



Serwery baz danych

laboratorium 5

dr inż. Arkadiusz Mirakowski



arkadiusz.mirakowski@ug.edu.pl

Wszystkie screenshots przedstawiające kod źródłowy skryptu, zapytania, polecenia etc. i efekty ich działań (jako załączniki) proszę wysłać na adres: arkadiusz.mirakowski@ug.edu.pl podając w temacie wiadomości → ServBD – lab 5

Licznik uruchomień skryptu

```
[user@server ~]$ cat -n skrypt.sh
 1  #!/bin/bash
 2
 3  if [ -f "plik.txt" ]
 4  then
 5      licznik=$(cat "plik.txt")
 6      licznik=$((licznik+1))
 7      echo $licznik>plik.txt
 8      echo $licznik
 9  else
10      echo "pliku nie ma"
11      licznik=0
12      licznik=$((licznik+1))
13      echo $licznik>plik.txt
14      echo ""
15      echo $licznik
16  fi
17
[user@server ~]$
```

```
[user@server ~]$ ./skrypt.sh
pliku nie ma
```

```
1
[user@server ~]$ ./skrypt.sh
2
[user@server ~]$ ./skrypt.sh
3
[user@server ~]$ ./skrypt.sh
4
[user@server ~]$ ./skrypt.sh
5
[user@server ~]$
```

Instrukcja FOR

Pierwszy rodzaj pętli FOR (konstrukcja analogiczna do instrukcji FOR w C++):

```
[user@serwer ~]$ for ((i=1;i<10;i++))  
> do  
> echo $i  
> done  
1  
2  
3  
4  
5  
6  
7  
8  
9
```

- ✓ do – początek pętli
- ✓ done – koniec pętli

Instrukcja FOR

Drugi rodzaj pętli FOR:

- przypadek 1

```
[user@serwer ~]$ ls
Dokumenty Muzyka Obrazy Pobrane Publiczny Pulpit Szablony Wideo

[user@serwer ~]$ for i in "P*"
> do
> echo $i
> done
Pobrane Publiczny Pulpit
```

wyświetl wszystkie zasoby,
których nazwa zaczyna się na
literę P

- przypadek 2

```
[user@serwer ~]$ for i in "P*"; do echo $i; done
Pobrane Publiczny Pulpit
```

- przypadek 3

```
[user@serwer ~]$ for i in 1 100 abc a x; do echo $i; done
1
100
abc
a
x
```

sekwencja, po której pętla ma
iterować została wpisana
ręcznie

Instrukcja FOR

- przypadek 4

```
[user@serwer ~]$ cat -n s1.sh
```

```
1  #!/bin/bash
```

```
2
```

```
3  for i in {1..3}
```

```
4  do
```

```
5      echo $i
```

```
6  done
```

```
7
```

```
[user@serwer ~]$ ./s1.sh
```

```
1
```

```
2
```

```
3
```

- przypadek 5 (krok=2)

```
[user@serwer ~]$ cat -n s1.sh
```

```
1  #!/bin/bash
```

```
2
```

```
3  for i in {2..10..2}
```

```
4  do
```

```
5      echo $i
```

```
6  done
```

```
7
```

```
[user@serwer ~]$ ./s1.sh
```

```
2
```

```
4
```

```
6
```

```
8
```

```
10
```

Instrukcja FOR

- przypadek 6

```
[user@server ~]$ cat -n s1.sh
1 #!/bin/bash
2
3 n=3
4 for i in {1..$n}
5 do
6     echo $i
7 done
8
[user@server ~]$ ./s1.sh
{1..3}
```



```
1 #!/bin/bash
2
3 n=3
4 for i in $(seq $n)
5 do
6     echo $i
7 done
```

- przypadek 7

```
[user@server ~]$ cat -n s1.sh
1 #!/bin/bash
2
3 n=3
4 for ((i=1;i<$n;i++))
5 do
6     echo $i
7 done
8
[user@server ~]$ ./s1.sh
1
2
```

Instrukcja FOR

- przypadek 8

```
[user@server ~]$ cat -n skrypt.sh
1  #!/bin/bash
2
3  n=3
4  for ((i=1;i<$n+1;i++))
5  do
6      ls -l test$i
7  done
```

```
[user@server ~]$ ./skrypt.sh
-rw-rw-r--. 1 user user 0 11-23 10:12 test1
-rw-rw-r--. 1 user user 0 11-23 10:12 test2
-rw-rw-r--. 1 user user 0 11-23 10:12 test3
[user@server ~]$
[user@server ~]$ ls -l t*
-rw-rw-r--. 1 user user 0 11-23 10:12 test1
-rw-rw-r--. 1 user user 0 11-23 10:12 test2
-rw-rw-r--. 1 user user 0 11-23 10:12 test3
[user@server ~]$
```


Tablica jednowymiarowa

- przypadek 1

```
1  #!/bin/bash
2
3  tab=(1 2 3 4 a b c)
4  echo ""
5  echo -n "pierwszy element tablicy: "
6  echo "${tab[0]}"
7
8  echo ""
9  echo "zawartość tablicy:"
10 for i in ${tab[*]}
11 do
12     echo $i
13 done
14
15 echo ""
16
```

pierwszy element tablicy: 1

zawartość tablicy:

1
2
3
4
a
b
c

[user@server ~]\$ █

Tablica jednowymiarowa

- przypadek 2

```
1  #!/bin/bash
2
3  n=3
4  echo ""
5  for i in $(seq $n)
6  do
7      read -p "podaj t[$i]: " element
8      tab=(${tab[@]} $element)
9  done
10 echo ""
11 echo "tab[2] = ${tab[1]}"
12
13 echo ""
14 echo -n "tab: "
15 echo ${tab[@]}
16
17 echo ""
18 echo "tab: "
19 for i in ${tab[*]}
20 do
21     echo $i
22 done
23
```

```
podaj t[1]: 1
podaj t[2]: 11
podaj t[3]: 111
```

```
tab[2] = 11
```

```
tab: 1 11 111
```

```
tab:
```

```
1
```

```
11
```

```
111
```

```
[user@server ~]$
```

```
podaj t[1]: abc
podaj t[2]: xyz
podaj t[3]: cde
```

```
tab[2] = xyz
```

```
tab: abc xyz cde
```

```
tab:
```

```
abc
```


```
xyz
```

```
cde
```

```
[user@server ~]$
```

Zadanie 20

Zaimportuj bazę danych z poniższego pliku do bazy danych o nazwie french.

 french-towns-communes-francaises-1.0.tar.gz

Następnie wyświetl zawartość tabeli regions z bazy danych french.

Zadanie 21

Utwórz katalog `/backup`, a następnie napisz skrypt, który:

- wykona kopię zapasową bazy danych french do pliku o rozszerzeniu `dump` zgodnie z poniższymi wytycznymi:
 - po pierwszym uruchomieniu skryptu powstanie kopia o nazwie `kopia1.dump` i zostanie ona zapisana w katalogu `backup`
 - po drugim uruchomieniu skryptu powstanie kopia o nazwie `kopia2.dump` i zostanie ona zapisana w katalogu `backup`
 - po trzecim, czwartym lub piątym uruchomieniu skryptu analogicznie
 - po szóstym uruchomieniu skryptu nastąpi usunięcie wszystkich plików `dump`, a następnie powstanie kopia o nazwie `kopia1.dump` i zostanie ona zapisana w katalogu `backup`
 - po siódmym uruchomieniu skryptu powstanie kopia o nazwie `kopia2.dump` i zostanie ona zapisana w katalogu `backup`
 - po 8, 9 lub 10 uruchomieniu skryptu analogicznie
 - etc.

Zadanie 22

Napisz skrypt, który:

- wyświetli zawartość katalogu /backup (listę pięciu plików dump)
- usunie bazę danych french i wyświetli komunikat, że baza danych french nie istnieje
- zostanie poinformowany, że za chwilę nastąpi przywrócenie bazy danych french
- poprosi użytkownika o podanie nazwy pliku dump, z którego ma zostać przywrócona baza danych french
- przeprowadzi kontrolę, czy wybrany przez użytkownika plik dump istnieje:
 - ✓ jeśli nie istnieje, wyświetli stosowny komunikat
 - ✓ jeśli istnieje, nastąpi przywrócenie bazy danych, a następnie wyświetli zawartość tabeli regions z bazy danych french

Zadanie 23

Wyczyść zawartość katalogu backup, a następnie napisz skrypt, który:

- wygeneruje tablicę zawierającą elementy: kopia1.dump, kopia2.dump, kopia3.dump, kopia4.dump, kopia5.dump
- jeśli będzie to pierwsze uruchomienie skryptu, to powstanie kopia o nazwie kopia1.dump i zostanie ona zapisana w katalogu backup
- jeśli będzie to drugie uruchomienie skryptu, to powstanie kopia o nazwie kopia2.dump i zostanie ona zapisana w katalogu backup
- jeśli będzie to trzecie, czwarte lub piąte uruchomieniu skryptu analogicznie
- jeśli będzie to szóste uruchomienie skryptu, nastąpi usunięcie wszystkich plików dump, a następnie powstanie kopia o nazwie kopia1.dump i zostanie ona zapisana w katalogu backup
- jeśli będzie to 7, 8, 9 lub 10 uruchomieniu skryptu analogicznie
- etc.

Zadanie 24

Napisz skrypt, który:

- wygeneruje tablicę zawierającą elementy: kopia1.dump, kopia2.dump, kopia3.dump, kopia4.dump, kopia5.dump
- usunie bazę danych french i wyświetli komunikat, że baza danych french nie istnieje
- zostanie poinformowany, że za chwilę nastąpi przywrócenie bazy danych french
- poprosi użytkownika o podanie nazwy pliku dump, z którego ma zostać przywrócona baza danych french
- przeprowadzi kontrolę, czy wybrany przez użytkownika plik dump istnieje:
 - ✓ jeśli nie istnieje, wyświetli stosowny komunikat
 - ✓ jeśli istnieje, nastąpi przywrócenie bazy danych, a następnie wyświetli zawartość tabeli regions z bazy danych french