

## Permutation tests

Permutation tests are very similar to bootstraps. They test if two distributions are the same:

```
> x=rnorm(10)
> y=rnorm(20)+1
> mean(x)-mean(y)
[1] -0.7245577
>
```

In a permutation test, we permute the data, and ask how often it has such a difference between the samples:

```
> perm=t( sapply( 1:1000, function(i) sample( c(x,y) ) ) )
> perm.diff = apply(perm, 1, function(xy){ mean(xy[1:10])-mean(xy[11:30]) } )
> sum( abs(perm.diff) > abs((mean(x)-mean(y))) )
[1] 36
> 36/1000
[1] 0.036
>
```

So this permutation test sees the data as significantly different

```
> t.test(x,y)$p.value
[1] 0.07868077
>
```

The permutation test tests the hypothesis that the two distributions are the same.

With a bootstrap test, we could do a very similar test, except that we sample **with replacement**.

## Lecture 8 - access to databases

First, let us look a bit more into saving and loading:

We saw how to read a table - with read.table

```
> a=read.table("data/regression.txt",sep=" ",head=T)
>
```

A very similar command allows us to save data to a table:

```
> write.table(a,file="test.txt",sep=" ")
>
```

We also saw before that R can save the session when you quit, so that all the variables are retained.

We can save variables using the save command:

```
> a=1:100
> save(a,file="a.Rdata")
```

```

> rm(a)
> a
Error: Object "a" not found
> load("a.Rdata")
> a
  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 [19] 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
 [37] 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
 [55] 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
 [73] 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90
 [91] 91 92 93 94 95 96 97 98 99 100

```

>

When we save a variable, and then load it, it is loaded with its own name.

We can also save several variables at once:

```

> b=1:3
> save(a,b,file="ab.Rdata")
>

```

or

```

> save(list=c("a","b"),file="ab.Rdata")
>

```

To save everything, we can combine save and ls:

```

> save(list=ls(),file="whole_session.Rdata")
>

```

## Interfacing with an Oracle database

### Connecting to the database

This consists of 3 steps:

```

> library(ROracle)
> ora=dbDriver("Oracle")
> con=dbConnect(ora,user="lachmann",password="alexandria",dbname="TEST_DB")

```

Now we are connected to the database. All our commands will use con.

Let us start with simple things:

```

> data(ToothGrowth)
> ls()
 [1] "a"          "con"          "last.warning" "ora"          "res"
 [6] "ToothGrowth" "USArrests"
>

```

the data() statement (which has nothing to do with databases or Oracle) loads sample data that is provided by some libraries. This is convenient for examples.

We can save a table to the database:

```
> dbWriteTable(con,"hamster",ToothGrowth)
```

```
[1] TRUE
```

```
> a=dbReadTable(con,"hamster")
```

```
> a
```

	LEN	SUPP	DOSE
1	4.2	VC	0.5
2	11.5	VC	0.5
3	7.3	VC	0.5
4	5.8	VC	0.5
5	6.4	VC	0.5
6	10.0	VC	0.5
7	11.2	VC	0.5
8	11.2	VC	0.5
9	5.2	VC	0.5
10	7.0	VC	0.5
11	16.5	VC	1.0
12	16.5	VC	1.0
13	15.2	VC	1.0
14	17.3	VC	1.0
15	22.5	VC	1.0
16	17.3	VC	1.0
17	13.6	VC	1.0
18	14.5	VC	1.0
19	18.8	VC	1.0
20	15.5	VC	1.0
21	23.6	VC	2.0
22	18.5	VC	2.0
23	33.9	VC	2.0
24	25.5	VC	2.0
25	26.4	VC	2.0
26	32.5	VC	2.0
27	26.7	VC	2.0
28	21.5	VC	2.0
29	23.3	VC	2.0
30	29.5	VC	2.0
31	15.2	OJ	0.5
32	21.5	OJ	0.5
33	17.6	OJ	0.5
34	9.7	OJ	0.5
35	14.5	OJ	0.5
36	10.0	OJ	0.5
37	8.2	OJ	0.5
38	9.4	OJ	0.5
39	16.5	OJ	0.5
40	9.7	OJ	0.5
41	19.7	OJ	1.0
42	23.3	OJ	1.0
43	23.6	OJ	1.0
44	26.4	OJ	1.0
45	20.0	OJ	1.0
46	25.2	OJ	1.0

```

47 25.8  OJ  1.0
48 21.2  OJ  1.0
49 14.5  OJ  1.0
50 27.3  OJ  1.0
51 25.5  OJ  2.0
52 26.4  OJ  2.0
53 22.4  OJ  2.0
54 24.5  OJ  2.0
55 24.8  OJ  2.0
56 30.9  OJ  2.0
57 26.4  OJ  2.0
58 27.3  OJ  2.0
59 29.4  OJ  2.0
60 23.0  OJ  2.0

```

>

So, we can load and save. We can also look at what tables are available:

> `dbListTables(con)`

```

 [1] "DUAL"                "SYSTEM_PRIVILEGE_MAP" "TABLE_PRIVILEGE_MAP"
 [4] "STMT_AUDIT_OPTION_MAP" "OL$"                  "OL$HINTS"
 [7] "OL$NODES"           "MAP_OBJECT"          "AUDIT_ACTIONS"
[10] "PSTUBTBL"           "ODCI_SECOBJ$"        "ODCI_WARNINGS$"
[13] "DEF$_TEMP$LOB"      "R_TEST"              "COUNTRIES"
[16] "DEPARTMENTS"        "EMPLOYEES"           "JOBS"
[19] "JOB_HISTORY"        "LOCATIONS"           "REGIONS"
[22] "TEST"               "A2_C3B"              "A_C1A"
[25] "ARRESTS"            "HAMSTER"

```

>

And, we can remove tables:

> `dbRemoveTable(con, "hamster")`

```

Error in oraFetch(rs, n = -1) : RS-DBI driver: (ORA-24374: define not done before
fetch or execute and fetch
)
Error in oraCloseConnection(conn, ...) : pending result sets in the Oracle server --
must close manually
[1] FALSE

```

> `dbListTables(con)`

```

 [1] "DUAL"                "SYSTEM_PRIVILEGE_MAP" "TABLE_PRIVILEGE_MAP"
 [4] "STMT_AUDIT_OPTION_MAP" "OL$"                  "OL$HINTS"
 [7] "OL$NODES"           "MAP_OBJECT"          "AUDIT_ACTIONS"
[10] "PSTUBTBL"           "ODCI_SECOBJ$"        "ODCI_WARNINGS$"
[13] "DEF$_TEMP$LOB"      "R_TEST"              "COUNTRIES"
[16] "DEPARTMENTS"        "EMPLOYEES"           "JOBS"
[19] "JOB_HISTORY"        "LOCATIONS"           "REGIONS"
[22] "TEST"               "A2_C3B"              "A_C1A"
[25] "ARRESTS"

```

>

(I don't know why we get the above error message)

This is already fairly convenient - it gives a convenient way to exchange data.

> a

	LEN	SUPP	DOSE
1	4.2	VC	0.5
2	11.5	VC	0.5
3	7.3	VC	0.5
4	5.8	VC	0.5
5	6.4	VC	0.5
6	10.0	VC	0.5
7	11.2	VC	0.5
8	11.2	VC	0.5
9	5.2	VC	0.5
10	7.0	VC	0.5
11	16.5	VC	1.0
12	16.5	VC	1.0
13	15.2	VC	1.0
14	17.3	VC	1.0
15	22.5	VC	1.0
16	17.3	VC	1.0
17	13.6	VC	1.0
18	14.5	VC	1.0
19	18.8	VC	1.0
20	15.5	VC	1.0
21	23.6	VC	2.0
22	18.5	VC	2.0
23	33.9	VC	2.0
24	25.5	VC	2.0
25	26.4	VC	2.0
26	32.5	VC	2.0
27	26.7	VC	2.0
28	21.5	VC	2.0
29	23.3	VC	2.0
30	29.5	VC	2.0
31	15.2	OJ	0.5
32	21.5	OJ	0.5
33	17.6	OJ	0.5
34	9.7	OJ	0.5
35	14.5	OJ	0.5
36	10.0	OJ	0.5
37	8.2	OJ	0.5
38	9.4	OJ	0.5
39	16.5	OJ	0.5
40	9.7	OJ	0.5
41	19.7	OJ	1.0
42	23.3	OJ	1.0
43	23.6	OJ	1.0
44	26.4	OJ	1.0
45	20.0	OJ	1.0
46	25.2	OJ	1.0
47	25.8	OJ	1.0
48	21.2	OJ	1.0
49	14.5	OJ	1.0
50	27.3	OJ	1.0
51	25.5	OJ	2.0

```

52 26.4  OJ  2.0
53 22.4  OJ  2.0
54 24.5  OJ  2.0
55 24.8  OJ  2.0
56 30.9  OJ  2.0
57 26.4  OJ  2.0
58 27.3  OJ  2.0
59 29.4  OJ  2.0
60 23.0  OJ  2.0

```

```

>
>

```

## Simple queries

One of the strength of interfacing with a database is that we can not just read a table, but format the data as a new table. For example:

```

> a=dbGetQuery(con,"SELECT * from hamster")
> a

```

	ROW_NAMES	LEN	SUPP	DOSE
0	1	4.2	VC	0.5
1	2	11.5	VC	0.5
2	3	7.3	VC	0.5
3	4	5.8	VC	0.5
4	5	6.4	VC	0.5
5	6	10.0	VC	0.5
6	7	11.2	VC	0.5
7	8	11.2	VC	0.5
8	9	5.2	VC	0.5
9	10	7.0	VC	0.5
10	11	16.5	VC	1.0
11	12	16.5	VC	1.0
12	13	15.2	VC	1.0
13	14	17.3	VC	1.0
14	15	22.5	VC	1.0
15	16	17.3	VC	1.0
16	17	13.6	VC	1.0
17	18	14.5	VC	1.0
18	19	18.8	VC	1.0
19	20	15.5	VC	1.0
20	21	23.6	VC	2.0
21	22	18.5	VC	2.0
22	23	33.9	VC	2.0
23	24	25.5	VC	2.0
24	25	26.4	VC	2.0
25	26	32.5	VC	2.0
26	27	26.7	VC	2.0
27	28	21.5	VC	2.0
28	29	23.3	VC	2.0
29	30	29.5	VC	2.0
30	31	15.2	OJ	0.5
31	32	21.5	OJ	0.5
32	33	17.6	OJ	0.5

33	34	9.7	OJ	0.5
34	35	14.5	OJ	0.5
35	36	10.0	OJ	0.5
36	37	8.2	OJ	0.5
37	38	9.4	OJ	0.5
38	39	16.5	OJ	0.5
39	40	9.7	OJ	0.5
40	41	19.7	OJ	1.0
41	42	23.3	OJ	1.0
42	43	23.6	OJ	1.0
43	44	26.4	OJ	1.0
44	45	20.0	OJ	1.0
45	46	25.2	OJ	1.0
46	47	25.8	OJ	1.0
47	48	21.2	OJ	1.0
48	49	14.5	OJ	1.0
49	50	27.3	OJ	1.0
50	51	25.5	OJ	2.0
51	52	26.4	OJ	2.0
52	53	22.4	OJ	2.0
53	54	24.5	OJ	2.0
54	55	24.8	OJ	2.0
55	56	30.9	OJ	2.0
56	57	26.4	OJ	2.0
57	58	27.3	OJ	2.0
58	59	29.4	OJ	2.0
59	60	23.0	OJ	2.0

```
> dbGetQuery(con,"select len,supp from hamster")
```

	LEN	SUPP
0	4.2	VC
1	11.5	VC
2	7.3	VC
3	5.8	VC
4	6.4	VC
5	10.0	VC
6	11.2	VC
7	11.2	VC
8	5.2	VC
9	7.0	VC
10	16.5	VC
11	16.5	VC
12	15.2	VC
13	17.3	VC
14	22.5	VC
15	17.3	VC
16	13.6	VC
17	14.5	VC
18	18.8	VC
19	15.5	VC
20	23.6	VC
21	18.5	VC
22	33.9	VC
23	25.5	VC

24	26.4	VC
25	32.5	VC
26	26.7	VC
27	21.5	VC
28	23.3	VC
29	29.5	VC
30	15.2	OJ
31	21.5	OJ
32	17.6	OJ
33	9.7	OJ
34	14.5	OJ
35	10.0	OJ
36	8.2	OJ
37	9.4	OJ
38	16.5	OJ
39	9.7	OJ
40	19.7	OJ
41	23.3	OJ
42	23.6	OJ
43	26.4	OJ
44	20.0	OJ
45	25.2	OJ
46	25.8	OJ
47	21.2	OJ
48	14.5	OJ
49	27.3	OJ
50	25.5	OJ
51	26.4	OJ
52	22.4	OJ
53	24.5	OJ
54	24.8	OJ
55	30.9	OJ
56	26.4	OJ
57	27.3	OJ
58	29.4	OJ
59	23.0	OJ

```
> dbGetQuery(con,"select len,supp,dose from hamster where dose > 1.0")
```

	LEN	SUPP	DOSE
0	23.6	VC	2
1	18.5	VC	2
2	33.9	VC	2
3	25.5	VC	2
4	26.4	VC	2
5	32.5	VC	2
6	26.7	VC	2
7	21.5	VC	2
8	23.3	VC	2
9	29.5	VC	2
10	25.5	OJ	2
11	26.4	OJ	2
12	22.4	OJ	2
13	24.5	OJ	2
14	24.8	OJ	2

```
15 30.9 OJ 2
16 26.4 OJ 2
17 27.3 OJ 2
18 29.4 OJ 2
19 23.0 OJ 2
```

>

or, something more complicated:

```
> a=dbGetQuery(con,"select len from hamster")
```

```
> a
```

```
      LEN
0  4.2
1 11.5
2  7.3
3  5.8
4  6.4
5 10.0
6 11.2
7 11.2
8  5.2
9  7.0
10 16.5
11 16.5
12 15.2
13 17.3
14 22.5
15 17.3
16 13.6
17 14.5
18 18.8
19 15.5
20 23.6
21 18.5
22 33.9
23 25.5
24 26.4
25 32.5
26 26.7
27 21.5
28 23.3
29 29.5
30 15.2
31 21.5
32 17.6
33  9.7
34 14.5
35 10.0
36  8.2
37  9.4
38 16.5
39  9.7
40 19.7
41 23.3
```

```
42 23.6
43 26.4
44 20.0
45 25.2
46 25.8
47 21.2
48 14.5
49 27.3
50 25.5
51 26.4
52 22.4
53 24.5
54 24.8
55 30.9
56 26.4
57 27.3
58 29.4
59 23.0
```

```
> b=data.frame(1:60,len=a$LEN,lensq=a$LEN^2); b=b[sample(1:60),]
```

```
> b
```

```
      X1.60  len  lensq
34      34  9.7   94.09
53      53 22.4  501.76
30      30 29.5  870.25
32      32 21.5  462.25
10      10  7.0   49.00
33      33 17.6  309.76
39      39 16.5  272.25
59      59 29.4  864.36
42      42 23.3  542.89
57      57 26.4  696.96
20      20 15.5  240.25
58      58 27.3  745.29
 2         2 11.5  132.25
13      13 15.2  231.04
36      36 10.0  100.00
29      29 23.3  542.89
60      60 23.0  529.00
37      37  8.2   67.24
52      52 26.4  696.96
 5         5  6.4   40.96
49      49 14.5  210.25
11      11 16.5  272.25
51      51 25.5  650.25
22      22 18.5  342.25
55      55 24.8  615.04
 6         6 10.0  100.00
38      38  9.4   88.36
17      17 13.6  184.96
12      12 16.5  272.25
18      18 14.5  210.25
19      19 18.8  353.44
27      27 26.7  712.89
```

14	14	17.3	299.29
31	31	15.2	231.04
56	56	30.9	954.81
24	24	25.5	650.25
50	50	27.3	745.29
54	54	24.5	600.25
1	1	4.2	17.64
35	35	14.5	210.25
3	3	7.3	53.29
7	7	11.2	125.44
41	41	19.7	388.09
44	44	26.4	696.96
47	47	25.8	665.64
40	40	9.7	94.09
4	4	5.8	33.64
26	26	32.5	1056.25
45	45	20.0	400.00
21	21	23.6	556.96
28	28	21.5	462.25
8	8	11.2	125.44
16	16	17.3	299.29
9	9	5.2	27.04
23	23	33.9	1149.21
46	46	25.2	635.04
25	25	26.4	696.96
43	43	23.6	556.96
48	48	21.2	449.44
15	15	22.5	506.25

```
> dbWriteTable(con,"hamsq",b)
```

```
[1] TRUE
```

```
> dbGetQuery(con,"select a.dose,b.lensq,a.len,b.len from hamster a, hamsq b where a.len=b.len")
```

	DOSE	LENSQ	LEN	LEN
0	0.5	17.64	4.2	4.2
1	0.5	27.04	5.2	5.2
2	0.5	33.64	5.8	5.8
3	0.5	40.96	6.4	6.4
4	0.5	49.00	7.0	7.0
5	0.5	53.29	7.3	7.3
6	0.5	67.24	8.2	8.2
7	0.5	88.36	9.4	9.4
8	0.5	94.09	9.7	9.7
9	0.5	94.09	9.7	9.7
10	0.5	94.09	9.7	9.7
11	0.5	94.09	9.7	9.7
12	0.5	100.00	10.0	10.0
13	0.5	100.00	10.0	10.0
14	0.5	100.00	10.0	10.0
15	0.5	100.00	10.0	10.0
16	0.5	125.44	11.2	11.2
17	0.5	125.44	11.2	11.2
18	0.5	125.44	11.2	11.2
19	0.5	125.44	11.2	11.2

20	0.5	132.25	11.5	11.5
21	1.0	184.96	13.6	13.6
22	1.0	210.25	14.5	14.5
23	0.5	210.25	14.5	14.5
24	1.0	210.25	14.5	14.5
25	1.0	210.25	14.5	14.5
26	0.5	210.25	14.5	14.5
27	1.0	210.25	14.5	14.5
28	1.0	210.25	14.5	14.5
29	0.5	210.25	14.5	14.5
30	1.0	210.25	14.5	14.5
31	1.0	231.04	15.2	15.2
32	0.5	231.04	15.2	15.2
33	1.0	231.04	15.2	15.2
34	0.5	231.04	15.2	15.2
35	1.0	240.25	15.5	15.5
36	1.0	272.25	16.5	16.5
37	0.5	272.25	16.5	16.5
38	1.0	272.25	16.5	16.5
39	1.0	272.25	16.5	16.5
40	0.5	272.25	16.5	16.5
41	1.0	272.25	16.5	16.5
42	1.0	272.25	16.5	16.5
43	0.5	272.25	16.5	16.5
44	1.0	272.25	16.5	16.5
45	1.0	299.29	17.3	17.3
46	1.0	299.29	17.3	17.3
47	1.0	299.29	17.3	17.3
48	1.0	299.29	17.3	17.3
49	0.5	309.76	17.6	17.6
50	2.0	342.25	18.5	18.5
51	1.0	353.44	18.8	18.8
52	1.0	388.09	19.7	19.7
53	1.0	400.00	20.0	20.0
54	1.0	449.44	21.2	21.2
55	2.0	462.25	21.5	21.5
56	0.5	462.25	21.5	21.5
57	2.0	462.25	21.5	21.5
58	0.5	462.25	21.5	21.5
59	2.0	501.76	22.4	22.4
60	1.0	506.25	22.5	22.5
61	2.0	529.00	23.0	23.0
62	2.0	542.89	23.3	23.3
63	1.0	542.89	23.3	23.3
64	2.0	542.89	23.3	23.3
65	1.0	542.89	23.3	23.3
66	2.0	556.96	23.6	23.6
67	1.0	556.96	23.6	23.6
68	2.0	556.96	23.6	23.6
69	1.0	556.96	23.6	23.6
70	2.0	600.25	24.5	24.5
71	2.0	615.04	24.8	24.8
72	1.0	635.04	25.2	25.2
73	2.0	650.25	25.5	25.5

```

74  2.0  650.25  25.5  25.5
75  2.0  650.25  25.5  25.5
76  2.0  650.25  25.5  25.5
77  1.0  665.64  25.8  25.8
78  2.0  696.96  26.4  26.4
79  2.0  696.96  26.4  26.4
80  1.0  696.96  26.4  26.4
81  2.0  696.96  26.4  26.4
82  2.0  696.96  26.4  26.4
83  2.0  696.96  26.4  26.4
84  1.0  696.96  26.4  26.4
85  2.0  696.96  26.4  26.4
86  2.0  696.96  26.4  26.4
87  2.0  696.96  26.4  26.4
88  1.0  696.96  26.4  26.4
89  2.0  696.96  26.4  26.4
90  2.0  696.96  26.4  26.4
91  2.0  696.96  26.4  26.4
92  1.0  696.96  26.4  26.4
93  2.0  696.96  26.4  26.4
94  2.0  712.89  26.7  26.7
95  1.0  745.29  27.3  27.3
96  2.0  745.29  27.3  27.3
97  1.0  745.29  27.3  27.3
98  2.0  745.29  27.3  27.3
99  2.0  864.36  29.4  29.4
100 2.0  870.25  29.5  29.5
101 2.0  954.81  30.9  30.9
102 2.0 1056.25  32.5  32.5
103 2.0 1149.21  33.9  33.9

```

```
> dbRemoveTable(con,"hamsq")
```

```

Error in oraFetch(rs, n = -1) : RS-DBI driver: (ORA-24374: define not done before
fetch or execute and fetch
)

```

```

Error in oraCloseConnection(conn, ...) : pending result sets in the Oracle server --
must close manually
[1] FALSE

```

```
> ?dbWriteTable
```

```
help() for dbWriteTable is shown in browser mozilla ...
```

```
Use      help( dbWriteTable , htmlhelp=FALSE)
```

```
or      options(htmlhelp = FALSE)
```

```
to revert.
```

```
Warning message:
```

```
Using non-linked HTML file: style sheet and hyperlinks may be incorrect in:
```

```
help("dbWriteTable")
```

```
>
```

## Adding data to a table

dbWriteTable has a flag: append, which allows us to add data to a table

```
> a=dbReadTable(con,"hamster")
```

```
> a
```

	LEN	SUPP	DOSE
1	4.2	VC	0.5
2	11.5	VC	0.5
3	7.3	VC	0.5
4	5.8	VC	0.5
5	6.4	VC	0.5
6	10.0	VC	0.5
7	11.2	VC	0.5
8	11.2	VC	0.5
9	5.2	VC	0.5
10	7.0	VC	0.5
11	16.5	VC	1.0
12	16.5	VC	1.0
13	15.2	VC	1.0
14	17.3	VC	1.0
15	22.5	VC	1.0
16	17.3	VC	1.0
17	13.6	VC	1.0
18	14.5	VC	1.0
19	18.8	VC	1.0
20	15.5	VC	1.0
21	23.6	VC	2.0
22	18.5	VC	2.0
23	33.9	VC	2.0
24	25.5	VC	2.0
25	26.4	VC	2.0
26	32.5	VC	2.0
27	26.7	VC	2.0
28	21.5	VC	2.0
29	23.3	VC	2.0
30	29.5	VC	2.0
31	15.2	OJ	0.5
32	21.5	OJ	0.5
33	17.6	OJ	0.5
34	9.7	OJ	0.5
35	14.5	OJ	0.5
36	10.0	OJ	0.5
37	8.2	OJ	0.5
38	9.4	OJ	0.5
39	16.5	OJ	0.5
40	9.7	OJ	0.5
41	19.7	OJ	1.0
42	23.3	OJ	1.0
43	23.6	OJ	1.0
44	26.4	OJ	1.0
45	20.0	OJ	1.0
46	25.2	OJ	1.0
47	25.8	OJ	1.0
48	21.2	OJ	1.0
49	14.5	OJ	1.0
50	27.3	OJ	1.0

```
51 25.5 OJ 2.0
52 26.4 OJ 2.0
53 22.4 OJ 2.0
54 24.5 OJ 2.0
55 24.8 OJ 2.0
56 30.9 OJ 2.0
57 26.4 OJ 2.0
58 27.3 OJ 2.0
59 29.4 OJ 2.0
60 23.0 OJ 2.0
```

```
> a2=a[1:3,]
```

```
> a2
```

```
      LEN SUPP DOSE
1  4.2   VC  0.5
2 11.5   VC  0.5
3  7.3   VC  0.5
```

```
> rownames(a2)=100:102
```

```
> a2
```

```
      LEN SUPP DOSE
100  4.2   VC  0.5
101 11.5   VC  0.5
102  7.3   VC  0.5
```

```
> dbWriteTable(con, "hamster", a2, append=T)
```

```
[1] TRUE
```

```
> a=dbReadTable(con, "hamster")
```

```
> a
```

```
      LEN SUPP DOSE
1  4.2   VC  0.5
2 11.5   VC  0.5
3  7.3   VC  0.5
4  5.8   VC  0.5
5  6.4   VC  0.5
6 10.0   VC  0.5
7 11.2   VC  0.5
8 11.2   VC  0.5
9  5.2   VC  0.5
10 7.0   VC  0.5
11 16.5   VC  1.0
12 16.5   VC  1.0
13 15.2   VC  1.0
14 17.3   VC  1.0
15 22.5   VC  1.0
16 17.3   VC  1.0
17 13.6   VC  1.0
18 14.5   VC  1.0
19 18.8   VC  1.0
20 15.5   VC  1.0
21 23.6   VC  2.0
22 18.5   VC  2.0
23 33.9   VC  2.0
```

```
24 25.5 VC 2.0
25 26.4 VC 2.0
26 32.5 VC 2.0
27 26.7 VC 2.0
28 21.5 VC 2.0
29 23.3 VC 2.0
30 29.5 VC 2.0
31 15.2 OJ 0.5
32 21.5 OJ 0.5
33 17.6 OJ 0.5
34 9.7 OJ 0.5
35 14.5 OJ 0.5
36 10.0 OJ 0.5
37 8.2 OJ 0.5
38 9.4 OJ 0.5
39 16.5 OJ 0.5
40 9.7 OJ 0.5
41 19.7 OJ 1.0
42 23.3 OJ 1.0
43 23.6 OJ 1.0
44 26.4 OJ 1.0
45 20.0 OJ 1.0
46 25.2 OJ 1.0
47 25.8 OJ 1.0
48 21.2 OJ 1.0
49 14.5 OJ 1.0
50 27.3 OJ 1.0
51 25.5 OJ 2.0
52 26.4 OJ 2.0
53 22.4 OJ 2.0
54 24.5 OJ 2.0
55 24.8 OJ 2.0
56 30.9 OJ 2.0
57 26.4 OJ 2.0
58 27.3 OJ 2.0
59 29.4 OJ 2.0
60 23.0 OJ 2.0
100 4.2 VC 0.5
101 11.5 VC 0.5
102 7.3 VC 0.5
```

>

## Going slowly through a query

If the database is very big, we might not want to load it all at once into memory. We might want to compute things slowly over it.

```
> x=rnorm(10000);y=1:10000
> a=data.frame(x=x,y=y)
> dbWriteTable(con,"bigtab",a)
```

```
[1] TRUE
```

>

Now assume we want to calculate the sum of all x:

```
> quer=dbSendQuery(con,"select X from bigtab")
> sumx=0
> while(!dbHasCompleted(quer) ) {
  dat=fetch(quer,n=1)
  if( !dbHasCompleted(quer) ){
    sumx = sumx+dat$X[1]
  }
}
+ + + + + dbClearResult(quer)
> [1] TRUE
> sumx
[1] -50.16107
```

You see that going over a table line by line is slow. We can do in in batches of 100:

```
> quer=dbSendQuery(con,"select X from bigtab")
> sumx=0
> while(!dbHasCompleted(quer) ) {
  dat=fetch(quer,n=100)
  if( !dbHasCompleted(quer) ){
    sumx = sumx+sum(dat$X)
  }
}
+ + + + + dbClearResult(quer)
> sumx
[1] -50.16107
> @@@
[1] -50.16107
>
```

The command `dbClearResult()` is very important. Without it, we can not submit another query. Let us see what can go wrong:

```
> quer=dbSendQuery(con,"select X from bigtab")
> quer=3
> quer=dbSendQuery(con,"select X from bigtab")
Error in oraPrepareStatement(con, statement, bind = NULL) :
  RS-DBI driver: (cannot allocate a new resultSet -- maximum of 1 resultSets
already reached)
Error in oraExecDirect(conn, statement, ...) :
  could not exec direct statement select X from bigtab
>
```

So, we would like to submit another query, but we can not, and we don't have quer any more to call `dbClearresult` with it.

There is the following convenient command:

```
> dbListResults(con)
```

```

[[1]]
<OraResult: (24319,1,25)>
> quer=dbListResults(con) [[1]]
> dbClearResult(quer)

[1] TRUE
> quer=dbSendQuery(con,"select X from bigtab")
>
Now all went well.

> dbClearResult(quer)

[1] TRUE
> dbListTables(con)

[1] "DUAL" "SYSTEM_PRIVILEGE_MAP" "TABLE_PRIVILEGE_MAP"
[4] "STMT_AUDIT_OPTION_MAP" "OL$" "OL$HINTS"
[7] "OL$NODES" "MAP_OBJECT" "AUDIT_ACTIONS"
[10] "PSTUBTBL" "ODCI_SECOBJ$" "ODCI_WARNINGS$"
[13] "DEF$_TEMP$LOB" "R_TEST" "COUNTRIES"
[16] "DEPARTMENTS" "EMPLOYEES" "JOBS"
[19] "JOB_HISTORY" "LOCATIONS" "REGIONS"
[22] "TEST" "A2_C3B" "A_C1A"
[25] "ARRESTS" "HAMSTER" "BIGTAB"

> dbRemoveTable(con,"hamster")

Error in oraFetch(rs, n = -1) : RS-DBI driver: (ORA-24374: define not done before
fetch or execute and fetch
)
[1] FALSE

> dbRemoveTable(con,"bigtab")

Error in oraFetch(rs, n = -1) : RS-DBI driver: (ORA-24374: define not done before
fetch or execute and fetch
)
Error in oraCloseConnection(conn, ...) : pending result sets in the Oracle server --
must close manually
[1] FALSE

> dbListTables(con)

[1] "DUAL" "SYSTEM_PRIVILEGE_MAP" "TABLE_PRIVILEGE_MAP"
[4] "STMT_AUDIT_OPTION_MAP" "OL$" "OL$HINTS"
[7] "OL$NODES" "MAP_OBJECT" "AUDIT_ACTIONS"
[10] "PSTUBTBL" "ODCI_SECOBJ$" "ODCI_WARNINGS$"
[13] "DEF$_TEMP$LOB" "R_TEST" "COUNTRIES"
[16] "DEPARTMENTS" "EMPLOYEES" "JOBS"
[19] "JOB_HISTORY" "LOCATIONS" "REGIONS"
[22] "TEST" "A2_C3B" "A_C1A"
[25] "ARRESTS"

```