HW1 for Data Mining

Problem 1

example of a document-term matrix

	database	SQL	index	regression	likelihood	linear
d1	24	21	9	0	0	3
d2	32	10	5	0	3	0
d3	12	16	5	0	0	0
d4	6	7	2	0	0	0
d5	43	31	20	0	3	0
d6	2	0	0	18	7	16
d7	0	0	1	32	12	0
d8	3	0	0	22	4	2
d9	1	0	0	34	27	25
d10	6	0	0	17	4	23

In a doc-term matrix, as we can see an example, the M(i,j) represents the the term in jth column occurs in the ith doc. Since the zero entries means the term doesnot appear in a specific doc, and its importance is not comparable to its counter part. So, this is a typical asymmetric discrete feature. On the other hand, the way of making a doc-term matrix to be continuous is through so called tf-idf normalization. As the equation shown below:

TF-IDF normalization

- Normalize the term weights
  - so longer docs not given more weight (fairness)
  - force all values to fall within a certain range: [0, 1]

$$w_{ik} = \frac{tf_{ik} \log(N / n_k)}{\sqrt{\sum_{k=1}^{t} (tf_{ik})^2 [\log(N / n_k)]^2}}$$

Based on slides by P. Raghavan, H. Schütze, R. Larson

However, tf-idf normalized features will keep asymmetric, since the whole term remains 0, if tf(i,k) is 0. So, the zero term remains meaningful.

Problem 2

1)

Terms appear in every doc will get 0, since log(1) = 0. Terms appear in only one doc will get the max value, e.g., log(m). m is the size of the docs.

2)

This transformation will cancel out the terms that occurs in every document, since those terms are not important. While the terms appear rarely have more importance.

Problem 3

In [1]:

```
library(readr)
sales <- read_csv("sales.csv")
```

```
Parsed with column specification:
cols(
  ID = col_integer(),
  year = col_integer(),
  mo = col_integer(),
  month = col_character(),
  quarter = col_character(),
  st = col_character(),
  state = col_character(),
  country = col_character(),
  product = col_character(),
  category = col_character(),
  unit_price = col_integer(),
  unit = col_integer(),
  amount = col_integer()
)
```

In [89]:

```
sales <- as.data.frame(sales)
head(sales)
```

ID	year	mo	month	quarter	st	state	country	product	category	unit_price
1	2013	7	Jul	Q3	ON	Ontario	Canada	Tablet	Electronic	500
2	2013	6	Jun	Q2	WA	Washington	USA	Chair	Furniture	120
3	2013	4	Apr	Q2	NY	New York	USA	Mattress	Furniture	800
4	2013	7	Jul	Q3	WA	Washington	USA	Laptop	Electronic	1000
5	2013	8	Aug	Q3	QU	Quebec	Canada	Chair	Furniture	120
6	2013	10	Oct	Q4	NY	New York	USA	Printer	Electronic	200

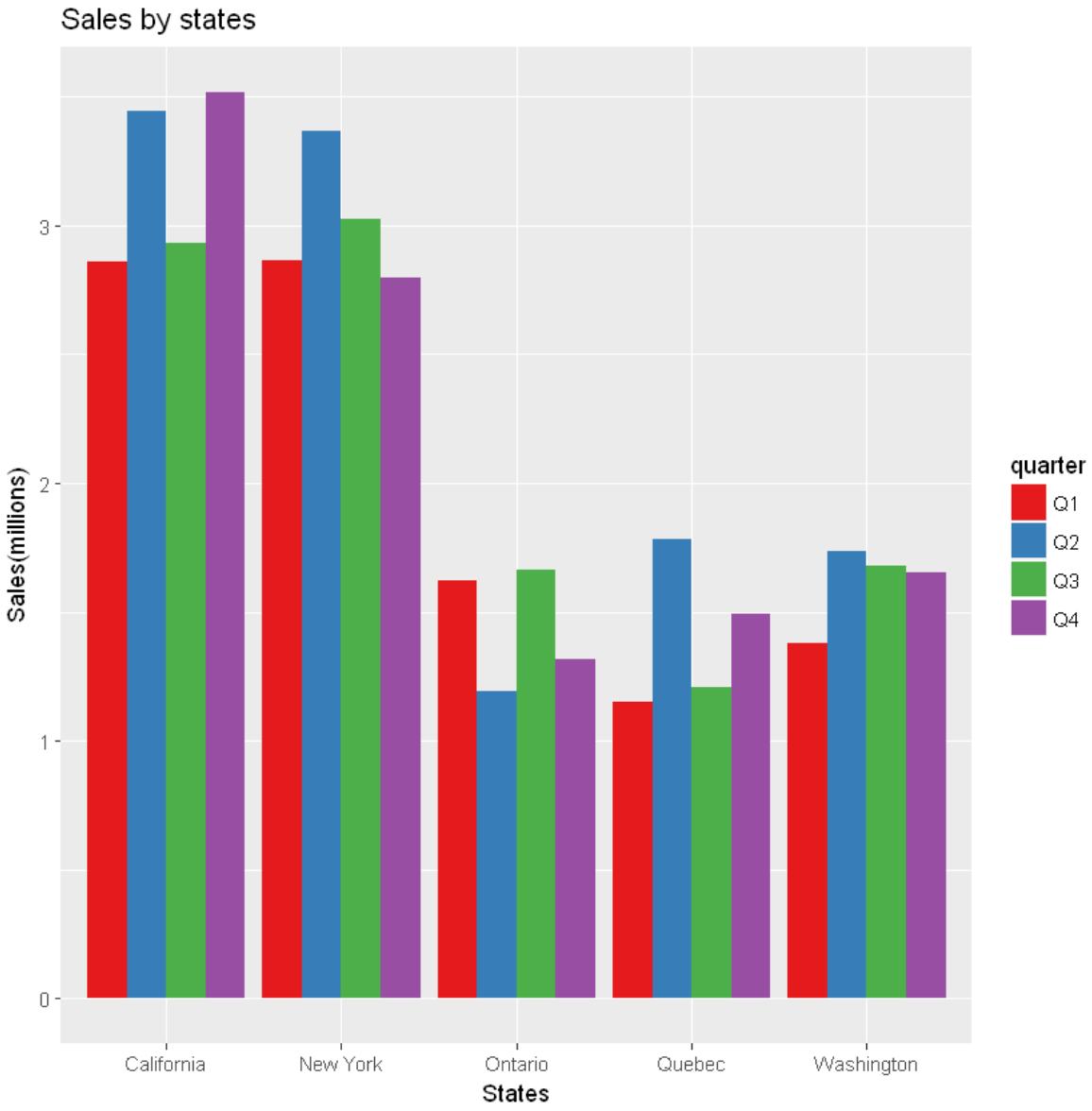
In [62]:

```
library(plyr)
# group the original dataset usint multiple column name, here we use state and quarter
dataSummayByStates = ddply(sales,.(state, quarter),numcolwise(sum))
keeps <- c("state","quarter", "amount")
dataSummayByStates <- dataSummayByStates[keeps]
dataSummayByStates
```

state	quarter	amount
California	Q1	2858120
California	Q2	3444980
California	Q3	2935020
California	Q4	3518500
New York	Q1	2862880
New York	Q2	3370020
New York	Q3	3027120
New York	Q4	2796860
Ontario	Q1	1621560
Ontario	Q2	1191220
Ontario	Q3	1664920
Ontario	Q4	1315340
Quebec	Q1	1148520
Quebec	Q2	1781880
Quebec	Q3	1207940
Quebec	Q4	1493440
Washington	Q1	1378720
Washington	Q2	1735540
Washington	Q3	1679640
Washington	Q4	1654780

In [66]:

```
library(ggplot2)
ggplot(dataSummayByStates, aes(factor(state), amount/1000000, fill = quarter)) +
       geom_bar(stat="identity", position = "dodge") +
       scale_fill_brewer(palette = "Set1") +
       xlab("States") +
       ylab("Sales(millions)")+
       ggtitle("Sales by states")
```



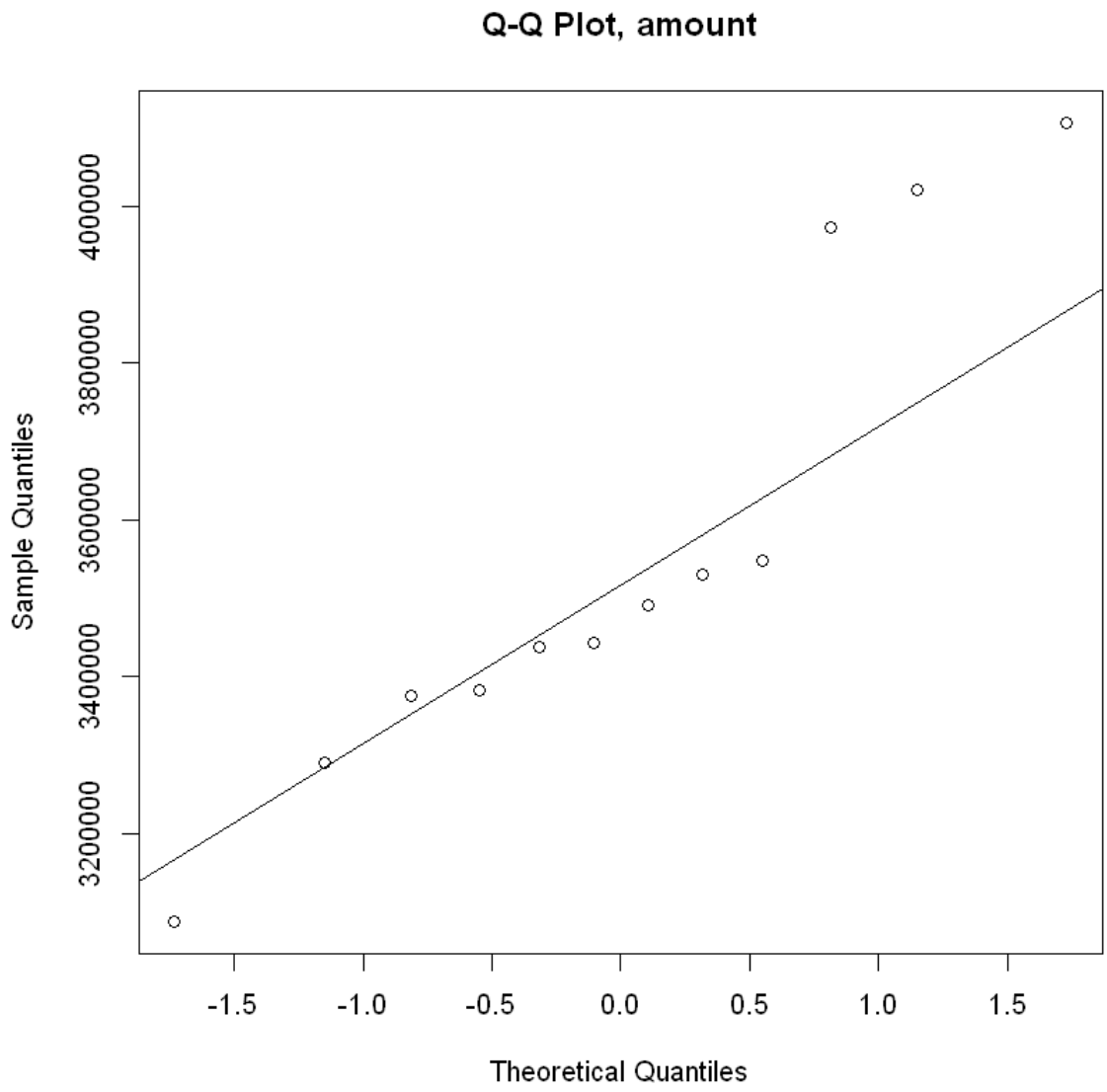
In [33]:

```
dataSummary = ddply(sales,.(month),numcolwise(sum))
dataSummary
```

month	ID	year	mo	unit_price	unit	amount
Apr	378538	521500	1036	106940	8250	3443900
Aug	395416	473175	1880	111240	7521	3547320
Dec	383058	511431	3048	104640	8215	3383040
Feb	326987	461084	458	94780	7405	3086980
Jan	338941	459076	228	101700	7390	3290880
Jul	368096	505392	1757	101280	8494	3437900
Jun	407271	535585	1596	117520	8814	4106920
Mar	396855	549692	819	107820	8937	3491940
May	389354	513441	1275	116020	8529	3972820
Nov	335051	473178	2585	102000	7766	3374660
Oct	397270	527538	2620	112800	9233	4021220
Sep	384663	509426	2277	105580	8540	3529420

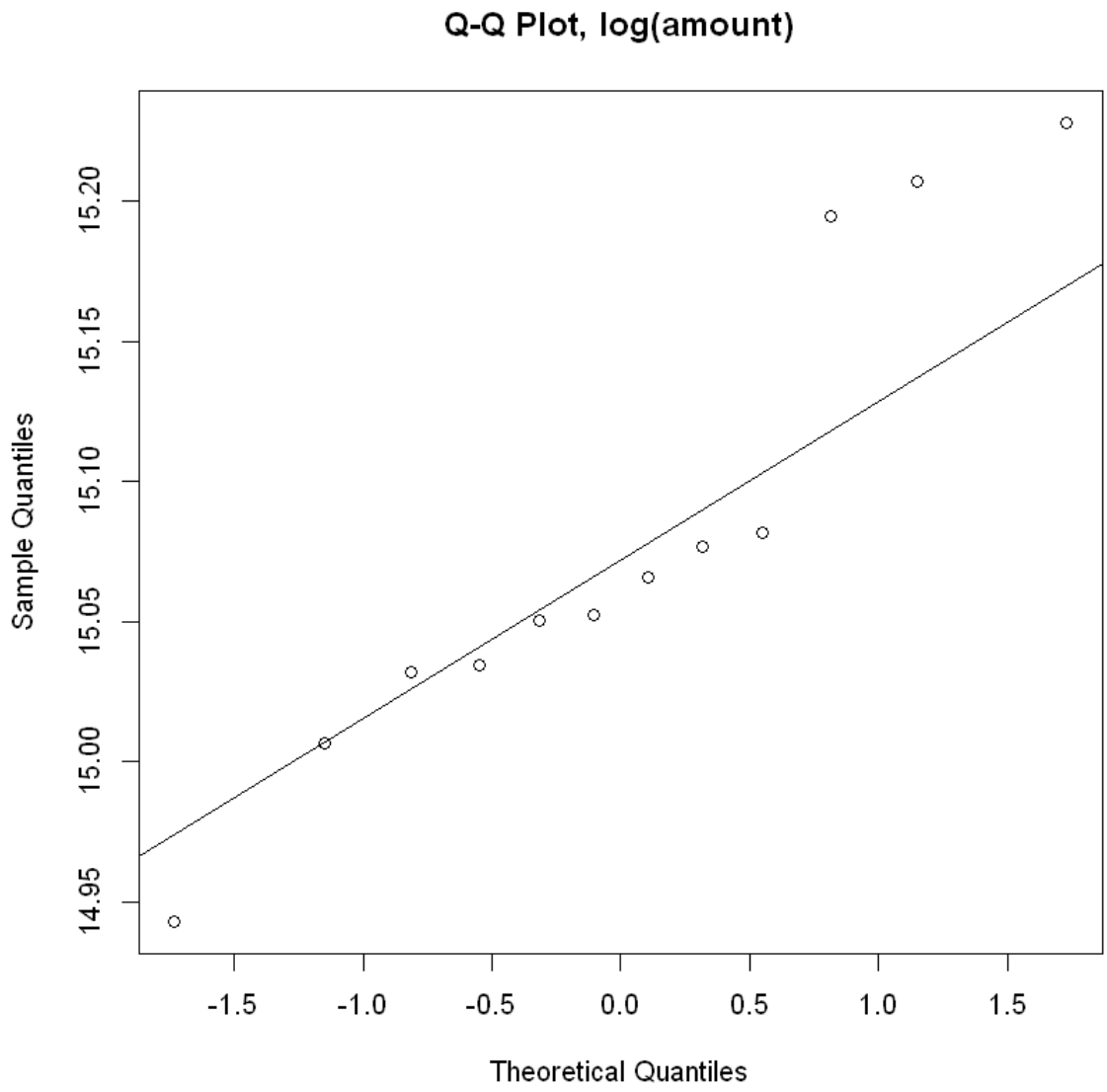
In [35]:

```
qqnorm(dataSummary$amount,main = "Q-Q Plot, amount")
qqline(dataSummary$amount)
```



In [39]:

```
qqnorm(log(dataSummary$amount),main = "Q-Q Plot, log(amount)")
qqline(log(dataSummary$amount))
```

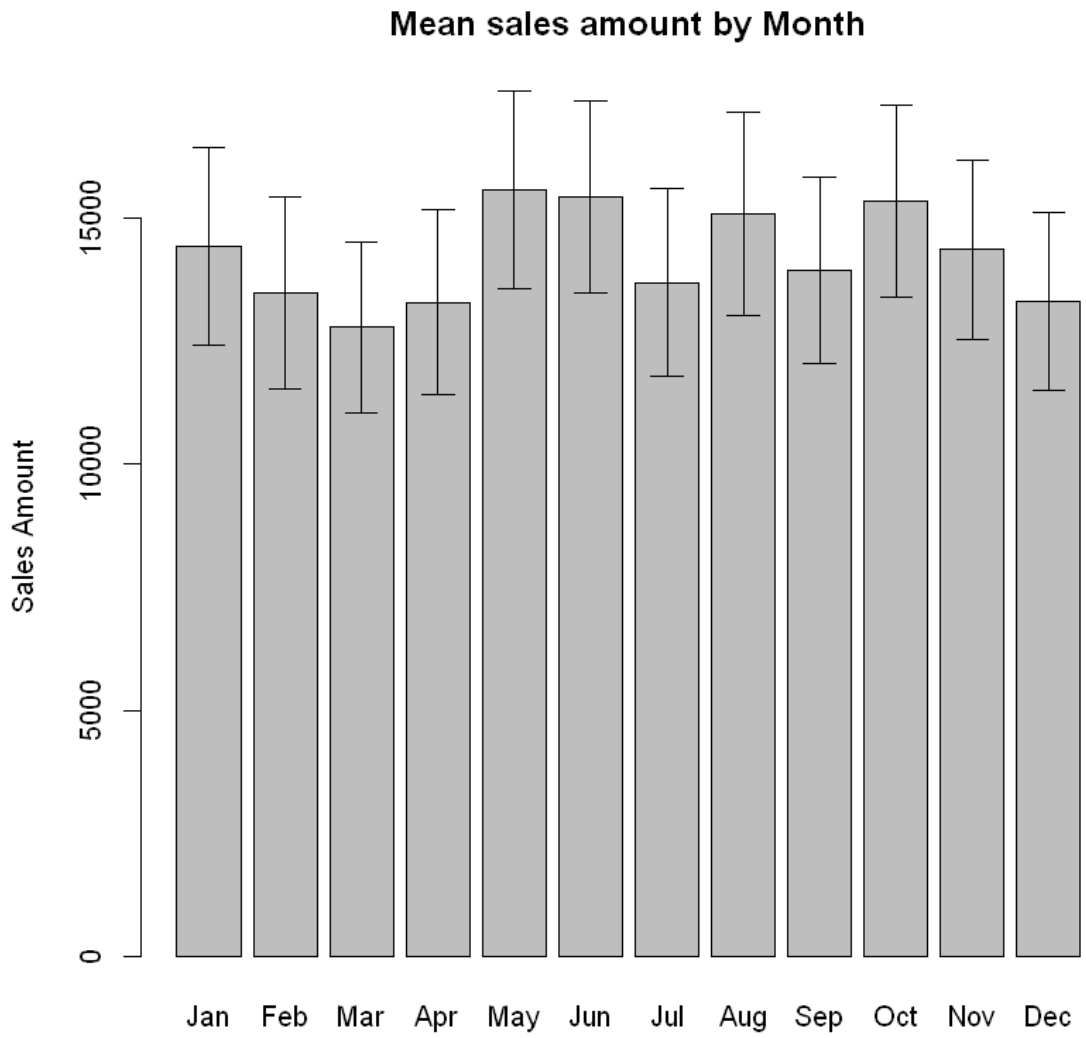


In [21]:

```
heights <- tapply(sales$amount, sales$mo, mean)
# add Confidence interval to data
lower <-tapply(sales$amount, sales$mo, function(v) t.test(v)$conf.int[1])
upper <-tapply(sales$amount, sales$mo, function(v) t.test(v)$conf.int[2])
```

In [29]:

```
library(gplots)
barplot2(heights,
        plot.ci = TRUE, ci.l = lower, ci.u = upper, xpd = FALSE,
        main = "Mean sales amount by Month",
        names.arg = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct",
        col=c("grey"),
        ylab = "Sales Amount")
```



Problem 4

In [88]:

```
head(sales)
```

ID	year	mo	month	quarter	st	state	country	product	category	unit_price
1	2013	7	Jul	Q3	ON	Ontario	Canada	Tablet	Electronic	500
2	2013	6	Jun	Q2	WA	Washington	USA	Chair	Furniture	120
3	2013	4	Apr	Q2	NY	New York	USA	Mattress	Furniture	800
4	2013	7	Jul	Q3	WA	Washington	USA	Laptop	Electronic	1000
5	2013	8	Aug	Q3	QU	Quebec	Canada	Chair	Furniture	120
6	2013	10	Oct	Q4	NY	New York	USA	Printer	Electronic	200

In [152]:

```
keep <- c("year","month", "state", "product",  "amount")
sales_revenue <- sales[keep]
sales_revenue <- as.data.frame(sales_revenue)
head(sales_revenue)
```

year	month	state	product	amount
2013	Jul	Ontario	Tablet	21000
2013	Jun	Washington	Chair	1440
2013	Apr	New York	Mattress	32800
2013	Jul	Washington	Laptop	27000
2013	Aug	Quebec	Chair	2760
2013	Oct	New York	Printer	5600

In [160]:

```
revenue_cube <-
    tapply(sales_revenue$amount,
           sales_revenue[,c("year", "month", "state", "product"),],
           FUN=function(x){return(sum(x))})
dimnames(revenue_cube)
```

$year

'2013' '2014'

$month

'Apr' 'Aug' 'Dec' 'Feb' 'Jan' 'Jul' 'Jun' 'Mar' 'May' 'Nov' 'Oct' 'Sep'

$state

'California' 'New York' 'Ontario' 'Quebec' 'Washington'

$product

'Chair' 'Laptop' 'Mattress' 'Printer' 'Tablet'

Slice

In [154]:

```
#Slice operation: compute the revenue for Laptop during January of 2013 in each state
revenue_cube['2013','Jan',,'Laptop']
```

California
130000
New York
153000
Ontario
39000
Quebec
62000
Washington
57000

Dice

In [ ]:

```
#Dice operation: compute the revenue for the furniture products (Mattress and Chair) during
# (April, May and June) of 2014 in each state.
# the following line could be correctly displayed on RStudio as follows, but not on Jupyter
revenue_cube['2014',c('Apr','May','Jun'),,c('Chair','Mattress')]
```

```
, , product = Chair

     state
month California New York Ontario Quebec Washington
  Apr      56160    56040   21960  25920      29400
  May      62640    64320   25200  39480      36960
  Jun      57960    43560   15480  18480      12720

, , product = Mattress

     state
month California New York Ontario Quebec Washington
  Apr      56000    47200      NA 127200         NA
  May      60000    15200      NA   2400      51200
  Jun     149600   111200      NA  59200     128800
```

Rollup

In [150]:

```
# Rollup operation: compute the annual revenue for each product and collapse the state and
apply(revenue_cube, c("year", "product"),
      FUN=function(x) {return(sum(x, na.rm=TRUE))})
```

	Chair	Laptop	Mattress	Printer	Tablet
2013	2412240	8109000	3420000	786600	6158500
2014	2382960	8435000	3358400	872800	6751500

Drilldown

In [ ]:

```
# Drilldown operation: compute the annual and monthly revenue for each product and collapse
apply(revenue_cube, c("year", "month", "product"),
      FUN=function(x) {return(sum(x, na.rm=TRUE))})
```

In [ ]:

```
, , product = Chair

      month
year      Apr    Aug    Dec    Feb    Jan    Jul    Jun    Mar    May    Nov    Oct    Sep
  2013 286320 125640 223920 159000 192960 204840 198720 237960 117120 137400 269160 259200
  2014 189480 204480 201120 201480 144720 237360 148200 247680 228600 174360 180960 224520

, , product = Laptop

      month
year      Apr    Aug    Dec    Feb    Jan    Jul    Jun    Mar     May    Nov    Oct    Sep
  2013 801000 895000 581000 558000 441000 614000 836000 622000 1087000 419000 827000 428000
  2014 648000 755000 623000 440000 638000 593000 896000 689000  581000 723000 892000 957000

, , product = Mattress

      month
year      Apr    Aug    Dec    Feb    Jan    Jul    Jun    Mar    May    Nov    Oct    Sep
  2013 280000 201600 374400 502400 443200 201600 148800 276800 425600 216800 212800 136000
  2014 230400 307200 243200 204800 312800 195200 448800 178400 128800 233600 490400 384800

, , product = Printer

      month
year     Apr   Aug   Dec   Feb   Jan   Jul    Jun    Mar   May    Nov    Oct   Sep
  2013 29600 67000 58200 93400 43400 35800  74800  71000 67600 100800 127400 17600
  2014 30600 55400 83200 65400 61800 71600 119600 105600 91600  73200  68000 46800

, , product = Tablet

      month
year      Apr    Aug    Dec    Feb    Jan    Jul    Jun    Mar    May    Nov    Oct    Sep
  2013 463500 455000 382500 414500 429500 720000 664500 492500 590500 595500 379000 571500
  2014 485000 481000 612500 448000 583500 564500 571500 571000 655000 701000 574500 504000
```