. This column MUST be of type String. The ResourceName value MAY be a nullable column as some cost data [rows](#) may not be associated with a *resource* or because a display name cannot be assigned to a *resource*. ResourceName MUST NOT be null if a display name can be assigned to a *resource*. *Resources* not provisioned interactively or only have a system-generated [ResourceId](#) MUST NOT duplicate the same value as the ResourceName.

### 2.39.1. Column ID

ResourceName

### 2.39.2. Display Name

Resource Name

### 2.39.3. Description

Display name assigned to a *resource*.

### 2.39.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

## 2.39.5. Introduced (version)

0.5

## 2.40. Resource Type

Resource Type describes the kind of [resource](#) the charge applies to. A Resource Type is commonly used for scenarios like identifying cost changes in groups of similar *resources* and may include values like Virtual Machine, Data Warehouse, and Load Balancer.

The ResourceType column MUST be present in a [FOCUS dataset](#) when the provider supports billing based on provisioned resources and supports assigning a type for resources. This column MUST be of type String and MUST NOT be null when a corresponding [ResourceId](#) is not null. When a corresponding ResourceId value is null, the ResourceType column value MUST also be null.

### 2.40.1. Column ID

ResourceType

### 2.40.2. Display Name

Resource Type

### 2.40.3. Description

The kind of *resource* the charge applies to.

### 2.40.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.40.5. Introduced (version)

1.0-preview

## 2.41. Service Category

The Service Category is the highest-level classification of a [service](#) based on the core function of the *service*. Each *service* should have one and only one category that best aligns with its primary purpose. The Service Category is commonly used for scenarios like analyzing costs across providers and tracking the migration of workloads across fundamentally different architectures.

The ServiceCategory column MUST be present in a [FOCUS dataset](#) and MUST NOT be null. This column is of type String and MUST be one of the allowed values.

### 2.41.1. Column ID

ServiceCategory

### 2.41.2. Display Name

Service Category

### 2.41.3. Description

Highest-level classification of a *service* based on the core function of the *service*.

### 2.41.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	Allowed Values

Allowed values:

Service Category	Description
AI and Machine Learning	Artificial Intelligence and Machine Learning related technologies.
Analytics	Data processing, analytics, and visualization capabilities.
Business Applications	Business and productivity applications and services.
Compute	Virtual, containerized, serverless, or high-performance computing infrastructure and services.
Databases	Database platforms and services that allow for storage and querying of data.
Developer Tools	Software development and delivery tools and services.
Multicloud	Support for interworking of multiple cloud and/or on-premises environments.
Identity	Identity and access management services.
Integration	Services that allow applications to interact with one another.
Internet of Things	Development and management of IoT devices and networks.

Service Category	Description
Management and Governance	Management, logging, and observability of a customer's use of cloud.
Media	Media and entertainment streaming and processing services.
Migration	Moving applications and data to the cloud.
Mobile	Services enabling cloud applications to interact via mobile technologies.
Networking	Network connectivity and management.
Security	Security monitoring and compliance services.
Storage	Storage services for structured or unstructured data.
Web	Services enabling cloud applications to interact via the Internet.
Other	New or emerging services that do not align with an existing category.

### 2.41.5. Introduced (version)

0.5

## 2.42. Service Name

A [service](#) represents an offering that can be purchased from a provider (e.g., cloud virtual machine, SaaS database, professional services from a systems integrator). A *service* offering can include various types of usage or other charges. For example, a cloud database *service* may include compute, storage, and networking charges.

The Service Name is a display name for the offering that was purchased. The Service Name is commonly used for scenarios like analyzing aggregate cost trends over time and filtering data to investigate anomalies.

The ServiceName column MUST be present in a [FOCUS dataset](#). This column MUST be of type String and MUST NOT contain null values.

### 2.42.1. Column ID

ServiceName

### 2.42.2. Display Name

Service Name

### 2.42.3. Description

An offering that can be purchased from a provider (e.g., cloud virtual machine, SaaS database, professional *services* from a systems integrator).

#### 2.42.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

#### 2.42.5. Introduced (version)

0.5

### 2.43. Service Subcategory

The Service Subcategory is a secondary classification of the [Service Category](#) for a [service](#) based on its core function. The Service Subcategory (in conjunction with the Service Category) is commonly used for scenarios like analyzing spend and usage for specific workload types across providers and tracking the migration of workloads across fundamentally different architectures.

The ServiceSubcategory column adheres to the following requirements:

- ServiceSubcategory is RECOMMENDED to be present in a [FOCUS dataset](#) and MUST NOT be null.
- ServiceSubcategory is of type String and MUST be one of the allowed values.
- Each ServiceSubcategory value MUST have one and only one parent ServiceCategory as specified in the allowed values below.
- Though a given *service* can have multiple purposes, each *service* SHOULD have one and only one ServiceSubcategory that best aligns with its primary purpose.

#### 2.43.1. Column ID

ServiceSubcategory

#### 2.43.2. Display Name

Service Subcategory

#### 2.43.3. Description

Secondary classification of the Service Category for a *service* based on its core function.

#### 2.43.4. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Recommended
Allows nulls	False
Data type	String
Value format	Allowed Values

Allowed values:

Service Category	Service Subcategory	Service Subcategory Description
AI and Machine Learning	AI Platforms	Unified solution that combines artificial intelligence and machine learning technologies.
AI and Machine Learning	Bots	Automated performance of tasks such as customer service, data collection, and content moderation.
AI and Machine Learning	Generative AI	Creation of content like text, images, and music by learning patterns from existing data.
AI and Machine Learning	Machine Learning	Creation, training, and deployment of statistical algorithms that learn from and perform tasks based on data.
AI and Machine Learning	Natural Language Processing	Generation of human language, handling tasks like translation, sentiment analysis, and text summarization.
AI and Machine Learning	Other (AI and Machine Learning)	AI and Machine Learning services that do not fall into one of the defined subcategories.
Analytics	Analytics Platforms	Unified solution that combines technologies across the entire analytics lifecycle.
Analytics	Business Intelligence	Semantic models, dashboards, reports, and data visualizations to track performance and identify trends.
Analytics	Data Processing	Integration and transformation tasks to prepare data for analysis.
Analytics	Search	Discovery of information by indexing and retrieving data from various sources.
Analytics	Streaming Analytics	Real-time data stream processes to detect patterns, trends, and anomalies as they occur.
Analytics	Other (Analytics)	Analytics services that do not fall into one of the defined subcategories.
Business Applications	Productivity and Collaboration	Tools that facilitate individuals managing tasks and working together.
Business Applications	Other (Business Applications)	Business Applications services that do not fall into one of the defined subcategories.
Compute	Containers	Management and orchestration of containerized compute platforms.
Compute	End User Computing	Virtualized desktop infrastructure and device / endpoint management.
Compute	Quantum Compute	Resources and simulators that leverage the principles of quantum mechanics.
Compute	Serverless Compute	Enablement of compute capabilities without provisioning or managing servers.

Service Category	Service Subcategory	Service Subcategory Description
Compute	Virtual Machines	Computing environments ranging from hosts with abstracted operating systems to bare-metal servers.
Compute	Other (Compute)	Compute services that do not fall into one of the defined subcategories.
Databases	Caching	Low-latency and high-throughput access to frequently accessed data.
Databases	Data Warehouses	Big data storage and querying capabilities.
Databases	Ledger Databases	Immutable and transparent databases to record tamper-proof and cryptographically secure transactions.
Databases	NoSQL Databases	Unstructured or semi-structured data storage and querying capabilities.
Databases	Relational Databases	Structured data storage and querying capabilities.
Databases	Time Series Databases	Time-stamped data storage and querying capabilities.
Databases	Other (Databases)	Database services that do not fall into one of the defined subcategories.
Developer Tools	Developer Platforms	Unified solution that combines technologies across multiple areas of the software development lifecycle.
Developer Tools	Continuous Integration and Deployment	CI/CD tools and services that support building and deploying code for software and systems.
Developer Tools	Development Environments	Tools and services that support authoring code for software and systems.
Developer Tools	Source Code Management	Tools and services that support version control of code for software and systems.
Developer Tools	Quality Assurance	Tools and services that support testing code for software and systems.
Developer Tools	Other (Developer Tools)	Developer Tools services that do not fall into one of the defined subcategories.
Identity	Identity and Access Management	Technologies that ensure users have appropriate access to resources.
Identity	Other (Identity)	Identity services that do not fall into one of the defined subcategories.
Integration	API Management	Creation, publishing, and management of application programming interfaces.
Integration	Messaging	Asynchronous communication between distributed applications.
Integration	Workflow Orchestration	Design, execution, and management of business processes and workflows.
Integration	Other (Integration)	Integration services that do not fall into one of the defined subcategories.
Internet of Things	IoT Analytics	Examination of data collected from IoT devices.
Internet of Things	IoT Platforms	Unified solution that combines IoT data collection, processing, visualization, and device management.



Service Category	Service Subcategory	Service Subcategory Description
Internet of Things	Other (Internet of Things)	Internet of Things (IoT) services that do not fall into one of the defined subcategories.
Management and Governance	Architecture	Planning, design, and construction of software systems.
Management and Governance	Compliance	Adherence to regulatory standards and industry best practices.
Management and Governance	Cost Management	Monitoring and controlling expenses of systems and services.
Management and Governance	Data Governance	Management of the availability, usability, integrity, and security of data.
Management and Governance	Disaster Recovery	Plans and procedures that ensure systems and services can recover from disruptions.
Management and Governance	Endpoint Management	Tools that configure and secure access to devices.
Management and Governance	Observability	Monitoring, logging, and tracing of data to track the performance and health of systems.
Management and Governance	Support	Assistance and expertise supplied by providers.
Management and Governance	Other (Management and Governance)	Management and governance services that do not fall into one of the defined subcategories.
Media	Content Creation	Production of media content.
Media	Gaming	Development and delivery of gaming services.
Media	Media Streaming	Multimedia delivered and rendered in real-time on devices.
Media	Mixed Reality	Technologies that blend real-world and computer-generated environments.
Media	Other (Media)	Media services that do not fall into one of the defined subcategories.
Migration	Data Migration	Movement of stored data from one location to another.
Migration	Resource Migration	Movement of resources from one location to another.
Migration	Other (Migration)	Migration services that do not fall into one of the defined subcategories.
Mobile	Other (Mobile)	All Mobile services.
Multicloud	Multicloud Integration	Environments that facilitate consumption of services from multiple cloud providers.
Multicloud	Other (Multicloud)	Multicloud services that do not fall into one of the defined subcategories.
Networking	Application Networking	Distribution of incoming network traffic across application-based workloads.

Service Category	Service Subcategory	Service Subcategory Description
Networking	Content Delivery	Distribution of digital content using a network of servers (CDNs).
Networking	Network Connectivity	Facilitates communication between networks or network segments.
Networking	Network Infrastructure	Configuration, monitoring, and troubleshooting of network devices.
Networking	Network Routing	Services that select paths for traffic within or across networks.
Networking	Network Security	Protection from unauthorized network access and cyber threats using firewalls and anti-malware tools.
Networking	Other (Networking)	Networking services that do not fall into one of the defined subcategories.
Security	Secret Management	Information used to authenticate users and systems, including secrets, certificates, tokens, and other keys.
Security	Security Posture Management	Tools that help organizations configure, monitor, and improve system security.
Security	Threat Detection and Response	Collect and analyze security data to identify and respond to potential security threats and vulnerabilities.
Security	Other (Security)	Security services that do not fall into one of the defined subcategories.
Storage	Backup Storage	Secondary storage to protect against data loss.
Storage	Block Storage	High performance, low latency storage that provides random access.
Storage	File Storage	Scalable, sharable storage for file-based data.
Storage	Object Storage	Highly available, durable storage for unstructured data.
Storage	Storage Platforms	Unified solution that supports multiple storage types.
Storage	Other (Storage)	Storage services that do not fall into one of the defined subcategories.
Web	Application Platforms	Integrated environments that run web applications.
Web	Other (Web)	Web services that do not fall into one of the defined subcategories.
Other	Other (Other)	Services that do not fall into one of the defined categories.

### 2.43.5. Introduced (version)

1.1

## 2.44. SKU ID

A SKU ID is a unique identifier that defines a provider-supported construct for organizing properties that are common across one or more [SKU Prices](#). SKU ID can be referenced on a catalog or [price list](#) published by a provider to look up detailed information about the SKU. The composition of the properties associated with the SKU ID may differ across providers. Some providers may not support the [SKU](#) construct and instead associate all such properties directly with the *SKU Price*. SKU ID is commonly used for analyzing cost based

on *SKU*-related properties above the pricing constructs.

The Skuld column **MUST** be present in a [FOCUS dataset](#) when the provider publishes a SKU list. This column **MUST** be of type String. It **MUST NOT** be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", **MUST** be null when ChargeCategory is "Tax", and **MAY** be null for all other combinations of ChargeClass and ChargeCategory. Skuld **MUST** equal SkuPriceld when a provider does not support an overarching SKU ID construct.

#### 2.44.1. Column ID

Skuld

#### 2.44.2. Display Name

SKU ID

#### 2.44.3. Description

A unique identifier that defines a provider-supported construct for organizing properties that are common across one or more *SKU Prices*.

#### 2.44.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

#### 2.44.5. Introduced (version)

1.0-preview

### 2.45. SKU Meter

The SKU Meter describes the functionality being metered or measured by a particular SKU in a charge.

Providers often have billing models in which multiple SKUs exist for a given service to describe and bill for different functionalities for that service. For example, an object storage service may have separate SKUs for functionalities such as object storage, API requests, data transfer, encryption, and object management. This field helps practitioners understand which functionalities are being metered by the different SKUs that appear in a [FOCUS dataset](#).

The SkuMeter column adheres to the following requirements:

- SkuMeter MUST be present in a *FOCUS dataset* when the provider includes a [Skuld](#).
- SkuMeter MUST be of type String.
- SkuMeter MUST be null when Skuld is null.
- SkuMeter SHOULD NOT be null when Skuld is not null.
- SkuMeter SHOULD remain consistent over time for a given Skuld.

### 2.45.1. Examples

Compute Usage, Block Volume Usage, Data Transfer, API Requests

### 2.45.2. Column ID

SkuMeter

### 2.45.3. Display Name

SKU Meter

### 2.45.4. Description

Describes the functionality being metered or measured by a particular SKU in a charge.

### 2.45.5. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.45.6. Introduced (version)

1.1

## 2.46. SKU Price Details

The SKU Price Details column represents a list of relevant properties shared by all charges with the same

[SKU Price ID](#). These properties provide qualitative and quantitative details about the service represented by a SKU Price ID. This can enable practitioners to calculate metrics such as total units of a service when it is not directly billed in those units (e.g. cores) and thus enables FinOps capabilities such as unit economics. These properties can also help a practitioner understand the specifics of a SKU Price ID and differentiate it from other SKU Price IDs.

The SkuPriceDetails column adheres to the following requirements:

- The SkuPriceDetails column MUST be in [KeyValueFormat](#).
- The key for a property SHOULD be formatted in [PascalCase](#).
- The properties (both keys and values) contained in the SkuPriceDetails column MUST be shared across all charges having the same SkuPricId, subject to the below provisions.
  - Additional properties (key-value pairs) MAY be added to SkuPriceDetails going forward for a given SkuPricId.
  - Properties SHOULD NOT be removed from SkuPriceDetails for a given SkuPricId, once they have been included.
  - Individual properties (key-value pairs) SHOULD NOT be modified for a given SkuPricId and SHOULD remain consistent over time.
- The key for a property SHOULD remain consistent across comparable SKUs having that property and the values for this key SHOULD remain in a consistent format.
- The SkuPriceDetails column MUST NOT contain properties which are not applicable to the corresponding SkuPricId.
- The SkuPriceDetails column MAY contain properties which are already captured in other dedicated columns.
- If a property has a numeric value, it MUST represent the value for a single [PricingUnit](#).
- The SkuPriceDetails column MUST be present in a [FOCUS dataset](#) when the provider includes a SkuPricId.
  - The SkuPriceDetails column MAY be null when SkuPricId is not null.
  - The SkuPriceDetails column MUST be null when SkuPricId is null.

### 2.46.1. Examples

```
{  
  "OperationClass": "A",  
  "PricingTier": 2,  
  "CoreHours": 4,  
  "PremiumProcessing": true,  
}
```

### 2.46.2. Column ID

SkuPriceDetails

### 2.46.3. Display Name

SKU Price Details

## 2.46.4. Description

A set of properties of a SKU Price ID which are meaningful and common to all instances of that SKU Price ID.

## 2.46.5. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	JSON
Value format	<a href="#">Key-Value Format</a>

## 2.46.6. Introduced (version)

1.1

## 2.47. SKU Price ID

A SKU Price ID is a unique identifier that defines the unit price used to calculate the charge. SKU Price ID can be referenced on a [price list](#) published by a provider to look up detailed information, including a corresponding list unit price. The composition of the properties associated with the SKU Price ID may differ across providers. SKU Price ID is commonly used for analyzing cost based on pricing properties such as Terms and Tiers.

The SkuPriceld column adheres to the following requirements:

- SkuPriceld MUST be present in a [FOCUS dataset](#) when the provider publishes a SKU price list and MUST be of type String.
- SkuPriceld MUST define a single unit price used for calculating the charge.
- [ListUnitPrice](#) MUST be associated with the SkuPriceld in the provider published price list.
- SkuPriceld MUST NOT be null when [ChargeClass](#) is not "Correction" and [ChargeCategory](#) is "Usage" or "Purchase", MUST be null when ChargeCategory is "Tax", and MAY be null for all other combinations of ChargeClass and ChargeCategory.
- A given value of SkuPriceld MUST be associated with one and only one [Skuld](#), except in cases of [commitment discount flexibility](#).
- If a provider does not have a SkuPriceld and wants to include information in columns linked to SkuPriceld such as ListUnitPrice or [SkuPriceDetails](#), the Skuld MAY be used in the SkuPriceld column as long as it adheres to the above conditions.

### 2.47.1. Column ID

SkuPriceld

### 2.47.2. Display Name

SKU Price ID

### 2.47.3. Description

A unique identifier that defines the unit price used to calculate the charge.

### 2.47.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.47.5. Introduced (version)

1.0-preview

## 2.48. Sub Account ID

A Sub Account ID is a provider-assigned identifier assigned to a [sub account](#). Sub Account ID is commonly used for scenarios like grouping based on organizational constructs, access management needs, and cost allocation strategies.

The SubAccountId column MUST be present in a [FOCUS dataset](#) when the provider supports a *sub account* construct. This column MUST be of type String. If a charge does not apply to a *sub account*, the SubAccountId column MUST be null.

See [Appendix: Grouping constructs for resources or services](#) for details and examples of the different grouping constructs supported by FOCUS.

### 2.48.1. Column ID

SubAccountId

### 2.48.2. Display Name

Sub Account ID

### 2.48.3. Description

An ID assigned to a grouping of [resources](#) or [services](#), often used to manage access and/or cost.

### 2.48.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

### 2.48.5. Introduced (version)

0.5

## 2.49. Sub Account Name

A Sub Account Name is a display name assigned to a [sub account](#). Sub account Name is commonly used for scenarios like grouping based on organizational constructs, access management needs, and cost allocation strategies.

The SubAccountName column MUST be present in a [FOCUS dataset](#) when the provider supports a *sub account* construct. This column MUST be of type String. If a charge does not apply to a *sub account*, the SubAccountName column MUST be null.

See [Appendix: Grouping constructs for resources or services](#) for details and examples of the different grouping constructs supported by FOCUS.

### 2.49.1. Column ID

SubAccountName

### 2.49.2. Display Name

Sub Account Name

### 2.49.3. Description

A name assigned to a grouping of [resources](#) or [services](#), often used to manage access and/or cost.



#### 2.49.4. Content constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	String
Value format	<not specified>

#### 2.49.5. Introduced (version)

0.5

### 2.50. Tags

The Tags column represents the set of tags assigned to [tag sources](#) that also account for potential provider-defined or user-defined tag evaluations. Tags are commonly used for scenarios like adding business context to cost and usage data to identify and accurately allocate charges. Tags may also be referred to by providers using other terms such as labels.

A tag becomes [finalized](#) when a single value is selected from a set of possible tag values assigned to the tag key. When supported by a provider, this can occur when a tag value is set by provider-defined or user-defined rules.

The Tags column adheres to the following requirements:

- The Tags column **MUST** be present in a [FOCUS dataset](#) when the provider supports setting user or provider-defined tags.
- The Tags column **MUST** contain user-defined and provider-defined tags.
- The Tags column **MUST** only contain finalized tags.
- The Tags column **MUST** be in [KeyValueFormat](#).
- A Tag key with a non-null value for a given resource **SHOULD** be included in the tags column.
- A Tag key with a null value for a given resource **MAY** be included in the tags column depending on the provider's tag finalization process.
- A Tag key that does *not* support a corresponding value, **MUST** have a corresponding true (boolean) value set.
- If Tag finalization is supported, providers **MUST** publish tag finalization methods and semantics within their respective documentation.
- Providers **MUST NOT** alter user-defined Tag keys or values.

Provider-defined Tags additionally adhere to the following requirements:

- Provider-defined tags **MUST** be prefixed with a provider-specified tag key prefix.
- Providers **SHOULD** publish all provider-specified tag key prefixes within their respective documentation.

#### 2.50.1. Provider-Defined vs. User-Defined Tags

This example illustrates three different tagging scenarios. The first two illustrate when the provider supports both keys and values, while the third is for supporting keys only. The first tag is user-defined and doesn't

have a provider prefix. The second tag is provider-defined and has a prefix of *acme/*, which is reserved by the provider. The third tag has a tag key of *baz* and its value is assigned the boolean value *true* since the tag doesn't support a value.

```
{
  "foo": "bar",
  "acme/foo": "bar",
  "baz": true,
}
```

### 2.50.2. Finalized Tags

Within a provider, tag keys may be associated with multiple values, and potentially defined at different levels within the provider, such as accounts, folders, [resource](#) and other *resource* grouping constructs. When finalizing, *providers* must reduce these multiple levels of definition to a single value where each key is associated with exactly one value. The method by which this is done and the semantics are up to each provider but must be documented within their respective documentation.

As an example, let's assume 1 [sub account](#) exists with 1 virtual machine with the following details, and tag inheritance favors Resources over *Sub Accounts*.

- Sub Account
  - id: *my-sub-account*
  - user-defined tags: *team:ops*, *env:prod*
- Virtual Machine
  - id: *my-vm*
  - user-defined tags: *team:web*

The table below represents a finalized dataset with these *resources*. It also shows the finalized state after all resource-oriented, tag inheritance rules are processed.

ResourceType	ResourceId	Tags
Sub Account	my-sub-account	{ "team": "ops", "env": "prod" }
Virtual Machine	my-vm	{ "team": "web", "env": "prod" }

Because the the Virtual Machine Resource did not have an *env* tag, it inherited tag, *env:prod* (italicized), from its parent *sub account*. Conversely, because the Virtual Machine Resource already has a *team* tag (*team:web*), it did not inherit *team:ops* from its parent *sub account*.

### 2.50.3. Column ID

Tags

### 2.50.4. Display Name

Tags

### 2.50.5. Description

The set of tags assigned to *tag sources* that account for potential provider-defined or user-defined tag evaluations.

### 2.50.6. Content Constraints

Constraint	Value
Column type	Dimension
Feature level	Conditional
Allows nulls	True
Data type	JSON
Value format	<a href="#">Key-Value Format</a>

### 2.50.7. Introduced (version)

1.0-preview

## 3. Attributes

Attributes are requirements that apply across a [FOCUS dataset](#) instead of an individual column level. Requirements on data content can include naming conventions, data types, formatting standardizations, etc. Attributes may introduce high-level requirements for data granularity, recency, frequency, etc. Requirements defined in attributes are necessary for servicing [FinOps capabilities](#) accurately using a standard set of instructions regardless of the origin of the data.

### 3.1. Column Naming and Ordering

Column IDs provided in cost data following a consistent naming and ordering convention reduce friction for FinOps practitioners who consume the data for analysis, reporting, and other use cases.

All columns defined in the FOCUS specification MUST follow the naming and ordering requirements listed below.

#### 3.1.1. Attribute ID

ColumnNamingAndOrdering

#### 3.1.2. Attribute Name

Column Naming and Ordering

### 3.1.3. Description

Naming and ordering convention for columns appearing in a [FOCUS dataset](#).

### 3.1.4. Requirements

#### 3.1.4.1. Column Names

- All columns defined by FOCUS MUST follow the following rules:
  - Column IDs MUST use [Pascal case](#).
  - Column IDs MUST NOT use abbreviations.
  - Column IDs MUST be alphanumeric with no special characters.
  - Columns that have an ID and a Name MUST have the Id or Name suffix in the Column ID. Display Name for a Column MAY avoid the Name suffix if there are no other columns with the same name prefix.
  - Column IDs SHOULD NOT use acronyms.
  - Column IDs SHOULD NOT exceed 50 characters to accommodate column length restrictions of various data repositories.
- All custom columns MUST be prefixed with a consistent x\_ prefix to identify them as external, custom columns and distinguish them from FOCUS columns to avoid conflicts in future releases.
- Columns that have an ID and a Name MUST have the Id or Name suffix in the Column ID. Display Name for a Column MAY avoid the Name suffix if it is considered superfluous.
- Columns with the Category suffix MUST be normalized.
- Custom (e.g., provider-defined) columns SHOULD follow the same rules listed above for FOCUS columns.

#### 3.1.4.2. Column Order

- All FOCUS columns SHOULD be first in the provided dataset.
- Custom columns SHOULD be listed after all FOCUS columns and SHOULD NOT be intermixed.
- Columns MAY be sorted alphabetically, but custom columns SHOULD be after all FOCUS columns.

### 3.1.5. Exceptions

- Identifiers will use the "Id" abbreviation since this is a standard pattern across the industry.
- Product offerings that incur charges will use the "Sku" abbreviation because it is a well-understood term both within and outside the industry.

### 3.1.6. Introduced (version)

0.5

## 3.2. Currency Code Format

Columns that contain currency information in cost data following a consistent format reduce friction for FinOps practitioners who consume the data for analysis, reporting, and other use cases.

All columns capturing a currency value, defined in the FOCUS specification, MUST follow the requirements listed below. Custom currency-related columns SHOULD also follow the same formatting requirements.

### 3.2.1. Attribute ID

CurrencyCodeFormat

### 3.2.2. Attribute Name

Currency Code Format

### 3.2.3. Description

Formatting for currency columns appearing in a [FOCUS dataset](#).

### 3.2.4. Requirements

Currency-related columns MUST be represented as a three-letter alphabetic code as dictated in the governing document [ISO 4217:2015](#).

### 3.2.5. Exceptions

None

### 3.2.6. Introduced (version)

0.5

## 3.3. Date/Time Format

Columns that provide date and time information conforming to specified rules and formatting requirements ensure clarity, accuracy, and ease of interpretation for both humans and systems.

All columns capturing a date/time value, defined in the FOCUS specification, MUST follow the formatting requirements listed below. Custom date/time-related columns SHOULD also follow the same formatting requirements.

### 3.3.1. Attribute ID

DateTimeFormat

### 3.3.2. Attribute Name

Date/Time Format

### 3.3.3. Description

Rules and formatting requirements for date/time-related columns appearing in a [FOCUS dataset](#).

### 3.3.4. Requirements

- Date/time values MUST be in UTC (Coordinated Universal Time) to avoid ambiguity and ensure consistency across different time zones.
- Date/time values format MUST be aligned with ISO 8601 standard, which provides a globally recognized format for representing dates and times (see [ISO 8601-1:2019](#) governing document for details).
- Values providing information about a specific moment in time MUST be represented in the extended ISO 8601 format with UTC offset ('YYYY-MM-DDTHH:mm:ssZ') and conform to the following guidelines:
  - Include the date and time components, separated with the letter 'T'
  - Use two-digit hours (HH), minutes (mm), and seconds (ss).
  - End with the 'Z' indicator to denote UTC (Coordinated Universal Time)

### 3.3.5. Exceptions

None

### 3.3.6. Introduced (version)

0.5

## 3.4. Discount Handling

A discount is a pricing construct where providers offer a reduced price for [services](#). Providers may have many types of discounts, including but not limited to commercially [negotiated discounts](#), [commitment discounts](#) when you agree to a certain amount of usage or spend, and bundled discounts where you receive free or discounted usage of one product or *service* based on the usage of another. Discount Handling is commonly used in scenarios like verifying discounts were applied and calculating cost savings.

Some discount offers can be purchased from a provider to get reduced prices. The most common example is a *commitment discount*, where you "purchase" a commitment to use or spend a specific amount within a period. When a commitment isn't fully utilized, the unused amount reduces the potential savings from the

discount and can even result in paying higher costs than without the discount. Due to this risk, unused commitment amounts need to be clearly identifiable at a granular level. To facilitate this, unused commitments are recorded with a separate row for each charge period where the commitment was not fully utilized. To show the impact of purchased discounts on each discounted row, discount purchases need the purchase amount to be amortized over the [term](#) the discount is applied to (e.g., 1 year) with each [charge period](#) split and applied to each row that received the discount.

Amortization is a process used to break down and spread purchase costs over a period of time or *term* of use. When a purchase is applicable to resources, like *commitment discounts*, the amortized cost of a resource takes the initial payment and *term* into account and distributes it out based on the resource's usage, attributing the prorated cost for each unit of billing. Amortization enables users of billing data to distribute purchase charges to the appropriate audience in support of cost allocation efforts. Discount Handling for purchased commitments is commonly used for scenarios like calculating utilization and implementing chargeback for the purchase amount.

While providers may use different terms to describe discounts, FOCUS identifies a discount as being a reduced price applied directly to a row. Any price or cost reductions that are awarded after the fact are identified as a "Credit" Charge Category. One example might be when a provider offers a reduced rate after passing a certain threshold of usage or spend.

All rows defined in FOCUS MUST follow the discount handling requirements listed below.

### 3.4.1. Attribute ID

DiscountHandling

### 3.4.2. Attribute Name

Discount Handling

### 3.4.3. Description

Indicates how to include and apply discounts to usage charges or rows in a FOCUS dataset.

### 3.4.4. Requirements

- All applicable discounts SHOULD be applied to each row they pertain to and SHOULD NOT be negated in a separate row.
- All discounts applied to a row MUST apply to the entire charge.
  - Multiple discounts MAY apply to a row, but they MUST apply to the entire charge covered by that row.
  - If a discount only applies to a portion of a charge, then the discounted portion of the charge MUST be split into a separate row.
  - Each discount MUST be identifiable using existing FOCUS columns.
    - Rows with a *commitment discount* applied to them MUST include a CommitmentDiscountId.
    - If a provider applies a discount that cannot be represented by a FOCUS column, they SHOULD include additional columns to identify the source of the discount.

- Purchased discounts (e.g., *commitment discounts*) MUST be amortized.
  - The BilledCost MUST be 0 for any row where the commitment covers the entire cost for the charge period.
  - The EffectiveCost MUST include the portion of the amortized purchase cost that applies to this row.
  - The sum of the EffectiveCost for all rows where CommitmentDiscountStatus is "Used" or "Unused" for each CommitmentDiscountId over the entire duration of the commitment MUST be the same as the total BilledCost of the *commitment discount*.
  - The CommitmentDiscountId and ResourceId MUST be set to the ID assigned to the *commitment discount*. ChargeCategory MUST be set to "Purchase" on rows that represent a purchase of a *commitment discount*.
  - CommitmentDiscountStatus MUST be "Used" for ChargeCategory "Usage" rows that received a reduced price from a commitment. CommitmentDiscountId MUST be set to the ID assigned to the discount. ResourceId MUST be set to the ID of the resource that received the discount.
  - If a commitment is not fully utilized, the provider MUST include a row that represents the unused portion of the commitment for that *charge period*. These rows MUST be represented with CommitmentDiscountStatus set to "Unused" and ChargeCategory set to "Usage". Such rows MUST have their CommitmentDiscountId and ResourceId set to the ID assigned to the *commitment discount*.
- Credits that are applied after the fact MUST use a ChargeCategory of "Credit".

### 3.4.5. Exceptions

None

### 3.4.6. Introduced (version)

1.0-preview

## 3.5. Key-Value Format

Columns that provide Key-Value information are often used in place of separate columns for enumerating data which would be inherently sparse and/or without predetermined keys. This consolidates related information and provides more consistency in the schema. Key-value pairs are also referred to as name-value pairs, attribute-value pairs, or field-value pairs.

All key-value related columns defined in the FOCUS specification MUST follow the key-value formatting requirements listed below.

### 3.5.1. Attribute ID

KeyValueFormat

### 3.5.2. Attribute Name

Key-Value Format



### 3.5.3. Description

Rules and formatting requirements for columns appearing in a [FOCUS dataset](#) that convey data as key-value pairs.

### 3.5.4. Requirements

- Key-Value Format columns MUST contain a serialized JSON string, consistent with the [ECMA 404](#) definition of an object.
- Keys in a key-value pair MUST be unique within an object.
- Values in a key-value pair MUST be one of the following types: number, string, true, false, or null.
- Values in a key-value pair MUST NOT be an object or an array.

### 3.5.5. Exceptions

None

### 3.5.6. Introduced (version)

1.0-preview

## 3.6. Null Handling

Cost data [rows](#) that don't have a value that can be presented for a column must be handled in a consistent way to reduce friction for FinOps practitioners who consume the data for analysis, reporting, and other use cases.

All columns defined in the FOCUS specification MUST follow the null handling requirements listed below. Custom columns SHOULD also follow the same formatting requirements.

### 3.6.1. Attribute ID

NullHandling

### 3.6.2. Attribute Name

Null Handling

### 3.6.3. Description

Indicates how to handle columns that don't have a value.

#### 3.6.4. Requirements

- Columns MUST use NULL when there isn't a value that can be specified for a nullable column.
- Columns MUST NOT use empty strings or placeholder values such as 0 for numeric columns or "Not Applicable" for string columns to represent a null or not having a value, regardless of whether the column allows nulls or not.

#### 3.6.5. Exceptions

None

#### 3.6.6. Introduced (version)

0.5

### 3.7. Numeric Format

Columns that provide numeric values conforming to specified rules and formatting requirements ensure clarity, accuracy, and ease of interpretation for humans and systems. The FOCUS specification does not require a specific level of precision for numeric values. The level of precision required for a given column is determined by the provider and should be part of a data definition published by the provider.

All columns capturing a numeric value, defined in the FOCUS specification, MUST follow the formatting requirements listed below. Custom numeric value capturing columns SHOULD adopt the same format requirements over time.

#### 3.7.1. Attribute ID

NumericFormat

#### 3.7.2. Attribute Name

Numeric Format

#### 3.7.3. Description

Rules and formatting requirements for numeric columns appearing in a [FOCUS dataset](#).

### 3.7.4. Requirements

- Columns with a Numeric value format MUST contain a single numeric value.
- Numeric values MUST be expressed as an integer value, a decimal value, or a value expressed in scientific notation. Fractional notation MUST NOT be used.
- Numeric values expressed using scientific notation MUST be expressed using E notation "mEn" with a real number m and an integer n indicating a value of "m x 10<sup>n</sup>". The sign of the exponent MUST only be expressed as part of the exponent value if n is negative.
- Numeric values MUST NOT be expressed with mathematical symbols, functions, or operators.
- Numeric values MUST NOT contain qualifiers or additional characters (e.g., currency symbols, units of measure, etc.).
- Numeric values MUST NOT contain commas or punctuation marks except for a single decimal point (".") if required to express a decimal value.
- Numeric values MUST NOT include a character to represent a sign for a positive value. A negative sign (-) MUST indicate a negative value.
- Columns with a Numeric value format MUST present one of the following values as the "Data type" in the column definition.

- Allowed values:

Data Type	Type Description
Integer	Specifies a numeric value represented by a whole number or by zero. Integer number formats correspond to standard data types defined by ISO/IEC 9899:2018
Decimal	Specifies a numeric value represented by a decimal number. Decimal formats correspond to ISO/IEC/IEEE 60559:2011 and IEEE 754-2008 definitions.

- Providers SHOULD define precision and scale for Numeric Format columns using one of the following precision values in a data definition document that providers publish.

- Allowed values:

Data Type	Precision	Definition	Range / Significant Digits
Integer	Short	16-bit signed short int ISO/IEC 9899:2018	-32,767 to +32,767
Integer	Long	32-bit signed long int ISO/IEC 9899:2018	-2,147,483,647 to +2,147,483,647
Integer	Extended	64-bit signed two's complement integer <i>or higher</i>	-(2 <sup>63</sup> - 1) to (2 <sup>63</sup> - 1)
Decimal	Single	32-bit binary format IEEE 754-2008 floating-point (decimal32)	9
Decimal	Double	64-bit binary format IEEE 754-2008 floating-point (decimal64)	16
Decimal	Extended	128-bit binary format IEEE 754-2008 floating-point (decimal128) or higher	36+

#### 3.7.4.1. Examples

This format requires that single numeric values be represented using an integer or decimal format without

additional characters or qualifiers. The following lists provide examples of values that meet the requirements and those that do not.

- Values Meeting Numeric Requirements:
  - -100.2
  - -3
  - 4
  - 35.2E-7
  - 1.234
- Values NOT Meeting Numeric Requirements
  - 1 1/2 - contains fractional notation
  - 35.2E+7 - contains a positive exponent with a sign
  - 35.24 x 10^7 - contains an invalid format for scientific notation
  - [3,5,8] - contains an array
  - [4:5] - contains a range
  - 5i + 4 - contains a complex number
  - sqrt(2) - contains a mathematical symbol or operation
  - 2.3^3 - contains an exponent
  - 32 GiB - contains a unit of measure
  - \$32 - contains a currency symbol
  - 3,432,342 - contains a comma
  - +333 - contains a positive sign

### 3.7.5. Exceptions

None

### 3.7.6. Introduced (version)

1.0-preview

## 3.8. String Handling

Columns that capture string values conforming to specified requirements foster data integrity, interoperability, and consistency, improve data analysis and reporting, and support reliable data-driven decision-making.

All columns capturing a string value, defined in the FOCUS specification, **MUST** follow the requirements listed below. Custom string value capturing columns **SHOULD** adopt the same requirements over time.

### 3.8.1. Attribute ID

StringHandling

### 3.8.2. Attribute Name

### 3.8.3. Description

Requirements for string-capturing columns appearing in a [FOCUS dataset](#).

### 3.8.4. Requirements

- String values MUST maintain the original casing, spacing, and other relevant consistency factors as specified by providers and end-users.
- [Charges](#) to mutable entities (e.g., resource names) MUST be accurately reflected in corresponding *charges* incurred after the change and MUST NOT alter *charges* incurred before the change, preserving data integrity and auditability for all *charge* records.
- Immutable string values that refer to the same entity (e.g., resource identifiers, region identifiers, etc.) MUST remain consistent and unchanged across all [billing periods](#).
- Empty strings and strings consisting solely of spaces SHOULD NOT be used in not-nullable string columns.

### 3.8.5. Exceptions

- When a record is provided after a change to a mutable string value and the [ChargeClass](#) is "Correction", the record MAY contain the altered value.

### 3.8.6. Introduced (version)

1.0

## 3.9. Unit Format

Billing data frequently captures data measured in units related to data size, count, time, and other [dimensions](#). The Unit Format attribute provides a standard for expressing units of measure in columns appearing in a [FOCUS dataset](#).

All columns defined in FOCUS specifying Unit Format as a value format MUST follow the requirements listed below.

### 3.9.1. Attribute ID

UnitFormat

### 3.9.2. Attribute Name

### 3.9.3. Description

Indicates standards for expressing measurement units in columns appearing in a *FOCUS dataset*.

### 3.9.4. Requirements

- Units SHOULD be expressed as a single unit of measure adhering to one of the following three formats.
  - <plural-units> - "GB", "Seconds"
  - <singular-unit>-<plural-time-units> - "GB-Hours", "MB-Days"
  - <plural-units>/<singular-time-unit> - "GB/Hour", "PB/Day"
- Units MAY be expressed with a unit quantity or time interval. If a unit quantity or time interval is used, the unit quantity or time interval MUST be expressed as a whole number. The following formats are valid:
  - <quantity> <plural-units> - "1000 Tokens", "1000 Characters"
  - <plural-units>/<interval> <plural-time-units> - "Units/3 Months"
- Unit values and components of columns using the Unit Format MUST use a capitalization scheme that is consistent with the capitalization scheme used in this attribute if that term is listed in this section. For example, a value of "gigabyte-seconds" would not be compliant with this specification as the terms "gigabyte" and "second" are listed in this section with the appropriate capitalization. If the unit is not listed in the table, it is to be used over a functional equivalent with a similar meaning with the same capitalization scheme.
- Units SHOULD be composed of the list of recommended units listed in this section unless the unit value covers a *dimension* not listed in the recommended unit set, or if the unit covers a count-based unit distinct from recommended values in the count *dimension* listed in this section.

#### 3.9.4.1. Data Size Unit Names

Data size unit names MUST be abbreviated using one of the abbreviations in the following table. For example, a unit name of "TB" is a valid unit name, and a unit name of "terabyte" is an invalid unit name. Data size abbreviations can be considered both the singular and plural form of the unit. For example, "GB" is both the singular and plural form of the unit "gigabyte", and "GBs" would be an invalid unit name. Values that exceed  $10^{18}$  MUST use the abbreviation for exabit, exabyte, exbibit, and exbibyte, and values smaller than a byte MUST use the abbreviation for bit or byte. For example, the abbreviation "YB" for "yottabyte" is not a valid data size unit name as it represents a value larger than what is listed in the following table.

The following table lists the valid abbreviations for data size units from a single bit or byte to  $10^{18}$  bits or bytes.

Data size in bits	Data size in bytes
b (bit) = $10^1$	B (byte = $10^1$ )
Kb (kilobit = $10^3$ )	KB (kilobyte = $10^3$ )
Mb (megabit = $10^6$ )	MB (megabyte = $10^6$ )
Gb (gigabit = $10^9$ )	GB (gigabyte = $10^9$ )
Tb (terabit = $10^{12}$ )	TB (terabyte = $10^{12}$ )
Pb (petabit = $10^{15}$ )	PB (petabyte = $10^{15}$ )

Data size in bits	Data size in bytes
Eb (exabit = $10^{18}$ )	EB (exabyte = $10^{18}$ )
Kib (kibibit = $2^{10}$ )	KiB (kibibyte = $2^{10}$ )
Mib (mebibit = $2^{20}$ )	MiB (mebibyte = $2^{20}$ )
Gib (gibibit = $2^{30}$ )	GiB (gibibyte = $2^{30}$ )
Tib (tebibit = $2^{40}$ )	TiB (tebibyte = $2^{40}$ )
Pib (pebibit = $2^{50}$ )	PiB (pebibyte = $2^{50}$ )
Eib (exbibit = $2^{60}$ )	EiB (exbibyte = $2^{60}$ )

### 3.9.4.2. Count-based Unit Names

A count-based unit is a noun that represents a discrete number of items, events, or actions. For example, a count-based unit can be used to represent the number of requests, instances, tokens, or connections.

If the following list of recommended values does not cover a count-based unit, a provider MAY introduce a new noun representing a count-based unit. All nouns appearing in units that are not listed in the recommended values table will be considered count-based units. A new count-based unit value MUST be capitalized.

Count
Count
Unit
Request
Token
Connection
Certificate
Domain
Core

### 3.9.4.3. Time-based Unit Names

A time-based unit is a noun that represents a time interval. Time-based units can be used to measure consumption over a time interval or in combination with another unit to capture a rate of consumption. Time-based units MUST match one of the values listed in the following table.

Time
Year
Month
Day
Hour
Minute
Second

### 3.9.4.4. Composite Units

If the unit value is a composite value made from combinations of one or more units, each component **MUST** also align with the set of recommended values.

Instead of "per" or "-" to denote a Composite Unit, slash ("/") and space(" ") **MUST** be used as a common convention. Count-based units like requests, instances, and tokens **SHOULD** be expressed using a value listed in the count *dimension*. For example, if a usage unit is measured as a rate of requests or instances over a period of time, the unit **SHOULD** be listed as "Requests/Day" to signify the number of requests per day.

### **3.9.5. Exceptions**

None

### **3.9.6. Introduced (version)**

1.0-preview



## 4. Metadata

The FOCUS specification defines a metadata structure to be supplied by data providers to facilitate practitioners' use of FOCUS data. This metadata includes general information about the data generator and the schema of the [FOCUS dataset](#).

FOCUS Metadata SHOULD be provided in a format that is accessible programmatically, such as a file, website, API, or table. Providers SHOULD provide documentation on their implementation of the FOCUS metadata.

### 4.1. Data Generator

The FOCUS metadata about the generator of the FOCUS data.

#### 4.1.1. Requirements

The FOCUS Data Generator metadata MUST be provided. This metadata MUST be of type Object and MUST NOT contain null values.

#### 4.1.2. Schema Example

For an example of the FOCUS Data Generator metadata please refer to: [Data Generator Example](#)

#### 4.1.3. Data Generator

Human-readable name of the entity that is generating the data.

The DataGenerator MUST be provided in the metadata. DataGenerator MUST be of type String and MUST NOT be null. The DataGenerator SHOULD be easily associated with the provider who generated the [FOCUS dataset](#).

##### 4.1.3.1. Metadata ID

DataGenerator

##### 4.1.3.2. Metadata Name

Data Generator

#### 4.1.3.3. Content constraints

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

#### 4.1.3.4. Introduced (version)

1.0

## 4.2. Schema

The schema metadata object and its contents provides information about the structure of the data provided.

### 4.2.1. Requirements

#### 4.2.1.1. Reference to FOCUS Data

FOCUS data artifacts, whether they are data files, data streams, or data tables, MUST provide a clear reference to the schema of the data. This reference MUST be retrievable without inspection of the contents of the FOCUS data within the data artifact. For some delivery mechanisms such as database tables, the provider may rely on the schema functionality of the providing system.

It is recommended that the schema reference be provided as an external reference rather than included in full as metadata accompanying the data artifact. This allows for easier understanding of when changes to the schema of the [FOCUS datasets](#) occurs.

#### 4.2.1.2. Schema Metadata Creation

Should the provider change the structure of the supplied FOCUS data artifact, a new schema metadata object MUST be supplied. These scenarios include, but are not limited to:

- [Adding a new column](#)
- [Removing a column](#)
- [Changing column metadata](#)
- [FOCUS Version has changed](#)
- [Provider Version has changed](#)
- [Correcting schema metadata errors](#)

#### 4.2.1.3. Schema Metadata Updates

Should there be an error where the schema metadata object does not match the schema of the FOCUS data

artifact, the provider MUST update the schema metadata object to match the schema of the FOCUS data artifact. This is to ensure that the schema metadata object is always accurate.

### 4.2.2. Schema Example

For an example of the FOCUS schema metadata please refer to: [Schema Metadata Example](#)

### 4.2.3. Schema ID

The Schema ID provides the reference item to associate which Schema was used for the generation of a FOCUS Dataset.

The Schemald MUST be present in the metadata. The Schemald MUST be of String. It is RECOMMENDED for Schemald to be a Globally Unique Identifier (GUID).

#### 4.2.3.1. Metadata ID

Schemald

#### 4.2.3.2. Metadata Name

Schema ID

#### 4.2.3.3. Content constraints

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	STRING
Value format	Recommend GUID String

#### 4.2.3.4. Introduced (version)

1.0

### 4.2.4. Creation Date

Date the schema was created.

The CreationDate MUST be present in the metadata. This MUST be of type Date/Time and MUST NOT contain null values. CreationDate MUST conform to [DateTimeFormat](#).

#### 4.2.4.1. Metadata ID

CreationDate

#### 4.2.4.2. Metadata Name

Creation Date

#### 4.2.4.3. Content constraints

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	Date/Time
Value format	<a href="#">Date/Time</a> <a href="#">Format</a>

#### 4.2.4.4. Introduced (version)

1.0

### 4.2.5. FOCUS Version

The version of FOCUS utilized for building the dataset.

The FocusVersion MUST be provided in the metadata. FocusVersion MUST be of type String and MUST NOT contain null values. FocusVersion MUST match one of the published versions of the FOCUS specification. FocusVersion MUST match the version of the FOCUS specification that the [FOCUS dataset](#) conforms to.

#### 4.2.5.1. Metadata ID

FocusVersion

#### 4.2.5.2. Metadata Name

FOCUS Version

#### 4.2.5.3. Content constraints

Constraint	Value
------------	-------

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	STRING
Value format	Must align with a published FocusVersion

#### 4.2.5.4. Introduced (version)

1.0

### 4.2.6. Provider Version

The ProviderVersion MAY be supplied to declare the version of logic by which the [FOCUS dataset](#) was generated and is separate from FOCUS Version. ProviderVersion allows for the provider to specify changes that may not result in a structural change in the data. It is suggested that the provider version use a versioning approach such as [SemVer](#) version.

ProviderVersion MUST be of type String and MUST NOT contain null values. If FocusVersion is changed a new ProviderVersion MUST be also changed. The provider MUST document what changes are present in the ProviderVersion.

#### 4.2.6.1. Metadata ID

ProviderVersion

#### 4.2.6.2. Metadata Name

Provider Version

#### 4.2.6.3. Content constraints

Constraint	Value
Feature level	Optional
Allows nulls	False
Data type	STRING
Value format	<not specified>

#### 4.2.6.4. Introduced (version)

1.1

### 4.2.7. Column Definition

The FOCUS metadata schema column definition provides a list of the columns present in the [FOCUS dataset](#) along with metadata about the columns.

#### 4.2.7.1. Requirements

This metadata MUST be present in the FOCUS metadata schema. This metadata MUST be of type Object and MUST NOT contain null values.

#### 4.2.7.2. Column Name

The name of the column provided in the [FOCUS dataset](#).

The ColumnName MUST be provided in the FOCUS Metadata schema. ColumnName MUST be of type String and MUST NOT contain null values.

##### 4.2.7.2.1. Metadata ID

ColumnName

##### 4.2.7.2.2. Metadata Name

Column Name

##### 4.2.7.2.3. Content constraints

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

##### 4.2.7.2.4. Introduced (version)

1.0

#### 4.2.7.3. Data Type

The data type of the column provided in the [FOCUS dataset](#).

The DataType MUST be provided in the FOCUS Metadata schema. DataType MUST be of type String and MUST NOT contain null values.

##### 4.2.7.3.1. Metadata ID

DataType

#### 4.2.7.3.2. Metadata Name

Data Type

#### 4.2.7.3.3. Content constraints

Constraint	Value
Feature level	Mandatory
Allows nulls	False
Data type	String
Value format	<not specified>

#### 4.2.7.3.4. Introduced (version)

1.0

#### 4.2.7.4. Numeric Precision

Numeric Precision is the maximum number of digits for the values in the column.

NumericPrecision SHOULD be provided in the FOCUS Metadata schema for Numeric Format columns.

NumericPrecision MUST be of type Integer and MUST NOT contain null values.

#### 4.2.7.4.1. Metadata ID

NumericPrecision

#### 4.2.7.4.2. Metadata Name

Numeric Precision

#### 4.2.7.4.3. Content constraints

Constraint	Value
Feature level	Conditional
Allows nulls	False
Data type	Integer
Value format	<a href="#">Numeric Format</a>

#### 4.2.7.4.4. Introduced (version)

1.0

#### 4.2.7.5. Number Scale

The number scale of the data provides the maximum number of digits after the decimal point in decimal numbers.

NumberScale SHOULD be provided in the FOCUS Metadata schema for Decimal columns. NumberScale MUST be of type Integer and MUST NOT contain null values.

##### 4.2.7.5.1. Metadata ID

NumberScale

##### 4.2.7.5.2. Metadata Name

Number Scale

##### 4.2.7.5.3. Content constraints

Constraint	Value
Feature level	Conditional
Allows nulls	False
Data type	Integer
Value format	<a href="#">Numeric Format</a>

#### 4.2.7.5.4. Introduced (version)

1.0

#### 4.2.7.6. Provider Tag Prefixes

The Provider Tag Prefixes defines the list of prefixes used in the tag name of provider-defined [tags](#). This metadata is useful for the consumer to identify which tags are provider-defined vs user-defined.

The ProviderTagPrefixes MUST be provided when ColumnName is equal to Tags. The ProviderTagPrefix MUST be of type Array of Strings. The ProviderTagPrefixes SHOULD be easily associated with the provider who generated the [FOCUS dataset](#).

##### 4.2.7.6.1. Metadata ID

ProviderTagPrefixes



#### 4.2.7.6.2. Metadata Name

Provider Tag Prefixes

#### 4.2.7.6.3. Content constraints

Constraint	Value
Feature level	Conditional
Allows nulls	False
Data type	Array
Value format	STRING datatype values in the array

#### 4.2.7.6.4. Introduced (version)

1.0

#### 4.2.7.7. String Encoding

The string encoding scheme of the column provided in the [FOCUS dataset](#).

StringEncoding SHOULD be provided in the FOCUS Metadata schema when it is required to know this information in order to successfully read the data. StringEncoding MUST be of type String and MUST NOT contain null values.

##### 4.2.7.7.1. Metadata ID

StringEncoding

##### 4.2.7.7.2. Metadata Name

StringEncoding

#### 4.2.7.7.3. Content constraints

Constraint	Value
Feature level	Conditional
Allows nulls	False
Data type	String
Value format	<not specified>

#### 4.2.7.7.4. Introduced (version)

1.0

#### 4.2.7.8. String Max Length

The string max length of the data that can be stored in the column.

StringMaxLength SHOULD be provided in the FOCUS Metadata schema for String columns. StringMaxLength MUST be of type Integer and MUST NOT contain null values.

##### 4.2.7.8.1. Metadata ID

StringMaxLength

##### 4.2.7.8.2. Metadata Name

String Max Length

##### 4.2.7.8.3. Content constraints

Constraint	Value
Feature level	Conditional
Allows nulls	False
Data type	Integer
Value format	<a href="#">Numeric Format</a>

##### 4.2.7.8.4. Introduced (version)

1.0

## 5. Use Case Library

This specification is based on a set of common FinOps use cases, which are publicly available at <https://focus.finops.org/use-cases/>(<https://focus.finops.org/use-cases/>). Developed by FinOps practitioners, these use cases are organized by persona and capability, making it easy to find relevant scenarios. Each use case includes sample SQL queries to help you get started with implementation.

## 6. Glossary

### **Adjustment**

A charge representing a modification to billing data to account for certain events or circumstances not previously captured, or captured incorrectly. Examples include billing errors, service disruptions, or pricing changes.

### **Amortization**

The distribution of upfront costs over time to accurately reflect the consumption or benefit derived from the associated resources or services. Amortization is valuable when the commitment period (time duration of the cost) extends beyond the granularity of the source report.

### **Availability Zone**

A collection of geographically separated locations containing a data center or cluster of data centers. Each availability zone (AZ) should have its own power, cooling, and networking, to provide redundancy and fault tolerance.

### **Billed Cost**

A charge that serves as the basis for invoicing. It includes the total amount of fees and discounts, signifying a monetary obligation. Valuable when reconciling cash outlay with incurred expenses is required, such as cost allocation, budgeting, and invoice reconciliation.

### **Billing Account**

A container for resources and/or services that are billed together in an invoice. A billing account may have sub accounts, all of whose costs are consolidated and invoiced to the billing account.

### **Billing Currency**

An identifier that represents the currency that a charge for resources and/or services was billed in.

### **Billing Period**

The time window that an organization receives an invoice for, inclusive of the start date and exclusive of the end date. It is independent of the time of usage and consumption of resources and services.

### **Block Pricing**

A pricing approach where the cost of a particular resource or service is determined based on predefined quantities or tiers of usage. In these scenarios, the Pricing Unit and the corresponding Pricing Quantity can be different from the Consumed Unit and Consumed Quantity.

### **Capacity Reservation**

A capacity reservation is an agreement that secures a dedicated amount of resources or services for a specified period. This ensures the reserved capacity is always available and accessible, even if it's not fully utilized. Customers are typically charged for the reserved capacity, regardless of actual consumption.

### **Charge**

A row in a FOCUS-compatible cost and usage dataset.

### **Charge Period**

The time window for which a charge is effective, inclusive of the start date and exclusive of the end date. The charge period for continuous usage should match the time granularity of the dataset (e.g., 1 hour for

hourly, 1 day for daily). The charge period for a non-usage charge with time boundaries should match the duration of eligibility.

### **Cloud Service Provider (CSP)**

A company or organization that provides remote access to computing resources, infrastructure, or applications for a fee.

### **Commitment**

A customer's agreement to consume a specific quantity of a service or resource over a defined period, usually also creating a financial commitment throughout the entirety of the commitment period. Some commitments also hold Providers to certain assurance levels of resource availability.

### **Commitment Discount**

A billing discount model that offers reduced rates on preselected SKUs in exchange for an obligated usage or spend amount over a predefined term. Commitment discount purchases, made upfront and/or with recurring monthly payments are amortized evenly across predefined charge periods (i.e. hourly), and unused amounts cannot be carried over to subsequent charge periods. Commitment discounts are publicly available to customers without special contract arrangements.

### **Commitment Discount Flexibility**

A feature of [commitment discounts](#) that may further transform the predetermined amount of usage purchased or consumed based on additional, provider-specific requirements.

### **Contracted Unit Price**

The agreed-upon unit price for a single [Pricing Unit](#) of the associated SKU, inclusive of negotiated discounts, if present, and exclusive of any other discounts. This price is denominated in the [Billing Currency](#).

### **Dimension**

A specification-defined categorical attribute that provides context or categorization to billing data.

### **Effective Cost**

The amortized cost of the charge after applying all reduced rates, discounts, and the applicable portion of relevant, prepaid purchases (one-time or recurring) that covered this charge.

### **Exclusive Bound**

A Date/Time Format value that is not contained within the ending bound of a time period.

### **Finalized Tag**

A tag with one tag value chosen from a set of possible tag values after being processed by a set of provider-defined or user-defined rules.

### **FinOps Cost and Usage Specification (FOCUS)**

An open-source specification that defines requirements for billing data.

### **FOCUS Dataset**

A structured collection of cost and usage data that meets or exceeds the Basic compliance criteria of FOCUS.

### **Inclusive Bound**

A Date/Time Format value that is contained within the beginning bound of a time period.

### **Interruptible**

A category of compute resources that can be paused or terminated by the CSP within certain criteria, often

advertised at reduced unit pricing when compared to the equivalent non-interruptible resource.

### **List Unit Price**

The suggested provider-published unit price for a single [Pricing Unit](#) of the associated [SKU](#), exclusive of any discounts. This price is denominated in the [Billing Currency](#).

### **Managed Service Provider (MSP)**

A company or organization that provides outsourced management and support of a range of IT services, such as network infrastructure, cybersecurity, cloud computing, and more.

### **Metric**

A FOCUS-defined column that provides numeric values, allowing for aggregation operations such as arithmetic operations (sum, multiplication, averaging etc.) and statistical operations.

### **Negotiated Discount**

A contractual agreement where a customer commits to specific spend or usage goals over a [term](#) in exchange for discounted rates across varying SKUs. Unlike [commitment discounts](#), negotiated discounts are typically more customized to customer's accounts, can be utilized at varying frequencies, and may overlap with *commitment discounts*.

### **On-Demand**

A term that describes a service that is available and provided immediately or as needed, without requiring a pre-scheduled appointment or prior arrangement. In cloud computing, virtual machines can be created and terminated as needed, i.e. on demand.

### **Pascal Case**

Pascal Case (PascalCase, also known as UpperCamelCase) is a format for identifiers which contain one or more words meaning the words are concatenated together with no delimiter and the first letter of each word is capitalized.

### **Potato**

A long and often painful conversation had by the FOCUS contributors. Sometimes the name of a thing that we could not yet name. No starchy root vegetables were harmed during the production of this specification. We thank potato for its contribution in the creation of this specification.

### **Practitioner**

An individual who performs FinOps within an organization to maximize the business value of using cloud and cloud-like services.

### **Price List**

A comprehensive list of prices offered by a provider.

### **Provider**

An entity that made internal or 3rd party resources and/or services available for purchase.

### **Resource**

A unique component that incurs a charge.

### **Row**

A row in a FOCUS-compatible cost and usage dataset.

### **Service**

An offering that can be purchased from a provider, and can include many types of usage or other charges; eg., a cloud database service may include compute, storage, and networking charges.

### **SKU**

A construct composed of the common properties of a product offering associated with one or many SKU Prices.

### **SKU Price**

The unit price used to calculate a charge that is associated with one SKU. SKU Prices are usually referenced from the provider's price list and are unique to various providers.

### **Sub Account**

A sub account is an optional provider-supported construct for organizing resources and/or services connected to a billing account. Sub accounts must be associated with a billing account as they do not receive invoices.

### **Tag**

A metadata label assigned to a resource to provide information about it or to categorize it for organizational and management purposes.

### **Tag Source**

A Resource or Provider-defined construct for grouping resources and/or other Provider-defined construct that a Tag can be assigned to.

### **Term**

A duration of a contractual agreement like with a [\*commitment discount\*](#) or [\*negotiated discount\*](#).

## 7. Appendix

*This section is non-normative.*

### 7.1. Commitment Discounts

A [commitment discount](#) is a billing discount model that offers reduced rates on preselected [SKUs](#) in exchange for an obligated usage or spend amount over a predefined [term](#). *Commitment discounts* typically consist of purchase and usage records within cost and usage datasets.

Usage-based *commitment discounts* obligate a customer to a predetermined amount of usage over a preselected *term*. In some cases, usage-based *commitment discounts* also feature [commitment discount flexibility](#) which may expand the types of [resources](#) that a *commitment discount* can cover. It is important to note when mixing *commitment discounts* with and without *commitment discount flexibility*, the [CommitmentDiscountUnit](#) should reflect this difference.

Spend-based commitment discounts obligate a customer to a predetermined amount of spend over a preselected *term*. In the usage examples below, each [row](#) measures the monetary amount of the hourly commit consumed by the *commitment discount*, so the [CommitmentDiscountUnit](#) chosen is "USD", or the [billing currency](#).

#### 7.1.1. Purchasing

While customers are bound to the *term* of a *commitment discounts*, providers offer some or all of the following payment options before and/or during the *term*:

- *All Upfront* - The *commitment discounts* is paid in full before the *term* begins.
- *No Upfront* - The *commitment discounts* is paid on a repeated basis, typically over each [billing period](#) of the *term*.
- *Partial Upfront* - Some of the *commitment discounts* is paid before the *term* begins, and the rest is paid repeatedly over the *term*.

For example, if a customer buys a 1-year, spend-based *commitment discount* with a \$1.00 hourly commit and pays with the partial option, the *commitment discount's* payment consists of a one-time purchase in the beginning of the *term* and monthly recurring purchases with the following totals:

1. *One-Time* - \$4,380 (24 hours \* 365 days \* &dollar;1.00 \* 0.5)
2. *Recurring* - \$182.50 (24 hours \* 365 days \* &dollar;1.00 / 12 months)

#### 7.1.2. Usage

Commitment discounts follow a "use-it-or-lose-it" model where the [amortization](#) of a *commitment discount's* purchase applies evenly to eligible *resources* over each [charge period](#) of the *term*.

For example, if a customer buys a spend-based *commitment discount* with a \$1.00 hourly commit in January (31 days), only \$1.00 is eligible for consumption for each hourly *charge period*. If a customer has eligible *resources* running during this *charge period*, an amount of up to \$1.00 will be allocated to these *resources*. Conversely, if a customer does have eligible *resources* running that fully take advantage of this \$1.00 during



this *charge period*, then some or all of this amount will go to waste.

### 7.1.3. Commitment Discounts in FOCUS

Within the FOCUS specification, the following examples demonstrate how a *commitment discount* appears across various payment and usage scenarios.

#### 7.1.3.1. Purchase Rows

All *commitment discount* purchases appear with a positive [BilledCost](#), [PricingCategory](#) as "Standard", and with the *commitment discount's* id populating both the [ResourceId](#) and [CommitmentDiscountId](#) value. One-time purchases appear as a single record with [ChargeCategory](#) as "Purchase", [ChargeFrequency](#) as "One-Time", and the total quantity and units for *commitment discount's term* reflected as [CommitmentDiscountQuantity](#) and [CommitmentDiscountUnit](#), respectively.

Recurring purchases are allocated across all corresponding *charge periods* of the *term* when [ChargeCategory](#) is "Purchase", [ChargeFrequency](#) is "Recurring", and [CommitmentDiscountQuantity](#) and [CommitmentDiscountUnit](#) are reflected only for that *charge period*.

Using the same *commitment discount* example as above with a one-year, spend-based *commitment discount* with a \$1.00 hourly commit purchased on Jan 1, 2023, various purchase options are available:

##### 7.1.3.1.1. Scenario #1: All Upfront

The entire *commitment discount* is billed *once* during the first *charge period* of the *term* for \$8,670 (derived as 24 hours \* 365 days \* &dollar;1.00).

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2024-01-01T00:00:00Z",
    "ChargeCategory": "Purchase",
    "ChargeFrequency": "One-Time",
    "PricingCategory": "Standard",
    "ResourceId": "<commitment-discount-id>",
    "BilledCost": 8760.00,
    "EffectiveCost": 0.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 8760.00,
    "CommitmentDiscountUnit": "USD"
  }
]
```

##### 7.1.3.1.2. Scenario #2: No Upfront

The *commitment discount* is billed across all 8,760 (24 hours \* 365 days) *charge periods* of the *term* with \$1.00 allocated to each *charge period* over the *term*.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Purchase",
    "ChargeFrequency": "Recurring",
    "PricingCategory": "Standard",
    "ResourceId": "<commitment-discount-id>",
    "BilledCost": 1.00,
    "EffectiveCost": 0.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 1.00,
    "CommitmentDiscountUnit": "USD"
  },
  /* ... 8,759 more recurring purchase records for the *term* ... */
]
```

#### 7.1.3.1.3. Scenario #3: Partial Upfront

With a 50/50 split, half of the commitment is billed *once* during the first *charge period* of the *term* for \$4,380 (derived as 24 hours \* 182.5 days \* &dollar;1.00), and the other half is billed across each *charge period* over the term, derived as (&dollar;1.00 \* 8,760 hours \* 0.5). Amortized costs incur half of the amount (i.e. \$0.50) from the one-time purchase and the other half from the recurring purchase.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2024-01-01T00:00:00Z",
    "ChargeCategory": "Purchase",
    "ChargeFrequency": "One-Time",
    "PricingCategory": "Standard",
    "ResourceId": "<commitment-discount-id>",
    "BilledCost": 4380.00,
    "EffectiveCost": 0.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 4380.00,
    "CommitmentDiscountUnit": "USD"
  },
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Purchase",
    "ChargeFrequency": "Recurring",
    "PricingCategory": "Standard",
    "ResourceId": "<commitment-discount-id>",
    "BilledCost": 0.50,
    "EffectiveCost": 0.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 0.50,
    "CommitmentDiscountUnit": "USD"
  },
  /* ... 8,759 more recurring purchase records for the *term* ... */
]
```

### 7.1.3.2. Usage Rows

*Amortization of commitment discounts* occur similarly regardless of how *commitment discount* purchases are made. The same usage-based or spend-based amount is applied evenly across all *charge periods* and potentially allocated to eligible *resources*. Continuing with the same *commitment discount* example, a one-year, spend-based *commitment discount* with a \$1.00 hourly commit and 1 *resource* (for simplicity) yields 4 types of scenarios that can occur during a *charge period*:

- Scenario #1: An eligible *resource* fully consumes the allocated amount (100% utilization)
- Scenario #2: No eligible *resource* consumes the allocated amount (0% utilization)
- Scenario #3: An eligible *resource* partially consumes the allocated amount (75% utilization)
- Scenario #4: An eligible *resource* fully consumes the \$1.00 hourly commit with an overage (100% utilization + overage)

#### 7.1.3.2.1. Scenario #1: An eligible *resource* fully consumes the allocated amount (100% utilization)

In this scenario, one eligible *resource* runs for the full hour and consumes \$1.00, so one *row* allocated to the

*resource* is produced.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Committed",
    "ResourceId": "<resource-id>",
    "ConsumedQuantity": 1,
    "ConsumedUnit": "Hour",
    "BilledCost": 0.00,
    "EffectiveCost": 1.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 1.00,
    "CommitmentDiscountStatus": "Used",
    "CommitmentDiscountUnit": "USD"
  }
]
```

#### 7.1.3.2.2. Scenario #2: No eligible *resource* consumes the allocated amount (0% utilization)

In this situation, the full eligible, \$1.00 amount remained unutilized and results in 1 unused *row*. In this scenario, it is important to note that while *CommitmentDiscountQuantity* is not because \$1 was still drawn down by the *commitment discount* even though, no *resource* was allocated, so [ConsumedQuantity](#) and [ConsumedUnit](#) are null.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Committed",
    "ResourceId": "<commitment-discount-id>",
    "ConsumedQuantity": null,
    "ConsumedUnit": null,
    "BilledCost": 0.00,
    "EffectiveCost": 1.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 1.00,
    "CommitmentDiscountStatus": "Unused",
    "CommitmentDiscountUnit": "USD"
  }
]
```

#### 7.1.3.2.3. Scenario #3: An eligible *resource* partially consumes the allocated amount (75% utilization)

In this scenario, one eligible *resource* runs for the full hour and consumes \$0.75 of the \$1.00 allocation. One *row* shows \$0.75 to a *resource*, and the other *row* shows that \$0.25 was unused.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Committed",
    "ResourceId": "<resource-id>",
    "ConsumedQuantity": 1,
    "ConsumedUnit": "Hour",
    "BilledCost": 0.00,
    "EffectiveCost": 0.75,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 0.75,
    "CommitmentDiscountStatus": "Used",
    "CommitmentDiscountUnit": "USD"
  },
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Committed",
    "ResourceId": "<commitment-discount-id>",
    "ConsumedQuantity": null,
    "ConsumedUnit": null,
    "BilledCost": 0.00,
    "EffectiveCost": 0.25,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 0.25,
    "CommitmentDiscountStatus": "Unused",
    "CommitmentDiscountUnit": "USD"
  }
]
```

#### 7.1.3.2.4. Scenario #4: An eligible *resource* fully consumes the \$1.00 hourly commit with an overage (100% utilization + overage)

In this scenario, one eligible *resource* runs for the full hour and is charged \$1.50. One *row* shows that \$1.00 was *amortized* from the *commitment discount*, and the other shows that \$0.50 was charged as standard, on-demand spend.

```
[
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Committed",
    "ResourceId": "<resource-id>",
    "ConsumedQuantity": 1,
    "ConsumedUnit": "Hour",
    "BilledCost": 0.00,
    "EffectiveCost": 1.00,
    "CommitmentDiscountId": "<commitment-discount-id>",
    "CommitmentDiscountQuantity": 1.00,
    "CommitmentDiscountStatus": "Used",
    "CommitmentDiscountUnit": "USD"
  },
  {
    "BillingPeriodStartDate": "2023-01-01T00:00:00Z",
    "BillingPeriodEndDate": "2023-02-01T00:00:00Z",
    "ChargePeriodStartDate": "2023-01-01T00:00:00Z",
    "ChargePeriodEndDate": "2023-01-01T01:00:00Z",
    "ChargeCategory": "Usage",
    "ChargeFrequency": "Usage-Based",
    "PricingCategory": "Standard",
    "ResourceId": "<resource-id>",
    "ConsumedQuantity": 1,
    "ConsumedUnit": "Hour",
    "BilledCost": 0.50,
    "EffectiveCost": 0.00
  }
]
```

## 7.2. Grouping constructs for resources or services

Providers natively support various constructs for grouping [resources](#) or [services](#). These grouping constructs are often used to mimic organizational structures, technical architectures, cost attribution/allocation and access management boundaries, or other customer-specific structures based on requirements.

Providers may support multiple levels of resource or service grouping mechanisms. FOCUS supports two distinct levels of groupings that are commonly needed for FinOps capabilities like chargeback, invoice reconciliation and cost allocation.

- [Billing account](#): A mandatory container for *resources* or *services* that are billed together in an invoice. *Billing accounts* are commonly used for scenarios like grouping based on organizational constructs, invoice reconciliation and cost allocation strategies.
- [Sub account](#): An optional provider-supported construct for organizing *resources* and *services* connected to a *billing account*. *Sub accounts* are commonly used for scenarios like grouping based on organizational constructs, access management needs and cost allocation strategies. *Sub accounts* must be associated with a *billing account* as they do not receive invoices.

The table below highlights key properties of the two grouping constructs supported by FOCUS.

Property	Billing account	Sub account
Requirement level	Mandatory	Optional
Receives an invoice?	Yes	No
Invoiced at	Self	Associated billing account
Examples	AWS: Management Account* GCP: Billing Account Azure MCA: Billing Profile Snowflake: Organizational Account	AWS: Member Account GCP: Project Azure MCA: Subscription Snowflake: Account

\* For organizations that have multiple AWS Member Accounts within an AWS Organization, consolidated billing is enabled by default and invoices are received at Management Account level. A Member Account can be removed from AWS consolidated billing whereby the removed account receives independent invoices and is responsible for payments.

## 7.3. Origination of Cost Data

Cost data presented in the billing datasets originates from various sources depending on the purchasing mechanism. There are at least 3 different pieces of information that are important for understanding where cost originated from.

- **Provider:** The entity that made the *resources* or *services* available for purchase.
- **Publisher:** The entity that produced the *resources* or *services* that were purchased.
- **Invoice Issuer:** The entity responsible for invoicing for the *resources* or *services* consumed.

The value for each of these may be different depending on the various purchasing scenarios for *resources* or *services*. Use cases for purchasing direct, via a Managed Service Provider (MSP), via a cloud marketplace, and from internal service offerings were considered. The table below presents a few scenarios to show how the value for each dimension may change based on the purchasing scenario.

#	Scenario	Provider	Publisher	Invoice Issuer
1.1	Purchasing cloud services directly from cloud provider	Cloud service provider	Cloud service provider	Cloud service provider
1.2	Purchasing cloud services from the cloud provider where the cloud region is operated by a 3rd party	Cloud service provider	Cloud service provider	Entity operating the region for the cloud service provider
2.1	Purchasing cloud services via MSP	Managed Service Provider	Cloud service provider	Managed Service Provider
2.2	Purchasing cloud-agnostic resources or services built/sold by an MSP	Managed Service Provider	Managed Service Provider	Managed Service Provider
2.3	Purchasing labor services from managed service provider	Managed Service Provider	Managed Service Provider	Managed Service Provider

#	Scenario	Provider	Publisher	Invoice Issuer
3.1	Purchasing a cloud marketplace offering that runs on the cloud provider	Cloud service provider	Company building the software or services (Cloud service provider OR third-party software or services company)	Cloud service provider
3.2	Purchasing a cloud marketplace offering that is not running directly on your cloud infrastructure (e.g., SaaS product, Professional Services)	Cloud service provider	Company producing the SaaS or services product	Cloud service provider
3.3	Purchasing a SaaS product that is not directly running on your cloud infrastructure from a 3rd party reseller managed cloud marketplace	Cloud service provider	SaaS provider	Reseller
4.1	Purchasing SaaS software directly from provider	SaaS provider	SaaS provider	SaaS provider
4.2	Purchasing SaaS software that additionally runs on your cloud resources (in addition to #4.1)	Cloud service provider	Cloud service provider	Cloud service provider
5.1	Purchasing internal infrastructure or services offerings running on-premise	Internal product name	Internal product name	Internal product name
5.2	Purchasing internal infrastructure or services offerings running on cloud	Internal product name	Internal product name	Internal product name
5.3	Associated software license cost for use on an on-premise infrastructure platform (Where license cost is presented separately in cost data)	Internal product name	Company producing the software	Internal product name

## 7.4. Examples

*This section is non-normative.*

### 7.4.1. Metadata Examples

The following sections contain examples of metadata provided by a hypothetical FOCUS data provider called ACME to supply the required reference between the [FOCUS dataset](#) and the schema metadata. Provider implementations will vary on how the metadata is disseminated; however, the provider's chosen metadata delivery approach should be able to support the structure represented in this example.

In this example, the provider supports delivery of FOCUS data via file export to a data storage system. It uses JSON as the format for providing the metadata. The provider delivers data every 12 hours into a path structure described below:

Type of data	Path
Export location	/FOCUS



Type of data	Path
--------------	------

Metadata location	/FOCUS/metadata
Cost data location	/FOCUS/data

Here are some metadata examples for various scenarios:

#### 7.4.1.1. Data Generator Metadata

##### 7.4.1.1.1. Scenario

Acme provides metadata about the data generator as a part of their FOCUS data export. They provide the relevant data via the [Data Generator](#) schema object.

##### 7.4.1.1.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/data\_generator.json.

The updated data generator related metadata could look like this:

```
{
  "DataGenerator": "Acme"
}
```

#### 7.4.1.2. Schema Metadata

##### 7.4.1.2.1. Scenario

ACME has only provided one [Schema](#) for their FOCUS data export. ACME provides a directory of schemas and each schema is a single file. Acme's provides a file representing the schema for the data they provide.

##### 7.4.1.2.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-1234-abcde-12345-abcde-12345.json.

The updated schema related metadata could look like this:

```

{
  "SchemaId": "1234-abcde-12345-abcde-12345",
  "FocusVersion": "1.0",
  "CreationDate": "2024-01-01T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    }
  ]
}

```

#### 7.4.1.3. Schema Metadata to FOCUS Data Reference

##### 7.4.1.3.1. Scenario

ACME makes a change to the [Schema](#) of their data exports. For each FOCUS data export, ACME includes a metadata reference to the schema object. Because multiple files are provided in each export, Acme has

elected to include a metadata file in each export folder that includes the FOCUS schema reference that applies to the data export files within that folder. When the schema changes, they include the new [Schema ID](#) in their export metadata file of the new folder.

#### 7.4.1.3.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/data/export1-metadata.json

The export metadata could look like this:

```
{
  "SchemaId": "1234-abcde-12345-abcde-12345",
  "data_location":
  [
    {
      "filepath": "/FOCUS/data/export1/export1-part1.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export1/export1-part2.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export1/export1-part3.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export1/export1-part4.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    }
  ]
}
```

New metadata can be provided at a location such as /FOCUS/data/export2-metadata.json.

The new export metadata could look like this:

```

{
  "SchemaId": "23456-abcde-23456-abcde-23456",
  "data_location":
  [
    {
      "filepath": "/FOCUS/data/export2/export2-part1.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export2/export2-part2.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export2/export2-part3.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    },
    {
      "filepath": "/FOCUS/data/export2/export2-part4.csv",
      "total_bytes": 9010387,
      "total_rows": 4450
    }
  ]
}

```

#### 7.4.1.4. Adding New Columns

##### 7.4.1.4.1. Scenario

ACME has decided add additional columns to their FOCUS data export. The new columns are x\_awesome\_column1, x\_awesome\_column2, and x\_awesome\_column3. The provider creates a new [Schema](#) object to represent the new schema, this schema object has a unique [SchemaId](#). The subsequent data exports that use the new schema include the new schema's id as a reference to their corresponding schema object.

##### 7.4.1.4.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-23456-abcde-23456-abcde-23456.json.

The updated schema related metadata could look like this:

```

{
  "SchemaId": "23456-abcde-23456-abcde-23456",
  "FocusVersion": "1.0",
  "CreationDate": "2024-02-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",

```

```

        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    },
    {
        "ColumnName": "BillingAccountName",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    },
    {
        "ColumnName": "ChargePeriodStart",
        "DataType": "DATETIME"
    },
    {
        "ColumnName": "ChargePeriodEnd",
        "DataType": "DATETIME"
    },
    {
        "ColumnName": "BilledCost",
        "DataType": "DECIMAL",
        "NumericPrecision": 20,
        "NumberScale": 10
    },
    {
        "ColumnName": "EffectiveCost",
        "DataType": "DECIMAL",
        "NumericPrecision": 20,
        "NumberScale": 10
    },
    {
        "ColumnName": "Tags",
        "DataType": "JSON",
        "ProviderTagPrefixes": ["awecorp", "ac"]
    },
    {
        "ColumnName": "x_awesome_column1",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    },
    {
        "ColumnName": "x_awesome_column2",
        "DataType": "DATETIME"
    },
    {
        "ColumnName": "x_awesome_column3",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    }
}
]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema](#)

#### **7.4.1.5. Removing Columns**

##### **7.4.1.5.1. Scenario**

ACME has decided to remove columns from their FOCUS data export. The column removed is x\_awesome\_column3. The provider creates a new [Schema](#) object to represent the new schema, with a unique [Schemald](#).

##### **7.4.1.5.2. Supplied Metadata**

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-34567-abcde-34567-abcde-34567.json.

The updated schema related metadata could look like this:

```

{
  "SchemaId": "34567-abcde-34567-abcde-34567",
  "FocusVersion": "1.0",
  "CreationDate": "2024-03-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
      "ColumnName": "x_awesome_column1",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "x_awesome_column2",
      "DataType": "DATETIME"
    }
  ]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema Metadata to FOCUS Data Reference](#)

#### **7.4.1.6. Changing Column Metadata**

##### **7.4.1.6.1. Scenario**

ACME has decided to change the datatype of column x\_awesome\_column1 from a string to a number. ACME creates a new [Schema](#) object with the modification to x\_awesome\_column2.

##### **7.4.1.6.2. Supplied Metadata**

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-67891-abcde-67891-abcde-67891.json.

The updated schema related metadata could look like this:



```

{
  "SchemaId": "67891-abcde-67891-abcde-67891",
  "FocusVersion": "1.0",
  "CreationDate": "2024-06-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
      "ColumnName": "x_awesome_column1",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "x_awesome_column2",
      "DataType": "DATETIME"
    }
  ]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema Metadata to FOCUS Data Reference](#)

#### 7.4.1.7. Provider Metadata Error Correction

##### 7.4.1.7.1. Scenario

ACME has discovered that while their export includes the column x\_awesome\_column3, the [Schema](#) metadata does not include this column. In this case, the provider fixes the metadata in the existing schema object and does not need to create a new schema object. Reference metadata remains the same.

##### 7.4.1.7.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-34567-abcde-34567-abcde-34567.json.

The updated schema related metadata could look like this:

```
{
  "SchemaId": "34567-abcde-34567-abcde-34567",
  "FocusVersion": "1.0",
  "CreationDate": "2024-03-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20
    }
  ]
}
```

```

        "numericPrecision": 20,
        "NumberScale": 10
    },
    {
        "ColumnName": "Tags",
        "DataType": "JSON",
        "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
        "ColumnName": "x_awesome_column1",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    },
    {
        "ColumnName": "x_awesome_column2",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    }
]
}

```

#### 7.4.1.8. FOCUS Version Changed

##### 7.4.1.8.1. Scenario

ACME's previous exports used FOCUS version 1.0. They are now going to adopt FOCUS version 1.1. It is required that they create a new schema metadata object which specifies the new FOCUS version via the [FOCUS Version](#) property - regardless of schema changes. In this example, the new FOCUS version adoption doesn't include columns changes. This is to illustrate that FOCUS version changes are independent of column changes, however, this scenario is unlikely.

##### 7.4.1.8.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-45678-abcde-45678-abcde-45678.json.

The updated schema related metadata could look like this:

```

{
  "SchemaId": "45678-abcde-45678-abcde-45678",
  "FocusVersion": "1.1",
  "CreationDate": "2024-04-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
      "ColumnName": "x_awesome_column1",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "x_awesome_column2",
      "DataType": "DATETIME"
    }
  ]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema Metadata to FOCUS Data Reference](#)

#### 7.4.1.9. FOCUS Version Changed by Provider Using Provider Version

##### 7.4.1.9.1. Scenario

ACME specifies the optional metadata property [Provider Version](#) in their [Schema](#) object. Their provider version 2.2 supported FOCUS version 1.0. They are now going to adopt FOCUS Version 1.1 which requires that they update their Provider Version when updating the FOCUS Version. They create a new schema object designating that both properties have changed. In this example, the adoption of the new FOCUS version doesn't include additional columns. This is to illustrate that Provider Version can change independent of column changes; however, this scenario is unlikely.

The provider creates a new schema object to represent the new schema. The provider includes both the new FOCUS Version and Provider Version in the schema object.

##### 7.4.1.9.2. Supplied Metadata

Metadata can be provided at a location such as /FOCUS/metadata/schemas/schema-45678-abcde-45678-abcde-45678.json.

The updated schema related metadata could look like this:

```
{
  "SchemaId": "45678-abcde-45678-abcde-45678",
  "FocusVersion": "1.1",
  "ProviderVersion": "2.3",
  "name": "New Columns",
  "CreationDate": "2024-04-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
```

```

        "DataType": "DECIMAL",
        "NumericPrecision": 20,
        "NumberScale": 10
    },
    {
        "ColumnName": "EffectiveCost",
        "DataType": "DECIMAL",
        "NumericPrecision": 20,
        "NumberScale": 10
    },
    {
        "ColumnName": "Tags",
        "DataType": "JSON",
        "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
        "ColumnName": "x_awesome_column1",
        "DataType": "STRING",
        "StringMaxLength": 64,
        "StringEncoding": "UTF-8"
    },
    {
        "ColumnName": "x_awesome_column2",
        "DataType": "DATETIME"
    }
]
}

```

For reference, the prior schema object looked like this:

```

{
  "SchemaId": "34567-abcde-34567-abcde-34567",
  "FocusVersion": "1.0",
  "ProviderVersion": "2.2",
  "CreationDate": "2024-04-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
      "ColumnName": "x_awesome_column1",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "x_awesome_column2",
      "DataType": "DATETIME"
    }
  ]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema Metadata to FOCUS Data Reference](#)

#### **7.4.1.10. Data Changed by Provider Using Provider Version**

##### **7.4.1.10.1. Scenario**

ACME specifies the optional metadata property [Provider Version](#) in their [Schema](#) object. They made a change to the [FOCUS dataset](#) they produce that does not adopt a new FOCUS Version, nor make a change the included columns but does impact values in the data. This example illustrates that Provider Version changes are independent of column changes, however provider version changes may include column changes.

The provider creates a new schema object to represent the new schema. The provider includes both the FOCUS Version and Provider Version in the schema object.

##### **7.4.1.10.2. Supplied Metadata**

Metadata can be provided at a location such as `/FOCUS/metadata/schemas/schema-56789-abcde-56789-abcde-56789.json`.

The updated schema related metadata could look like this:



```

{
  "SchemaId": "56789-abcde-56789-abcde-56789",
  "FocusVersion": "1.1",
  "ProviderVersion": "2.4",
  "CreationDate": "2024-05-02T12:01:03.083z",
  "ColumnDefinition": [
    {
      "ColumnName": "BillingAccountId",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "BillingAccountName",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "ChargePeriodStart",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "ChargePeriodEnd",
      "DataType": "DATETIME"
    },
    {
      "ColumnName": "BilledCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "EffectiveCost",
      "DataType": "DECIMAL",
      "NumericPrecision": 20,
      "NumberScale": 10
    },
    {
      "ColumnName": "Tags",
      "DataType": "JSON",
      "ProviderTagPrefixes": ["acme", "ac"]
    },
    {
      "ColumnName": "x_awesome_column1",
      "DataType": "STRING",
      "StringMaxLength": 64,
      "StringEncoding": "UTF-8"
    },
    {
      "ColumnName": "x_awesome_column2",
      "DataType": "DATETIME"
    }
  ]
}

```

For an example of how ACME ensures the schema metadata reference requirement is met see: [Schema Metadata to FOCUS Data Reference](#)