

# Package ‘AnVILBilling’

October 14, 2021

**Title** Provide functions to retrieve and report on usage expenses in NHGRI AnVIL ([anvilproject.org](http://anvilproject.org)).

**Date** 2020-09-30

**Version** 1.2.0

**Description** AnVILBilling helps monitor AnVIL-related costs in R, using queries to a BigQuery table to which costs are exported daily. Functions are defined to help categorize tasks and associated expenditures, and to visualize and explore expense profiles over time. This package will be expanded to help users estimate costs for specific task sets.

**License** Artistic-2.0

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 4.1)

**Imports** methods, DT, shiny, bigrquery, shinytoastr, DBI, magrittr, dplyr, lubridate, plotly, ggplot2

**Suggests** testthat, knitr, BiocStyle

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**biocViews** Infrastructure, Software

**BugReports** <https://github.com/vjcitn/AnVILBilling/issues>

**git\_url** <https://git.bioconductor.org/packages/AnVILBilling>

**git\_branch** RELEASE\_3\_13

**git\_last\_commit** c339a21

**git\_last\_commit\_date** 2021-05-19

**Date/Publication** 2021-10-14

**Author** BJ Stubbs [aut],  
Vince Carey [aut, cre]

**Maintainer** Vince Carey <[stvjc@channing.harvard.edu](mailto:stvjc@channing.harvard.edu)>

**R topics documented:**

ab_reckoning . . . . .	2
browse_reck . . . . .	3
demo_rec . . . . .	3
getBilling . . . . .	4
getKeys . . . . .	5
getSkus . . . . .	5
getValues . . . . .	6
reckon . . . . .	6
reckoning . . . . .	7
reckoning,avReckoning-method . . . . .	7
setup_billing_request . . . . .	8
subsetByKeyValue . . . . .	8
subsetBySku . . . . .	9

<b>Index</b>	<b>10</b>
--------------	-----------

---

ab_reckoning	<i>accessor for reckoning element</i>
--------------	---------------------------------------

---

**Description**

accessor for reckoning element

**Usage**

```
ab_reckoning(x)
```

**Arguments**

x                    an instance of avReckoning

**Value**

a tibble with one row for each expense type by time slice

**Examples**

```
dim(ab_reckoning(demo_rec))
```

---

browse_reck	<i>prototypical cost exploring app</i>
-------------	--

---

**Description**

prototypical cost exploring app

**Usage**

```
browse_reck()
```

**Value**

returns "NULL"

**Examples**

```
if (interactive()) browse_reck()
```

---

demo_rec	<i>a demonstration avReckoning object</i>
----------	---

---

**Description**

a demonstration avReckoning object

**Usage**

```
demo_rec
```

**Format**

avReckoning instance

**Note**

This is a snapshot of cost data collected for a specific project.

**Examples**

```
demo_rec
```

---

getBilling	<i>request billing data</i>
------------	-----------------------------

---

**Description**

request billing data

**Usage**

```
getBilling(  
  startDate,  
  endDate,  
  bqProject,  
  bqDataset,  
  bqTable,  
  bqBilling_code,  
  page_size = 50000  
)
```

**Arguments**

startDate	character(1) date of start of reckoning
endDate	character(1) date of end of reckoning
bqProject	character(1) GCP project id
bqDataset	character(1) GCP dataset id for billing data in BQ
bqTable	character(1) GCP table for billing data in BQ
bqBilling_code	character(1) GCP billing code
page_size	numeric(1) passed to dbConnect

**Value**

tbl\_df

**Note**

On 21 August 2020 VJC changed condition on endDate to <=

**Examples**

```
if (interactive()) {  
  getBilling(startDate="2020-08-01",  
            endDate="2020-08-15", bqProject="bjbilling",  
            bqTable="gcp_billing_export_v1_015E39_38569D_3CC771",  
            bqDataset="anvilbilling", bqBilling_code="landmarkanvil2")  
}
```

---

getKeys	<i>return keys</i>
---------	--------------------

---

**Description**

return keys

**Usage**

getKeys(mybilling)

**Arguments**

mybilling      tbl\_df

**Value**

character()

---

getSkus	<i>List the available GCP product skus</i>
---------	--

---

**Description**

List the available GCP product skus

**Usage**

getSkus(mybilling)

**Arguments**

mybilling      tbl\_df

**Value**

character()

---

getValues                      *deal with nested tables in a reckoning*

---

**Description**

deal with nested tables in a reckoning

**Usage**

```
getValues(mybilling, mykey)
```

**Arguments**

mybilling	tbl_df from reckon()
mykey	character(1) key

**Value**

character()

**Examples**

```
if (interactive()) getValues(reckoning(demo_rec), "security")
```

---

reckon                      *perform reckoning*

---

**Description**

perform reckoning

**Usage**

```
reckon(obj)
```

**Arguments**

obj	instance of avReckoningRequest
-----	--------------------------------

**Value**

instance of avReckoning

**Examples**

```
data(demo_rec)
if (interactive()) reckon(demo_rec)
```

---

reckoning	<i>generic for accessor for reckoning component</i>
-----------	---

---

**Description**

generic for accessor for reckoning component

**Usage**

```
reckoning(x)
```

**Arguments**

x                    object inheriting from avReckoning

**Value**

tbl\_df

**Examples**

```
if (interactive()) reckoning(reckon(demo_rec))
```

---

reckoning, avReckoning-method	<i>accessor for reckoning component</i>
-------------------------------	---

---

**Description**

accessor for reckoning component

**Usage**

```
## S4 method for signature 'avReckoning'  
reckoning(x)
```

**Arguments**

x                    instance of avReckoning

**Value**

tbl\_df

**Examples**

```
if (interactive()) reckoning(reckon(demo_rec))
```

---

```
setup_billing_request set up request object
```

---

**Description**

set up request object

**Usage**

```
setup_billing_request(start, end, project, dataset, table, billing_code)
```

**Arguments**

start	character(1) date of start of reckoning
end	character(1) date of end of reckoning
project	character(1) GCP project id
dataset	character(1) GCP dataset id for billing data in BQ
table	character(1) GCP table for billing data in BQ
billing_code	character(1) GCP billing code

**Value**

instance of avReckoningRequest

**Examples**

```
lk1 = setup_billing_request("2020-08-01", "2020-08-15",
  "bq_scoped_project", "bq_dataset", "bq_table", "billcode")
lk1
```

---

```
subsetByKeyValue filter a reckoning by 'label' retaining records associated with a particular key-value pair
```

---

**Description**

filter a reckoning by 'label' retaining records associated with a particular key-value pair

**Usage**

```
subsetByKeyValue(mybilling, mykey, myvalue)
```

**Arguments**

mybilling      instance of avReckoning  
mykey          character(1)  
myvalue        character(1)

**Value**

data.frame

**Examples**

```
example(reckon) # makes rec  
v = getValues(ab_reckoning(demo_rec), "terra-submission-id")[1] # for instance  
nt = subsetByKeyValue(ab_reckoning(demo_rec), "terra-submission-id", v)  
head(nt)  
dim(nt)
```

---

subsetBySku                      *subset a billing object by sku*

---

**Description**

subset a billing object by sku

**Usage**

```
subsetBySku(mybilling, mysku)
```

**Arguments**

mybilling      tbl\_df  
mysku          character(1) GCP product sku

**Value**

data.frame

# Index

## \* datasets

demo\_rec, 3

ab\_reckoning, 2

browse\_reck, 3

demo\_rec, 3

getBilling, 4

getKeys, 5

getSkus, 5

getValues, 6

reckon, 6

reckoning, 7

reckoning, avReckoning-method, 7

setup\_billing\_request, 8

subsetByKeyValue, 8

subsetBySku, 9