Session 2a: Oracle R Enterprise 1.5.1 OREdplyr

Oracle R Technologies

Mark Hornick
Director, Advanced Analytics and Machine Learning
mark.hornick@oracle.com

October 2018

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Agenda

1. What is dplyr?
2. Functionality of OREdplyr
3. Examples using OREdplyr

What is dplyr?

What is dplyr?

- A grammar for data manipulation
- An R package that provides fast, consistent tool for working with data frame like objects, both in memory and out of memory
- Operates on data.frame or numeric vector objects
- Widely used package that also interfaces to database management systems
- https://cran.r-project.org/web/packages/dplyr/index.html

- dplyr + Oracle Database via Oracle R Enterprise…

OREdplyr

- A subset of dplyr functionality extending ORE transparency layer
- Use ore.frames instead of data.frames for in-database execution
- Avoid costly movement of data
- Scale to larger data volumes since not constrained by R Client memory

Functionality of OREdplyr

OREdplyr functions in ORE 1.5.1

- OREdplyr functionality maps closely to CRAN dplyr package, e.g., function and args
- OREdplyr operates on ore.frame or ore.numeric objects
- Functions support non-standard evaluation (NSE) and standard evaluation (SE) interface
  - Difference noted with a _ at the end of function name, e.g.,
    - NSE → select, filter, arrange, mutate, transmute
    - SE → select_, filter_, arrange_, mutate_, transmute_
  - NSE interface is good for interactive use while SE ones are convenient for programming
  - See https://cran.r-project.org/web/packages/dplyr/vignettes/programming.html for details

OREdplyr functions by category

- Data manipulation
  - select, filter, arrange, rename, mutate, transmute, distinct, slice, desc, select_, filter_, arrange_, rename_, mutate_, transmute_ , distinct_, slice_, inner_join, left_join, right_join, full_join
- Grouping
  - group_by, groups, ungroup, group_size, n_groups, group_by_
- Aggregation
  - summarise, summarise_, tally, count, count_
- Sampling
  - sample_n, sample_frac
- Ranking
  - row_number, min_rank, dense_rank, percent_rank, cume_dist, ntile, nth, first, last, n_distinct, top_n

Examples using OREdplyr

Content adapted from original dplyr vignettes (e.g., link)

Examples: basic operations

```
library(OREdplyr)

library(nycflights13)    # contains data sets

# Import data to Oracle Database

ore.drop("FLIGHTS") # remove database table, if exists
# create table from data.frame
ore.create(as.data.frame(flights), table="FLIGHTS")

dim(FLIGHTS)     # get # rows and # columns
names(FLIGHTS)   # view names of columns
head(FLIGHTS)    # verify data.frame appears as expected

# Basic operations

select(FLIGHTS, year, month, day, dep_delay, arr_delay)
       %>% head() # select columns
select(FLIGHTS, -year,-month, -day)
       %>% head() # exclude columns
```

```
select(FLIGHTS, tail_num = tailnum)
        %>% head() # rename columns, but drops others
rename(FLIGHTS, tail_num = tailnum)
        %>% head() # rename columns

filter(FLIGHTS, month == 1, day == 1)
        %>% head() # filter rows
filter(FLIGHTS, dep_delay > 240) %>% head()
filter(FLIGHTS, month == 1 | month == 2) %>% head()

arrange(FLIGHTS, year, month, day)
        %>% head() # sort rows by specified columns
arrange(FLIGHTS, desc(arr_delay))
        %>% head() # sort in descending order

distinct(FLIGHTS, tailnum)
         %>% head() # see distinct values
distinct(FLIGHTS, origin, dest)
         %>% head() # see distinct pairs
```

OREdplyr caveats

- ‘:’ not supported for range of column specification, e.g., V1:V10
- Variables cannot be referenced within a mutate() and transmute()
  - Restate computation where needed
- Functions supported for summarise when using grouped ore.frame
  - 'min', 'mean', 'max', 'median', 'length', 'IQR', 'prod', 'sum', 'range', 'quantile', 'fivenum', 'summary', 'sd','var', 'all', 'any‘
- n_distinct()
  - Works with non-grouped ore.frame
  - Not supported for summarise with grouped ore.frame
    - Work around: use dense_rank, top_n, and unique
      # compute number of distinct planes over destination
      destinations %>% transmute(dest, planes = dense_rank(tailnum)) %>% top_n(1) %>% unique
- filter() does not apply non-ranking function per group
- Use ore.pull instead of dplyr collect

Summary

- OREdplyr provides a subset of dplyr functionality working with ore.frames
- Use popular API conveniently with Oracle Database tables
- Avoid costly movement of data
- Scale to larger data volumes since not constrained by R Client memory
- Use Oracle Database as high performance compute engine

To Learn More about Oracle's R Technologies…

http://oracle.com/goto/R

R Technologies from Oracle

Bringing the Power of R to the Enterprise


ORACLE®