Модели и технологии оперативного анализа данных

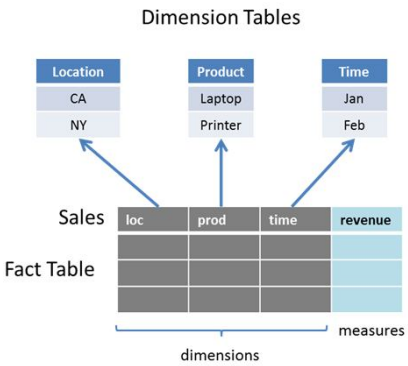
Лекция 5
OLAP операции в R

Гедранович Ольга Брониславовна,
старший преподаватель кафедры ИТ, МИУ
volha.b.k@gmail.com

Вопросы лекции

- Facts and Dimensions tables
- Multi-dimensional Cube
- OLAP operations: Slice, Dice, Roll-Up, Drill-down, Pivot

Facts and Dimensions tables



In a typical setting of Multi-dimensional model:

- Each fact table contains foreign keys that reference the primary key of multiple dimension tables. In the simplest form it is called a STAR schema.
- Dimension tables can contain foreign keys that reference other dimensional tables. This provides a sophisticated detail breakdown of the contextual aspects. This is also called a SNOWFLAKE schema.
- Also this is not a hard rule, Fact table tends to be independent of other Fact table and usually doesn't contain reference pointer among each other.
- However, different Fact table usually share the same set of dimension tables. This is also called GALAXY schema.
- But it is a hard rule that Dimension table NEVER points / references Fact table.

Facts and Dimensions tables

```
state_table <- data.frame(key = c("CA", "NY", "WA", "ON", "QU"),
                          name = c("California", "New York",
                         "Washington", "Ontario", "Quebec"),
                          country = c("USA", "USA", "USA", "Canada",
                         "Canada"))
row.names(state_table) <- state_table$key


month_table <- data.frame(key = 1:12,
                          desc = c("Jan", "Feb", "Mar", "Apr", "May",
                          "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
                          "Dec"),
                          quarter = c("Q1", "Q1", "Q1", "Q2", "Q2",
                          "Q2", "Q3", "Q3", "Q3", "Q4", "Q4", "Q4"))
row.names(month_table) <- month_table$key


prod_table <- data.frame(key = c("Printer", "Tablet", "Laptop"),
                         price = c(225, 570, 1120))
row.names(prod_table) <- prod_table$key
```

Multi-dimensional Cube

```
# Function to generate the Sales table
gen_sales <- function(no_of_recs) {

# Generate transaction data randomly
loc <- sample(state_table$key,
no_of_recs, replace = TRUE, prob = c(2,
2, 1, 1, 1))

country <- state_table[loc, ]$country

time_month <- sample(month_table$key,
no_of_recs, replace = TRUE)

time_q <- month_table[time_month,
]$quarter

time_year <- sample(c(2012, 2013),
no_of_recs, replace = TRUE)

prod <- sample(prod_table$key,
no_of_recs, replace = TRUE, prob = c(1,
3, 2))

unit <- sample(c(1, 2), no_of_recs,
replace = TRUE, prob = c(10, 3))

amount <- unit * prod_table[prod, ]$price

sales <- data_frame(month = time_month,
                    quarter = time_q,
                    year = time_year,
                    loc = loc,
                    country = country,
                    prod = prod,
                    unit = unit,
                    amount = amount)

# Sort the records by time order
sales <- sales[order(sales$year,
sales$month), ]

sales
}
```


Multi-dimensional Cube

```
# Create the sales table
sales_fact <- gen_sales(500)

# Build up a cube
revenue_cube <- tapply(sales_fact$amount,
                sales_fact[, c("prod", "month",
                "year", "country")],
                FUN = function(x)
                {return(sum(x))} )

# Showing the cells of the cube
revenue_cube

# Showing the cube's dimensions
dimnames(revenue_cube)
```

Slice

```
revenue_cube[ , "1", "2012", ]

revenue_cube["Tablet", "1", "2012", ]
```

The slice operation selects particular dimension(s) from a given cube and provides a new sub-cube.

"Slice" is about fixing certain dimensions to analyze the remaining dimensions.

Dice

```
revenue_cube[c("Tablet", "Laptop"),
             c("1", "2", "3"),
             ,
             "USA"]
```

The dice operation limits dimensions' values and provides a new sub-cube.

"Dice" is about limiting each dimension to a certain range of values while keeping the number of dimensions the same in the resulting sub-cube.

Roll-up

```
apply(revenue_cube, c("year", "prod"),
      FUN = function(x) {return(sum(x,
      na.rm = TRUE))} )
```

The roll-up operation performs aggregation on a data cube in any of the following ways:

- by climbing up a concept hierarchy for a dimension;
- by dimension reduction.

Drill-down

```
revenue_cube1 <-
  tapply(sales_fact$amount,
         sales_fact[, c("prod", "month", "year", "loc")],
         FUN = function(x) {return(sum(x))} )

revenue_cube2 <-
  tapply(sales_fact$amount,
         sales_fact[, c("prod", "month", "year",
        "country", "loc")],
         FUN = function(x) {return(sum(x))} )
```

Drill-down is the reverse operation of roll-up. It is performed by either of the following ways:

- by stepping down a concept hierarchy for a dimension;
- by introducing a new dimension.

Pivot

```
apply(revenue_cube, c("prod", "country"),
      FUN = function(x)
      {return(sum(x, na.rm = TRUE))} ) %>%
  t()


revenue_cube[ , "2", "2012", ] %>%
  t()
```

The pivot operation is also known as rotation.

It rotates the data axes in view in order to provide an alternative presentation of data.

Основная литература

- OLAP-системы [Electronic resource] / TAdviser. – Mode of access: http://www.tadviser.ru/index.php/Статья:OLAP-системы. – Date of access: 20.01.2017.
- Ho, R. OLAP operation in R [Electronic resource] / Pragmatic Programming Techniques. – Mode of access: http://horicky.blogspot.com.by/2013/07/olap-operation-in-r.html. – Date of access: 20.01.2017.