Partitioning Mutate, Example 3

John Mount, Win-Vector LLC

2017-12-28

This third article shows our second example being processed by the `rquery` big data scale relational data operator system (currently in development).

We will repeat the steps from Partitioning Mutate, Example 2, using only the `rquery` package and `DBI`/`sparklyr` (no `dplyr`).

```
library("wrapr")
library("rquery")

class(sc)
```

```
## [1] "spark_connection"
## [2] "spark_shell_connection"
## [3] "DBIConnection"
```

```
class(d)
```

```
## [1] "relop_table_source" "relop"
```

```
d %.>%
  to_sql(., sc) %.>%
  DBI::dbGetQuery(sc, .) %.>%
  knitr::kable(.)
```

rowNum	a_1	a_2	b_1	b_2	c_1	c_2	d_1	d_2	e_1	e_2
1	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
3	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
4	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
5	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

```
dQ <- d %.>%
  extend_se(.,
            if_else_block(
              testexpr =
                "rand()>=0.5",
              thenexprs = qae(
                a_1 := 'treatment',
                a_2 := 'control'),
              elseexprs = qae(
                a_1 := 'control',
```

```
                a_2 := 'treatment'))) %.>%
  select_columns(., c("rowNum", "a_1", "a_2"))

cat(format(dQ))

table('d') %.>%
 extend(.,
  ifebtest_1 := rand() >= 0.5) %.>%
 extend(.,
  a_1 := ifelse(ifebtest_1, "treatment", a_1),
  a_2 := ifelse(ifebtest_1, "control", a_2)) %.>%
 extend(.,
  a_1 := ifelse(!( ifebtest_1 ), "control", a_1),
  a_2 := ifelse(!( ifebtest_1 ), "treatment", a_2)) %.>%
 select_columns(., rowNum, a_1, a_2)
```

Notice the `rquery` `extend_se` command accepts the `if_else_block` *and* partitions it into conflict-free segments. Also the `rquery` presentation lets the user inspect the operation plan before attempting execution.

```
sql <- to_sql(dQ, sc)
DBI::dbGetQuery(sc, sql) %.>%
  knitr::kable(.)
```

rowNum	a_1	a_2
1	control	treatment
2	control	treatment
3	control	treatment
4	control	treatment
5	treatment	control

The underlying `SQL` is fairly involved, but can be performant at big-data scale.

```
cat(sql)

SELECT
 `rowNum`,
 `a_1`,
 `a_2`
FROM (
 SELECT
  `ifebtest_1`,
  `rowNum`,
```

```
  ( CASE WHEN ( ( NOT ( `ifebtest_1` ) ) ) THEN ( "control" ) ELSE ( `a_1` ) END )  AS `a_1`,
  ( CASE WHEN ( ( NOT ( `ifebtest_1` ) ) ) THEN ( "treatment" ) ELSE ( `a_2` ) END )  AS `a_2`
 FROM (
  SELECT
   `ifebtest_1`,
   `rowNum`,
   ( CASE WHEN ( `ifebtest_1` ) THEN ( "treatment" ) ELSE ( `a_1` ) END )  AS `a_1`,
   ( CASE WHEN ( `ifebtest_1` ) THEN ( "control" ) ELSE ( `a_2` ) END )  AS `a_2`
  FROM (
   SELECT
    `rowNum`,
    `a_1`,
    `a_2`,
    rand ( ) >= 0.5  AS `ifebtest_1`
   FROM (
    SELECT
     `d`.`rowNum`,
     `d`.`a_1`,
     `d`.`a_2`
    FROM
     `d`
    ) tsql_0000
   ) tsql_0001
  ) tsql_0002
) tsql_0003
```

Links

Win-Vector LLC supplies a number of open-source `R` packages for working effectively with big data. These include:

- **wrapr**: supplies code re-writing tools that make coding *over* "non standard evaluation" interfaces (such as `dplyr`) *much* easier.
- **cdata**: supplies pivot/un-pivot functionality at big data scale.
- **rquery**: (in development) big data scale relational data operators.
- **seplyr**: supplies improved interfaces for many data manipulation tasks.
- **replyr**: supplies tools and patches for using `dplyr` on big data.

Partitioning mutate articles:

- **Partitioning Mutate**: basic example.
- **Partitioning Mutate, Example 2**: `ifelse` example.
- **Partitioning Mutate, Example 3** `rquery` example.

Topics such as the above are often discussed on the Win-Vector blog.