INSTYTUT ŁĄCZNOŚCI
PAŃSTWOWY INSTYTUT BADAWCZY

Dostęp do danych w Rweb

Instrukcja dla użytkowników Rweb portalu widok.gov.pl

Wstęp

Ten dokument określa sposób dostępu do tabel bazy danych systemu Widok. Zawarto w nim podstawowe zalecenia oraz krótko scharakteryzowano najczęściej używane funkcje służące do operowania na danych w bazie danych.

Organizacja danych w bazie danych

Dane dostępne dla użytkownika znajdują się w dwóch schematach bazy danych:

1. schemat „IN”
   Zawiera surowe dane wejściowe z Widoku. Schemat i wszystkie jego tabele są dostępne w Rweb tylko do odczytu.
2. schemat „OUT”
   Zawiera wszystkie tabele utworzone przez użytkowników Rweb. Schemat dostępny jest do zapisu dla wszystkich użytkowników. Z tego powodu użytkownik powinien przy nadawaniu nazw tabelom wynikowym upewnić się, że nie nadpisze tabeli utworzonej przez innego użytkownika.
   Zaleca się ustalenie wspólnego dla wszystkich użytkowników schematu nazewnictwa tabel w ramach tego schematu, np. poprzez dodanie przed właściwą nazwa tabeli skrótu w postaci pierwszej litery imienia i pierwszej litery nazwiska i pierwszej spółgłoski pozostałej części nazwiska (przykładowo tabela „WYNIKI” utworzona przez użytkownika Adama Kowalskiego nazywałaby się „AKW_WYNIKI”).
   Uwaga: Długość nazwy tabeli wynikowej nie może przekroczyć 63 znaków!

Połączenie do bazy danych

Dostęp do danych realizowany jest za pomocą pakietów RPostgreSQL oraz DBI. Połączenie do bazy jest zestawiane dla użytkownika funkcją dbConnect w sposób automatyczny przed każdym wysłanym kodem R i jest dostępne dla użytkownika za pośrednictwem zmiennej „conn”. W przypadku błędu połączenia (a więc problemu z dostępem do bazy danych) wartość zmiennej „conn” jest ustawiana na „NULL”. Sprawdzenie, czy połączenie do bazy danych powiodło się można wykonać warunkiem:

```
if (!is.null(conn))
```

Dodatkowe informacje o połączeniu można wyświetlić za pomocą polecenia:

```
summary(conn)
```

Odwołanie się do nazw tabel

We wszystkich operacjach na tabelach w bazie danych zaleca się używanie nazw tabel poprzedzonych nazwą schematu. Robi się to poprzez zdefiniowanie wektora funkcją „c”.

```
nazwa_out = c("OUT","AKW_WYNIKI")
```

Powyższy kod definiuje nazwę tabeli „AKW_WYNIKI” na schemacie „OUT” i zapisuje tak określona pełną nazwę tabeli w zmiennej „nazwa_out”.

Dozwolone jest pominięcie nazwy schematu w przypadku odczytu z schematu „IN” oraz zapisu do schematu „OUT”, gdyż są to domyślne schematy dla tych operacji. W takim przypadku nie jest konieczne używanie funkcji c.

```
nazwa_out = "AKW_WYNIKI"
```

Zarządzanie tabelami w bazie danych

Lista tabel w bazie danych

Do pobrania listy tabel i widoków używa się polecenia dbListTables

```
dbListTables(conn)
```

Sprawdzenie czy tabela istnieje

Do sprawdzenia czy tabela istnieje w bazie danych używa się funkcji dbExistsTable

```
if (dbExistsTable(conn, "AKW_WYNIKI"))
```

Usuwanie tabel

Do usuwania tabel używa się polecenia dbRemoveTable.

```
dbRemoveTable(conn, c("OUT","AKW_WYNIKI"))
```

Zapis danych użytkownika do bazy danych

Zapis ramki danych

Zapis ramki danych do tabeli bazy danych wykonuje się funkcją `dbWriteTable`.

```
dbWriteTable(conn, name = nazwa_out, value = dane,
overwrite = TRUE)
```

Pierwszym parametrem jest omówiona wcześniej zmienna „conn", zawierająca informacje o połączeniu do bazy danych.

Wszystkie pozostałe parametry sugeruje się używać w postaci parametrów nazwanych (występujących w postaci „`nazwa = wartość`"):

- `name` – nazwa tabeli w bazie danych,
- `value` – ramka danych do zapisania jako tabela,
- `overwrite` – opcjonalna flaga mówiąca o tym, czy ramka danych powinna nadpisać obecna zawartość tabeli, domyślnie `FALSE`.
- `append` – opcjonalna flaga mówiąca o tym, czy ramka danych powinna dopisać rekordy na końcu obecnej tabeli w bazie danych, domyślnie `FALSE`.

Uwaga: Zapis tabeli w bazie danych możliwy jest wyłącznie na schemacie „OUT"!

Wstawianie i usuwanie rekordów

Wstawianie i usuwanie rekordów odbywa się poprzez wywołanie zapytania SQL funkcją `dbGetQuery`.

Wstawianie rekordów:

```
sql <- "INSERT INTO tabela (kol1, kol2, kol3) VALUES (1, 2, '3')"

dbGetQuery(conn, sql)
```

Usuwanie rekordów:

```
sql <- "DELETE from tabela where kol1=1"

dbGetQuery(conn, sql)
```

Odczyt tabeli z bazy danych

Odczyt tabeli z bazy danych można wykonać na kilka sposobów:

1) Odczyt całej tabeli do ramki danych:

```
tables <- dbReadTable(conn, c("pg_catalog","pg_tables"))
```

2) Wykonanie zapytania na bazie danych i zwrócenie wyniku do ramki danych

```
tables <- dbGetQuery(conn, "SELECT * FROM pg_catalog.pg_tables")
```

3) Użycie obiektów PostgreSQLResult (DBIResult) – za pomocą funkcji **dbSendQuery** wysyła się zapytanie SQL na serwer bazy danych, następnie za pomocą funkcji **fetch** pobiera się wyniki (częściowe lub całe).

   Przykład:

```
Rweb:> rs <- dbSendQuery(conn, "SELECT * FROM information_schema.tables")
Rweb:> dbColumnInfo(rs)
                          name    Sclass    type len precision scale nullOK
1                table_catalog character VARCHAR  -1        -1    -1   TRUE
2                 table_schema character VARCHAR  -1        -1    -1   TRUE
3                   table_name character VARCHAR  -1        -1    -1   TRUE
4                   table_type character VARCHAR  -1        -1    -1   TRUE
5  self_referencing_column_name character VARCHAR  -1        -1    -1   TRUE
6          reference_generation character VARCHAR  -1        -1    -1   TRUE
7     user_defined_type_catalog character VARCHAR  -1        -1    -1   TRUE
8      user_defined_type_schema character VARCHAR  -1        -1    -1   TRUE
9        user_defined_type_name character VARCHAR  -1        -1    -1   TRUE
10           is_insertable_into character VARCHAR  -1         7    -1   TRUE
11                     is_typed character VARCHAR  -1         7    -1   TRUE
12               commit_action character VARCHAR  -1        -1    -1   TRUE
Rweb:> tab.first10 <- fetch(rs, 10)
Rweb:> tab.rest <- fetch(rs, -1)
Rweb:> dbGetRowCount(rs)
[1] 159
Rweb:> summary(rs)
<PostgreSQLResult:(22431,0,1)>
  Statement: SELECT * FROM information_schema.tables
  Has completed? yes
  Affected rows: -1
  Rows fetched: 159
Rweb:> dbClearResult(rs)
[1] TRUE
Rweb:> tables <- rbind(tab.first10, tab.rest)
```

Utworzenie tabeli z danych użytkownika

W Rweb udostępniono 3 mechanizmy przekazywania danych użytkownika do R. Wybór metody przesłania danych dokonuje się za pa pomocą kontrolki typu radio znajdującej się poniżej edytora kodu.

1) Dane – służy do przekazania danych użytkownika za pomocą pola tekstowego
2) Plik – służy do przekazania danych użytkownika poprzez wskazanie pliku z danymi
3) URL- służy do przekazania danych użytkownika poprzez wskazanie adresu URL z danymi użytkownika.

W każdej z trzech metod dane powinny zaczynać się od linii z nazwami kolumn tabeli. Następnie w każdym wierszu powinien znaleźć się jeden rekord danych. Rekord danych składa się z wartości oddzielanych białymi znakami (tabulatory, spacje). Jeżeli w wartości którejś ze zmiennych występuje znak biały, należy taką wartość wpisać w cudzysłowie. Separatorem dziesiętnym dla liczb jest znak kropki (np. 0.0). Linie zaczynające się od znaku „#" są komentarzem i przy wczytywaniu danych są ignorowane. Puste pola powinny być wypełniane wartością „NA". Dane są wczytywane za pomocą funkcji `read.table` z parametrem „`header=T`".

Przykład:

```
user   int    float  bool   other
# komentarz
arek   1      1.1    T      "Dlugi \" tekst"
czarek 2      2.2    F      NA
darek  3      3.3    NA     kro"tki
jarek  4      NA     FALSE  <NA>
marek  NA     5.5    TRUE   'x'
```

Powyższe dane powodują utworzenia ramki danych jak na poniższym listingu:

```
Rweb:> sapply(X, class)
     user       int     float      bool     other
 "factor" "integer" "numeric" "logical"  "factor"
Rweb:> X
    user int float  bool         other
1   arek   1   1.1  TRUE Dlugi " tekst
2 czarek   2   2.2 FALSE          <NA>
3  darek   3   3.3    NA       kro"tki
4  jarek   4    NA FALSE          <NA>
5  marek  NA   5.5  TRUE             x
```

Szczegółowa dokumentacja online

Szczegółową dokumentację funkcji manipulujących na bazie danych można znaleźć pod wskazanymi poniżej adresami:

- https://cran.r-project.org/web/packages/DBI/DBI.pdf
- https://cran.r-project.org/web/packages/RPostgreSQL/RPostgreSQL.pdf