Get consumption data for an Azure subscription

Updated: August 4, 2015

Enables you to query aggregate Azure subscription consumption data by:

- Start and end date/time
- Aggregation granularity (ie: daily, hourly)
- Instance level detail (ie: for multiple instances of the same resource)

Important

The metadata associated with the billing meters, including but not limited to service names, types, resources, units of measure, and regions, is subject to change at any time and withou automated fashion, please use the billing meter GUID to uniquely identify each billable item. If the billing meter GUID is scheduled to change due to a new billing model, you will be nc

Request

See the **Common parameters and headers** section in Resource Usage (Preview) for headers and parameters that are used by all requests related to the Resource Usage API.

Method	Request URI
GET	`https://management.azure.com/subscriptions/{subscription-Id}/providers/Microsoft.Commerce/UsageAggregates?api-version={api-version}&rep value}&reportedEndTime={dateTimeOffset-value}&aggregationGranularity={granularity-value}&showDetails={showdetail-boolean-Value}&continu`

- Set {dateTimeOffset-value} for reportedStartTime and reportedEndTime to valid dateTime values. Please note that this dateTimeOffset value represents the timestamp at which billing system. As Azure is a distributed system, spanning across 19 datacenters around the world, there is bound to be a delay between the resource usage time (when the res usage reported time (when the usage event reached the billing system) and callers need a predictable way to get all usage events for a subscription for a given time period. Fo Time to ensure that they get all the usage events reported within a specific time period within the billing system. Even though the query is made with the Reported Time, the u time, which is the useful pivot for callers. To learn more, please see the FAQ section.

  Important

  Please note that the dateTime value format must be URL encoded as ISO 8601 format, and non-numeric characters must use escape codes (i.e. colon is escaped to %3a, plus These refer to the start and end time ranges of your query. This dateTime parameter must also be specified in Universal Time Coordinated (UTC).

- Set {aggregationGranularity} to either 'Daily' or 'Hourly'. This is an optional parameter with two discrete potential values: Daily and Hourly. As the values suggest, the former on latter one is hourly resolution. Daily is the default.
- Set {showDetails} to either true or false. This is an optional Boolean flag, which can be configured to specify whether the caller wants instance-level details with the usage data. false, there will be fewer aggregates returned by the API, as the service will do server-side aggregation. For example, if you are running 3 websites (web1, web2, web3), by defa If you specify showDetails = false, then we will do further aggregation and provide a single line item for website consumption for that particular time period. Setting this flag tc subscriptionId, meterId, usageStartTime and usageEndTime to be single data aggregate only.
- Set {continuationToken} to the continuation token string as retrieved from the response body in the previous call. This is the bookmark for progress when you are working with with continuation tokens, and this parameter expects the value of the token retrieved from the last call to Usage API provider. If not present, the data is retrieved from the begi passed in. The recommended way to page through response is to follow the next link in the response.

There is no request body.

Response

Sample Responses

Below are examples of HTTP responses, which include the request URL and associated substitution values for query parameters with proper encoding for dateTime parameter value response body.

`https://management.azure.com/subscriptions/{subscription-Id}/providers/Microsoft.Commerce/UsageAggregates?api-version=2015-06-01-preview&reporte 01T00%3a00%3a00%2b00%3a00&reportedEndTime=2015-06-01T00%3a00%3a00%2b00%3a00&aggregationGranularity=Daily&showDetails=false`

JSON

```
{
    "value": [
        {
            "id": "/subscriptions/f68815e6-3c41-45ef-bbd8-5f83303c396b/providers/Microsoft.Commerce/UsageAggregates/Daily_BRSDF_20140501_000
            "name": "Daily_BRSDF_20140501_0000",
            "type": "Microsoft.Commerce/UsageAggregate",
            "properties": {
                "subscriptionId": "f68815e6-3c41-45ef-bbd8-5f83303c396b",
                "usageStartTime": "2015-03-03T00:00:00+00:00",
                "usageEndTime": "2015-03-04T00:00:00+00:00",
                "meterName": "Standard IO – Page Blob/Disk (GB)",
                "meterCategory": "Storage",
                "meterSubCategory": "Geo Redundant",
                "unit": "GB",
                "meterId": "0e9d0c9b-ab6d-4312-9c7e-3794e22af9c4",
                "infoFields": {
                },
                "quantity": 0.057865
            }
        },
…
```

The following example is the same as the one above, but with **showDetails=true**, which provides additional details for the infoFields element :

```
https://management.azure.com/subscriptions/{subscription-Id}/providers/Microsoft.Commerce/UsageAggregates?api-version=2015-06-01-preview&reporte
01T00%3a00%3a00%2b00%3a00&reportedEndTime=2015-06-01T00%3a00%3a00%2b00%3a00&aggregationGranularity=Daily&showDetails=true
```

JSON

```
{
    "value": [
        {
            "id": "/subscriptions/f68815e6-3c41-45ef-bbd8-5f83303c396b/providers/Microsoft.Commerce/UsageAggregates/Daily_BRSDT_20140501_000
            "name": "Daily_BRSDT_20140501_0000",
            "type": "Microsoft.Commerce/UsageAggregate",
            "properties": {
                "subscriptionId": "f68815e6-3c41-45ef-bbd8-5f83303c396b",
                "usageStartTime": "2015-03-03T00:00:00+00:00",
                "usageEndTime": "2015-03-04T00:00:00+00:00",
                "meterName": "Standard IO – Page Blob/Disk (GB)",
                "meterCategory": "Storage",
                "meterSubCategory": "Geo Redundant",
                "unit": "GB",
                "meterId": "0e9d0c9b-ab6d-4312-9c7e-3794e22af9c4",
                "infoFields": {
                    "meteredRegion": "West US",
                    "meteredService": "Storage",
                    "project": "devtestvhdsd37a7bb567f9"
                },
                "quantity": 0.057865
            }
        },
…
```

Below is a sample showing the new response format, which implements the “instanceData” element, providing instance-level details including resource tags and the resource URI. T
current “infoFields” element. Currently, we are in a state of transition between the old version and the new version, and we expect all Azure resources to move over to the new form

JSON

```
{
  "value": [
        {
      "id": "/subscriptions/d657c399-e17c-405d-859e-9f2efb6462e5/providers/Microsoft.Commerce/UsageAggregates/Daily_BRSDT_20150515_0000",
      "name": "Daily_BRSDT_20150515_0000",
      "type": "Microsoft.Commerce/UsageAggregate",
      "properties": {
        "subscriptionId": "d657c399-e17c-405d-859e-9f2efb6462e5",
        "usageStartTime": "2015-05-15T00:00:00+00:00",
        "usageEndTime": "2015-05-16T00:00:00+00:00",
        "instanceData": "{\"Microsoft.Resources\":{\"resourceUri\":\"/subscriptions/d657c399-e17c-405d-859e-
9f2efb6462e5/resourceGroups/moinakrg/providers/Microsoft.Storage/storageAccounts/moinakstorage\",\"location\":\"West US\",\"tags\":{\"depart
        "meterName": "Storage Transactions (in 10,000s)",
        "meterCategory": "Data Management",
        "unit": "10,000s",
        "meterId": "964c283a-83a3-4dd4-8baf-59511998fe8b",
        "infoFields": {

        },
        "quantity": 9.8390
      }
    },
    {
      "id": "/subscriptions/d657c399-e17c-405d-859e-9f2efb6462e5/providers/Microsoft.Commerce/UsageAggregates/Daily_BRSDT_20150515_0000",
      "name": "Daily_BRSDT_20150515_0000",
      "type": "Microsoft.Commerce/UsageAggregate",
      "properties": {
        "subscriptionId": "d657c399-e17c-405d-859e-9f2efb6462e5",
        "usageStartTime": "2015-05-15T00:00:00+00:00",
          "usageEndTime": "2015-05-16T00:00:00+00:00",
          "instanceData": "{\"Microsoft.Resources\":{\"resourceUri\":\"/subscriptions/d657c399-e17c-405d-859e-
9f2efb6462e5/resourceGroups/moinakrg/providers/Microsoft.Storage/storageAccounts/moinakstorage\",\"location\":\"West US\",\"tags\":{\"depart
          "meterName": "Data Transfer In (GB)",
          "meterRegion": "Zone 1",
          "meterCategory": "Networking",
          "unit": "GB",
          "meterId": "32c3ebec-1646-49e3-8127-2cafbd3a04d8",
          "infoFields": {

          },
          "quantity": 0.000066
        }
      },
  …
```

JSON Element Definitions

Below is the list of possible JSON data elements you will find in the HTTP response body.

Element Name	Description
Id	Unique Id for the usage aggregate.
name	Name of the usage aggregate.
subscriptionId	The subscription identifier for the Azure user.
meterId	Unique ID for the resource that was consumed (aka ResourceID).
usageStartTime	UTC start time for the usage bucket to which this usage aggregate belongs.
usageEndTime	UTC end time for the usage bucket to which this usage aggregate belongs.
Quantity	The amount of the resource consumption that occurred in this time frame.
Unit	The unit in which the usage for this resource is being counted, e.g. Hours, GB.
meterName	Friendly name of the resource being consumed.
meterCategory	Category of the consumed resource.
meterSubCategory	Sub-category of the consumed resource.
meterRegion	Region of the meterId used for billing purposes.
infoFields	Key-value pairs of instance details (**legacy format**). This field captures the key value pairs to store the instance level details in the legacy format. The most important item here is 'Project' because this carries th we spun up a website and called it "azuretestvm1", the string "azuretestvm1" would show up as a value for Project for the website events for this website. Fo fields, please take a dependence on the higher level resource metadata (meterName, meterCategory, meterSubCategory, unit, meterRegion).
instanceData	Key-value pairs of instance details (**new format**. See the FAQ section below for more details on the difference between the current infoFields design and the • resourceUri: This is the fully qualified resource ID, which includes the resource groups and the instance name. • tags: Contains the resource tags specified by the user. • location: The region in which this service was run. • additionalInfo: More details about the resource being consumed. For example, OS version, Image Type. • partNumber: Unique namespace used to identify the resource for Azure Marketplace 3rd party usage • orderNumber: Unique ID that represents the 3rd party order identifier. Presence of an orderNumber states that this usage record was incurred on a r

Response Codes

HTTP Status Code	Error Code	Description
200/OK	n/a	Normal response for a successful query. The response body will contain the data that matches the filters specified in the query p
202/Accepted	ProcessingNotCompleted	The data requested has not yet been processed. Please try again in {0} minutes. This error is thrown when the data being requested is not ready to be consumed by the user yet, as it has not yet been fully proc representing the number of minutes the user should wait before trying the call again.

400/BadRequest	InvalidInput	Error Message: Header {0} was missing or had an unacceptable value. This is a generic error message for an invalid or missing header. {0} will be replaced with the header name.
400/BadRequest	InvalidInput	Error Message: Parameter {0} was missing or had an unacceptable value. This is a generic error message for an invalid or missing parameter. {0} will be replaced with the parameter name.
400/BadRequest	InvalidInput	Error Message: {0} cannot be in the future. Occurs when a requested time parameter is in the future. {0} will be replaced with the time parameter that was invalid (reported
400/BadRequest	InvalidInput	Error Message: reportedStartTime must be earlier than reportedEndTime. Occurs when the requested reportedStartDateTime is not chronologically earlier than reportedEndDateTime.
400/BadRequest	InvalidInput	Error Message: The {0} for daily aggregation granularity must have the time set to midnight (0:00:00Z). This error is thrown when the requested time parameter is not on an exact day (midnight) when daily granularity is requested. {0 not on an exact day (reportedStartTime or reportedEndDateTime)
400/BadRequest	InvalidInput	Error Message: The {0} for hourly aggregation granularity needs to have the time set using only the hours portion, with zeroes fo This error is thrown when the requested time parameter is not on an exact hour when hourly granularity is requested. {0} will be an exact hour (reportedStartTime or reportedEndDateTime)
401/Unauthorized	AuthorizationError	Error Message: The HTTP request was forbidden with client authentication scheme 'Anonymous'. This will be displayed when the user is not authorized to view the content. Please see the main Azure Billing REST API Reference your calls, and obtaining and specifying a secure access token.
500/Internal Server Error	UnknownError	Error Message: An unknown error has occurred. Reference #: {0} This will be displayed as a generic error message when a better one does not exist. {0} will be a reference number that can be us

Frequently Asked Questions

Question	Answer
What is the difference between Usage Time and Reported Time?	There are 2 different time concepts involved in understanding consumption data: - **Reported DateTime**: This is the timestamp at which the Azure usage event was recorded in the billing system. This is the ti particular usage event made it into a billing cycle, or whether it was dropped due to lateness. - **Usage DateTime**: This is the timestamp of the actual consumption of the Azure resource. At the time of this writing, Azure has 19 Data Centers around the world where customers can run their workloads. Because of this c Usage DateTime and the Reported DateTime for a particular usage event. Although the Azure service providers will move to report (every hour), that is not the case today, and there can be multiple hours of delay between the usage DateTime and the reported Da **Important** Currently, we allow callers to query by reported DateTime ONLY.
Why not just query by Usage Date?	As there can be delay in when the usage events make it into the commerce system (and hence available through this Azure Usage incomplete information to callers. For example, if a caller queried for usage date of 4/2 on 4/3 for particular resource, he would probably get X units back (X is sum o but if he made the same query on 4/10, there is a chance that there could be a different number (say Y). If callers took that as dete date 4/2, they would have incomplete information. That is why, querying by reported DateTime is deterministic, in that callers can query every hour and deterministically get all the us previous hour, and then keep shifting the reported DateTime parameters forward. The response of the usage events is always aggr query by reported Date time, you can get a very good idea of when the particular resources were actually consumed.
What does Azure do with late usage? Does it throw it away?	Azure does discard late usage, but the lateness policies are different for different offer types: **Typically, a 24-hour lateness period is allowed, but usage that comes in the grace period is billed in the following month.** - Usage for billing period 4/12 – 5/11 reported within the commerce system by 5/12 00:00:00 UTC: Goes on the 5/11 bill to th - Usage for billing period 4/12 – 5/11 reported within the commerce system between 5/12 00:00:00 and 5/13 00:00:00- charg **Usage is reported within the allowed lateness period but the usage reported within the grace period is moved to the** - Usage for billing period 4/12 – 5/11 reported within the commerce system after 5/13 00:00:00 – usage discarded and not cc **allowed lateness period, so discarded by the billing system.** Enterprise Agreement (EA) customers have a different behavior- the EA system waits for 5 days to receive all the usage data before subscriptions is otherwise similar except they have a lateness period of 5 days after the end of the month (1/1 – 1/31, usage accept

What is the recommended frequency to make calls to the Usage API?	Here is the basic guidance: - **If you query every hour or every day**: you should query every hour/day by reported DateTime and store the data locally a as volatile data as this can change with late, incoming usage. - **If you only query at the end of the billing cycle**: your usage query should be different for different offer types with report aggregated by usage DateTime, you should discard the usage aggregations that don't fall into the billing cycle that you are
What is the difference between the format in the current 'infoFields' JSON element in the response body, and the future 'instanceData' field?	For each usage aggregate, you will either see infoFields or instanceData. infoFields captures only a subset of the information provic resource tags). We are working to move all Azure service providers to the new format to capture and push through more attributes use instanceData to provide all instance details for the related usage events, which will also support resource groups and tags

See Also

Resource RateCard (Preview)
Azure Billing Usage and RateCard APIs Overview