Laboratorio di Basi di Dati

R e le basi di dati

Dario Della Monica

19 dicembre 2018

Outline

Connessione ad una base di dati e esecuzione di query

Utilizzo di R per popolare la base di dati con dati casuali

Analisi statistica dei dati con grafici

Outline

Connessione ad una base di dati e esecuzione di query

Utilizzo di R per popolare la base di dati con dati casuali

Analisi statistica dei dati con grafici

Connessione ad una base di dati

- installare “RPostgreSQL”
  - `install.packages("RPostgreSQL")`

  potrebbe essere necessario installare dei pacchetti per le dipendenze a livello di sistema operativo

- caricare la libreria “RPostgreSQL”
  - `library("RPostgreSQL")`

Connessione ad una base di dati (2)

- caricare il driver PostgreSQL (e memorizzarlo in una variabile)
  - `drv <- dbDriver(‘‘PostgreSQL’’)`

- creare la connessione (e memorizzarla in una variabile)
  - `con <- dbConnect(drv,`
    `dbname="<nome_database>",`
    `host="<indirizzo_IP_dbserver>",`
    `port=<port_number>, # usually 5432`
    `user="<username>",`
    `password="<password>")`

Esecuzione di una query

- con visualizzazione del risultato (dataframe)
  - `dbGetQuery(con, "<SQLquery>")`
- con memorizzazione del risultato in un dataframe
  - `res <- dbGetQuery(con, "<SQLquery>")`
- per visualizzazioni e manipolazioni successive
  (il risultato è una variabile di tipo PostgreSQLResult)
  - `res <- dbSendQuery(con, "<SQLquery>")`
  - `df <- fetch(res, n = 3)`
  - `dbHasCompleted(res)` *# FALSE finché la fine della tabella*
    # è raggiunta

Esecuzione di una query

- con visualizzazione del risultato (dataframe)
  - `dbGetQuery(con, "<SQLquery>")`
- con memorizzazione del risultato in un dataframe
  - `res <- dbGetQuery(con, "<SQLquery>")`
- per visualizzazioni e manipolazioni successive
  (il risultato è una variabile di tipo PostgreSQLResult)
  - `res <- dbSendQuery(con, "<SQLquery>")`
  - `df <- fetch(res, n = 3)`
  - `dbHasCompleted(res)` *# FALSE finché la fine della tabella*
    # è raggiunta
  - `df <- fetch(res, n = 3)`
  - `dbHasCompleted(res)`
  - `...`
  - `dbClearResult(res)`

Disconnessione dalla base di dati

- `dbDisconnect(con)`

Outline

Connessione ad una base di dati e esecuzione di query

Utilizzo di R per popolare la base di dati con dati casuali

Analisi statistica dei dati con grafici

Funzione dbWriteTable

```
dbWriteTable( con,
              name="<table_name>",
              value=<data_frame>,
              append=<T/F>,
              row.names=<T/F>)
```

Popolare la tabella `studenti`

1. scaricare degli elenchi di nomi e cognomi (facile da google)
2. leggere i 2 file: memorizzare in un vettore una stringa per ogni riga
   - `v_nomi<-readLines("./Desktop/lezioniBD/lezione14/nomi.txt")`
   - `v_cognomi<-readLines("./Desktop/lezioniBD/lezione14/cognomi.txt")`
3. creare un dataframe a partire da `v_nomi` e `v_cognomi`
   -
     ```
     studenti_df <- data.frame(
         nome=sample(v_nomi, 10000, replace = T),
         cognome=sample(v_cognomi, 10000, replace = T),
         sesso=sample(c("m","f"), 10000, replace=T))
     ```
4. inserire le righe del dataframe nella tabella del DB con `dbWriteTable`

Popolare la relazione `iscritto_a`

1. creare un vettore di 10000 (numero studenti) elementi presi a caso dall'elenco dei corsi di laurea
   (a) `temp_cdl <- dbGetQuery(con, "select nome from corsi_di_laurea")`
   (b) `temp_cdl <- temp_cdl$nome`
   (c) `iscritto_a.cdl <- sample(temp_cdl, 10000, replace=T)`
2. creare il vettore delle matricole presenti in tabella `studenti`
   (a) `temp_matricola <- dbGetQuery(con, "select matricola from studenti")`
   (b) `iscritto_a.stud <- temp_matricola$matricola`
3. creare un vettore di 10000 interi casuali tra 1978 (fondazione di uniud) e 2018
   (a) `iscritto_a.anno <- sample(1978:2018, 10000, replace=T)`
4. creare dataframe a partire da `iscritto_a.cdl`, `iscritto_a.stud` e `iscritto_a.anno`
5. inserire le righe del dataframe nella tabella del DB con `dbWriteTable`

Popolare la relazione `iscritto_a` (2)

Assumiamo che il 10% degli iscritti si è iscritto a 2 corsi di laurea

6. creare un vettore contenente il 1000 (10% di 10000) corsi di laurea a caso
   (a) `temp_cdl <- dbGetQuery(con, "select nome from corsi_di_laurea")`
   (b) `temp_cdl <- temp_cdl$nome`
   (c) `iscritto_a.cdl <- sample(temp_cdl, 1000, replace=T)`
7. creare un vettore contenente il 10% (1000) delle matricole studenti
   (a) `temp_matricola <- dbGetQuery(con, "select matricola from studenti")`
   (b) `temp_matricola <- temp_matricola$matricola`
   (c) `iscritto_a.stud <- sample(temp_matricola, 1000, replace=F)`
8. creare un vettore di 1000 interi casuali tra 1978 (fondazione di uniud) e 2018
   (a) `iscritto_a.anno <- sample(1978:2018, 1000, replace=T)`
9. creare dataframe a partire da `iscritto_a.cdl`, `iscritto_a.stud` e `iscritto_a.anno`
10. inserire le righe del dataframe nella tabella del DB con `dbWriteTable`

Outline

Visualizzare numero iscritti per anno

1. ottenere i dati dalla base di dati
   - ```
     df <- dbGetQuery(con,
                      "SELECT isc.anno, count(*)
                       FROM iscritto_a as isc
                       GROUP BY isc.anno
                       ORDER BY isc.anno")
     ```
2. visualizzare il grafico con `plot`
   - `plot(df$anno, df$count, "o")`
     ```

Visualizzare numero iscritti per anno e corso di laurea

1. ottenere i dati dalla base di dati
   - ```
     df <- dbGetQuery(con,
                "SELECT isc.anno, cdl.nome, count(*)
                 FROM iscritto_a as isc, corsi_di_laurea as cdl
                 WHERE isc.cdl=cdl.nome
                 GROUP BY isc.anno, cdl.nome
                 ORDER BY isc.anno, cdl.nome")
     ```
2. manipolare i dati per ottenere un dataframe con una colonna per ogni corso di laurea
   - `colonna_anno <- unique(df$anno)`
   - `colonna_info <- df$count[df$nome == "informatica"]`
   - `colonna_infe <- df$count[df$nome == "scienze infermieristiche"]`
   - `colonna_med <- df$count[df$nome == "medicina"]`
   - `colonna_twm <- df$count[df$nome == "twm"]`
   - ```
     df_out <- data.frame(anno = colonna_anno,
                  informatica = colonna_info, medicina = colonna_med,
                  infermieristiche = colonna_infe, twm = colonna_twm)
     ```
3. visualizzare il grafico con `matplot`
   - `matplot(df_out$anno, df_out[-1], type="l")`

Visualizzare grafico a barre degli studenti divisi per sesso e corsi di laura

1. ottenere i dati dalla base di dati
   - ```
     df <- dbGetQuery(con,
         "SELECT stud.sesso, cdl.nome, count(*)
          FROM studenti as stud, corsi_di_laurea as cdl, iscritto_a as isc
          WHERE (stud.matricola = isc.stud) AND (isc.cdl = cdl.nome)
          GROUP BY stud.sesso,cdl.nome")
     ```
2. ri-arrangiare i dati in un formato *barplot-friendly*
   - `df_ordered <- df[order(df$nome,df$sesso),]`
   - `matr <- matrix(df_ordered$count, nrow = 2)`
   - `rownames(matr) <- c("f","m")`
   - `colnames(matr) <- unique(df_ordered$nome)`
3. visualizzare il grafico con `barplot`
   - `barplot(matr)`
   - `barplot(matr, beside=T)`
     ```