



Serwery baz danych

wykład 6

dr inż. Arkadiusz Mirakowski



arkadiusz.mirakowski@ug.edu.pl

Skrypt z edytorem vi

```
[user@server ~]$  
[user@server ~]$ touch s.sh  
[user@server ~]$  
[user@server ~]$ vi s.sh
```

- włączenie trybu pisania → a

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

```
ls  
pwd
```

- zapis i wyjście: (1) ESC (2) :wq
- wyjście (bez zapisu): (1) ESC (2) :q!
- włączenie numeracji linii: (1) (ESC) (2) :set number
- zmiana uprawnień do skryptu ...

Skrypty w powłoce BASH

- skrypt 1

- na chwilę obecną skrypt można uruchomić tylko z bieżącego miejsca

```
1 #!/bin/bash
2
3 sudo su -c "psql" postgres
4 #sudo su -c "psql -d postgres" postgres
```

- aby skrypt można było wywołać z dowolnego miejsca, należy go przekopiować do katalogu /usr/bin

Skrypty w powłoce BASH

```
[user@server ~]$ pwd
/home/user
[user@server ~]$ ls -l s.sh
-rwxrwxr-x. 1 user user 84 12-09 10:51 s.sh
[user@server ~]$

[user@server ~]$ sudo cp s.sh /usr/bin
[user@server ~]$
[user@server ~]$ sudo ls -l /usr/bin/s.sh
-rwxr-xr-x. 1 root root 84 12-09 10:54 /usr/bin/s.sh
[user@server ~]$
[user@server ~]$ cd /home
[user@server home]$ ls
user
[user@server home]$ s.sh
psql (13.8)
Type "help" for help.

postgres=#
```


Skrypty w powłoce BASH

- skrypt 2

```
3 bd="postgres"
4 user="postgres"
5 echo ""
6 sudo -i -u $user psql -d $bd -c "\conninfo"
7
8 user_search="user12"
9 spr=$(sudo -i -u $user psql -t -d $bd -c "select 1 as ile
10      from pg_catalog.pg_roles where rolname='$user_search'")
11
12 echo ""
13 if [ $spr ]
14 then
15     echo "użytkownik istnieje"
16 else
17     echo "użytkownika brak"
18 fi
19
20 echo ""
```

You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql"

użytkownik istnieje

Skrypty w powłoce BASH

- skrypt 3

```
3 bd="postgres"
4 user="postgres"
5 echo ""
6 sudo -i -u $user psql -d $bd -c "\conninfo"
7
8 tab_search="osoby"
9 spr=$(sudo -i -u $user psql -t -d $bd -c "select 1 as spr
10      from information_schema.tables where table_name='$tab_search'")
11
12 echo ""
13 if [ $spr ]
14 then
15     echo "tabela istnieje"
16 else
17     echo "tabeli brak"
18 fi
19
20 echo ""
```

You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql"

tabela istnieje

Skrypty w powłoce BASH

- skrypt 4

```
3 bd="postgres"
4 user="user1"
5 echo ""
6 sudo -i psql -U $user -h localhost -d $bd -c "\conninfo"
7
8 echo ""
```

Password for user user1:

You are connected to database "postgres" as user "user1" on host "localhost"

```
3 bd="postgres"
4 user="user1"
5 echo ""
6 sudo -i psql -U $user -d $bd -c "\conninfo"
7
8 echo ""
```

psql: error: FATAL: Peer authentication failed for user "user1"

Skrypty w powłoce BASH

- skrypt 5

```
postgres=# \conninfo
```

```
You are connected to database "postgres" as user "postgres" via socket in  
postgres=#
```

```
postgres=# SELECT privilege_type FROM information_schema.table_privileges  
postgres-# WHERE grantee='postgres' AND table_name='osoby';  
privilege_type
```

```
-----  
INSERT  
SELECT  
UPDATE  
DELETE  
TRUNCATE  
REFERENCES  
TRIGGER
```

```
(7 rows)
```

```
postgres=> \conninfo
```

```
You are connected to database "postgres" as user "user1" on host "localhost"  
postgres=>
```

```
postgres=> SELECT privilege_type FROM information_schema.table_privileges  
postgres-> WHERE grantee='postgres' AND table_name='osoby';  
privilege_type
```

```
-----  
(0 rows)
```


Skrypty w powłoce BASH

- skrypt

```
3  bd="postgres"
4  user="postgres"
5
6  echo ""
7  sudo -i psql -U $user -h localhost -d $bd -c "\conninfo"
8
9  spr=$(sudo -i psql -U $user -h localhost -t -d $bd -c "select count(*)
10   from information_schema.table_privileges where grantee='$user' and table_name='osoby'")
11
12  echo ""
13  if [ $spr -gt 0 ]
14  then
15      echo "liczba uprawnień: $spr"
16  else
17      echo "uprawnień brak"
18  fi
19
20  echo ""
21  spr=$(sudo -i psql -U $user -h localhost -d $bd -c "select privilege_type
22   from information_schema.table_privileges where grantee='$user' and table_name='osoby'")
23  echo $spr
```

Skrypty w powłoce BASH

Password for user postgres:
You are connected to database "postgres" as user "postgres" on host "localhost" (address "::1") a
Password for user postgres:

liczba uprawnień: 7

Password for user postgres:
privilege_type ----- INSERT SELECT UPDATE DELETE TRUNCATE REFERENCES TRIGGER (7 rows)

```
3 bd="postgres"
4 #user="postgres"
5 user="user1"
6
7 echo ""
8 sudo -i psql -U $user -h localhost -d $bd -c "\conninfo"
```

...

Password for user user1:
You are connected to database "postgres" as user "user1" on host "localhost"
Password for user user1:

uprawnień brak

Password for user user1:
privilege_type ----- (0 rows)

Skrypty w powłoce BASH

- skrypt 6

```
3  bd="postgres"
4  user="user1"
5
6  echo ""
7  sudo -i psql -U postgres -h localhost -d $bd -c "\conninfo"
8  sudo -i psql -U postgres -h localhost -d $bd -c "create role rola1"
9  sudo -i psql -U postgres -h localhost -d $bd -c "grant select on osoby to rola1"
10 sudo -i psql -U postgres -h localhost -d $bd -c "grant rola1 to $user"
11 sudo -i psql -U postgres -h localhost -d $bd -c "\du"
12
13
14 echo ""
15 sudo -i psql -U $user -h localhost -d $bd -c "\conninfo"
16 echo ""
17 sudo -i psql -U $user -h localhost -d $bd -c "select*from osoby"
18
19
20 sudo -i psql -U postgres -h localhost -d $bd -c "revoke select on osoby from rola1"
21 sudo -i psql -U postgres -h localhost -d $bd -c "drop role rola1"
22 sudo -i psql -U postgres -h localhost -d $bd -c "\du"
23
24 echo ""
```

Skrypty w powłoce BASH

```
3 bd="postgres"
4 user="user1"
5
6 echo ""
7 sudo -i psql -U postgres -h localhost -d $bd -c "\conninfo"
8 sudo -i psql -U postgres -h localhost -d $bd -c "create role rolal"
9 sudo -i psql -U postgres -h localhost -d $bd -c "grant select on osoby to rolal"
10 sudo -i psql -U postgres -h localhost -d $bd -c "grant rolal to $user"
11 sudo -i psql -U postgres -h localhost -d $bd -c "\du"
```

Password for user postgres:

You are connected to database "postgres" as user "postgres" on host "localhost" (address ":::1") at port "5432".

Password for user postgres:

CREATE ROLE

Password for user postgres:

GRANT

Password for user postgres:

GRANT ROLE

Password for user postgres:

Role name	List of roles Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
rolal	Cannot login	{}
user1		{rolal}

Skrypty w powłoce BASH

```
14 echo ""
15 sudo -i psql -U $user -h localhost -d $bd -c "\conninfo"
16 echo ""
17 sudo -i psql -U $user -h localhost -d $bd -c "select*from osoby"
18
19
20 sudo -i psql -U postgres -h localhost -d $bd -c "revoke select on osoby from rolal"
21 sudo -i psql -U postgres -h localhost -d $bd -c "drop role rolal"
22 sudo -i psql -U postgres -h localhost -d $bd -c "\du"
```

Password for user user1:

You are connected to database "postgres" as user "user1" on host "localhost" (address "::1") at port "5432".

Password for user user1:

lp	nazwisko	miasto
1	Kowalski	Gdańsk
2	Malinowski	Sopot
3	Kowalewski	Gdańsk

(3 rows)

Password for user postgres:

REVOKE

Password for user postgres:

DROP ROLE

Password for user postgres:

Role name	List of roles Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
user1		{}

Skrypty w powłoce BASH

■ skrypt 7

```
3 bd="postgres"
4 user="user1"
5
6 echo ""
7 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\conninfo"
8 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "create role rola1"
9 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "grant select on osoby to rola1"
10 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "grant rola1 to $user"
11 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\du"
12
13
14 echo ""
15 sudo -i PGPASSWORD=user1 psql -U $user -h localhost -d $bd -c "\conninfo"
16 echo ""
17 #sudo -i PGPASSWORD=user1 psql -U $user -h localhost -d $bd -c "select*from osoby"
18
19
20 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "revoke select on osoby from rola1"
21 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "drop role rola1"
22 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\du"
```

Skrypty w powłoce BASH

You are connected to database "postgres" as user "postgres" on host "localhost" (address ":::1") at port "5432".

CREATE ROLE

GRANT

GRANT ROLE

List of roles		
Role name	Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
rola1	Cannot login	{}
user1		{rola1}

You are connected to database "postgres" as user "user1" on host "localhost" (address ":::1") at port "5432".

REVOKE

DROP ROLE

List of roles		
Role name	Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
user1		{}

Skrypty w powłoce BASH

- skrypt 8

```
3 bd="postgres"
4 user_new="user2"
5 pass_new=user_new
6
7 echo ""
8 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\conninfo"
9 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "create
10 user $user_new with password '$pass_new'"
11 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\du"
12
13
14 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "drop user $user_new"
15 sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d $bd -c "\du"
```

Skrypty w powłoce BASH

You are connected to database "postgres" as user "postgres" on host "localhost" (address "localhost")
CREATE ROLE

List of roles		
Role name	Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
user1		{}
user2		{}

DROP ROLE

List of roles		
Role name	Attributes	Member of
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}
user1		{}

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[@]}
```

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

tab[2] = 11

tab: 1 11 111

Tablica jednowymiarowa

- przypadek 3

```
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 "tab: "
15 for i in ${tab[*]}
16 do
17     echo $i
18 done
19
```

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

tab[2] = 11

tab:
1
11
111

Tablica jednowymiarowa

- przypadek 4

```
4  tab=(1 2 3 4 5)
5  pozycja=1
6
7  echo ""
8  echo ${tab[@]}
9  echo ""
10 read -p "Wprowadź nowy element tablicy: " element
11 tab[$pozycja]=$element
12
13 echo ""
14 echo ${tab[$pozycja]}
15
16 echo ""
17 echo ${tab[@]}
18 echo ""
```

1 2 3 4 5

Wprowadź nowy element tablicy: 100

100

1 100 3 4 5

Skrypty w powłoce BASH

- skrypt 9 – utworzenie użytkowników na podstawie zawartości tablicy

```
3. echo ""
4. tab=(u1 u2)
5. echo "elementy tablicy: ${tab[@]}"

7. echo ""
8. ile_elem=0
9. for i in ${tab[@]}
10 do
11     ile_elem=$((ile_elem+1))
12 done
13 echo "ilość elementów: $ile_elem"
14
15 echo ""
16 for i in ${tab[*]}
17 do
18     echo $i
19     sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d postgres -c "create
20     user $i with password '$i'"
21 done
```

Skrypty w powłoce BASH

```
22
23 echo ""
24 for i in ${tab[*]}
25 do
26     echo $i
27     sudo -i PGPASSWORD=postgres psql -U postgres -h localhost -d postgres -c "drop
28     user $i"
29 done
30
31 echo ""
```

elementy tablicy: u1 u2

ilość elementów: 2

u1
CREATE ROLE
u2
CREATE ROLE

u1
DROP ROLE
u2
DROP ROLE

Zapis do pliku

- przypadek 1 – zapis danych z klawiatury do pliku

```
1  #!/bin/bash
2
3  touch tab.txt
4
5  n=3
6
7  for ((i=0;i<$n;i++))
8  do
9      echo ""
10     echo "iteracja $((i+1))"
11     read -p "pierwszy element: " el1
12     read -p "drugi element: " el2
13     echo $el1,"$el2">>tab.txt
14 done
15
16 echo ""
17 cat tab.txt
```

iteracja 1
pierwszy element: a1
drugi element: a2

iteracja 2
pierwszy element: b1
drugi element: b2

iteracja 3
pierwszy element: c1
drugi element: c2

a1,a2
b1,b2
c1,c2

Zapis do pliku

- przypadek 2 – zapis dwóch tablic jednowymiarowych do pliku

```
1  #!/bin/bash
2
3  touch tab.txt
4
5  n=3
6  tab1=(a b c)
7  tab2=(a1 b1 c1)
8
9  for ((i=0;i<$n;i++))
10 do
11     echo "${tab1[$i]}","${tab2[$i]}>>tab.txt
12 done
13
14 cat tab.txt
15
16 rm tab.txt
```

a,a1
b,b1
c,c1

Podstawowe operacje na plikach

- komenda tr
- przypadek 1

```
[user@serwer ~]$ cat lorem-ipsu
```

```
 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam ac eni
ur in velit nunc. In ultricies iaculis lectus, at hendrerit massa. Fusc
 non nunc nec malesuada. Aenean luctus, sapien id fermentum finibus, me
it. Donec semper dui turpis, quis ullamcorper metus laoreet vel. Cras f
endum. Nunc ac arcu non urna auctor congue eu et nibh. Nullam a maximus
molestie justo non magna lacinia facilisis. Cras lobortis rutrum ex, vi
bero sit amet nisl iaculis iaculis eget a justo. Duis dui lacus, fringi
```

```
[user@serwer ~]$ cat lorem-ipsu
```

```
 Lorem ipsum dolor sit *met, consectetur *dipiscing elit. Nul
ur in velit nunc. In ultricies i*culis lectus, *t hendrerit m
 non nunc nec m*lesu*d*. Aene*n luctus, s*pien id fermentum f
it. Donec semper dui turpis, quis ull*mcorper metus l*oreet v
endum. Nunc *c *rcu non urn* *uctor congue eu et nibh. Null*m
molestie justo non m*gn* l*cini* f*cilisis. Cr*s lobortis rut
bero sit *met nisl i*culis i*culis eget * justo. Duis dui l*c
```

Podstawowe operacje na plikach

- przypadek 2

```
[user@serwer ~]$ cat lorem-ipsum.txt | tr -d "a"  
Lorem ipsum dolor sit met, consectetur dipiscing elit. Nul  
lit nunc. In ultricies iculis lectus, t hendrerit mss. Fusc  
lesud. Aenen luctus, spien id fermentum finibus, metus nisl  
rpis, quis ullmcorper metus loreet vel. Crs fermentum mgn i  
tor congue eu et nibh. Nullm mximus tortor. Nunc phretr ni  
Crs lobortis rutrum ex, vite luctus mss congue in. Vestibu  
is dui lcus, fringill c libero sed, feugit phretr neque.
```

Podstawowe operacje na plikach

- przypadek 3

```
[user@server ~]$ cat lorem-ipsum.txt | tr -d "bcdef"
```

```
Lorm ipsum olor sit amt, onsttur aipising lit. Nullam a nim  
riis iaulis ltus, at hnrrit massa. Fus uismo posur ros non vu  
um inius, mtus nisl vivrra x, i onvallis ros ligula s lit. Do  
gna i ui inum, in ultris nih inum. Nun a aru non urna autor o  
sagittis. Ut molsti justo non magna lainia ailisis. Cras loo  
o sit amt nisl iaulis iaulis gt a justo. Duis ui laus, ringil
```


Podstawowe operacje na plikach

- przypadek 4

```
[user@serwer ~]$ cat lorem-ipsum.txt | tr -d "\n"
Lorem ipsum dolor sit amet, consectetur adipiscing elit. I
ur in velit nunc. In ultricies iaculis lectus, at hendrerit
non nunc nec malesuada. Aenean luctus, sapien id fermentu
it. Donec semper dui turpis, quis ullamcorper metus laoree
endum. Nunc ac arcu non urna auctor congue eu et nibh. Nul
molestie justo non magna lacinia facilisis. Cras lobortis
bero sit amet nisl iaculis iaculis eget a justo. Duis dui
```

Podstawowe operacje na plikach

- komenda wc

```
[user@serwer ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
[user@serwer ~]$ wc osoby.txt
 3  6 39 osoby.txt
[user@serwer ~]$
```

- 3 – ilość linii
- 6 – ilość słów
- 39 – rozmiar pliku w B
- osoby.txt – nazwa pliku

```
[user@serwer ~]$ wc osoby.txt
 3  6 39 osoby.txt
[user@serwer ~]$
[user@serwer ~]$ wc -l osoby.txt
3 osoby.txt
[user@serwer ~]$ wc -w osoby.txt
6 osoby.txt
[user@serwer ~]$ wc -m osoby.txt
39 osoby.txt
```

Podstawowe operacje na plikach

- potok (połączenie kilku komend w jedno polecenie)

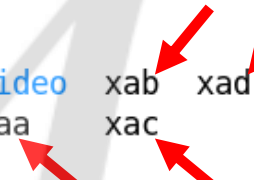
```
[user@serwer ~]$ cat osoby.txt | wc -l  
3  
[user@serwer ~]$ cat osoby.txt | wc -w  
6
```

- 3 – ilość linii
- 6 – ilość słów
- wynik pierwszej komendy został przekazany do drugiej komendy
- wynik ostateczny generuje ostatnia komenda w potoku

Podstawowe operacje na plikach

- komenda split (dzielenie pliku na części, domyślna ilość linii = 1000)
- przypadek 1 – oddzielenie pliku na skończoną ilość elementów

```
[user@serwer ~]$ ls
Dokumenty  lorem-ipsu[m].txt  Obrazy      Pobrane  Pulpit  Szablony  Wideo
liczby.txt Muzyka          osoby.txt   Publiczny s.sh     testowy
[user@serwer ~]$
[user@serwer ~]$ split -n 4 osoby.txt
[user@serwer ~]$
[user@serwer ~]$ ls
Dokumenty  lorem-ipsu[m].txt  Obrazy      Pobrane  Pulpit  Szablony  Wideo  xab  xad
liczby.txt Muzyka          osoby.txt   Publiczny s.sh     testowy xaa  xac
[user@serwer ~]$
```



nazwy plików są generowane automatycznie → pierwszy jest prefix „x”, następnie x+„aa”, x+„ab”, x+„ac”, x+„ad”

Podstawowe operacje na plikach

- przypadek 2 – podzielenie pliku na skończoną ilość (4) elementów + własny prefix

```
[user@serwer ~]$ split -n 4 osoby.txt osoby
[user@serwer ~]$ ls
Dokumenty  lorem-ipsu[m].txt  Obrazy  osobyab  osobyad
liczby.txt Muzyka             osobyaa  osobyac  osoby.txt
```

- przypadek 3 – podzielenie pliku na skończoną ilość elementów (4) + własny prefix i suffix

```
[user@serwer ~]$ split -n 4 osoby.txt osoby- --additional-suffix=PART
[user@serwer ~]$ ls
Dokumenty  lorem-ipsu[m].txt  Obrazy  osoby-abPART  osoby-adPART
liczby.txt Muzyka            osoby-aaPART  osoby-acPART  osoby.txt
```


Podstawowe operacje na plikach

- komenda cut (wycinanie kolumn)
- przypadek 1

```
[user@serwer ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
[user@serwer ~]$ cat osoby.txt | cut -d ' ' -f1
user1
user2
user3
[user@serwer ~]$ cat osoby.txt | cut -d ' ' -f2
grupa1
grupa2
grupa3
```

Podstawowe operacje na plikach

- przypadek 2 – wycięcie dwóch pierwszych kolumn

```
[user@serwer ~]$ cat osoby.txt
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@serwer ~]$ cat osoby.txt | cut -d ' ' -f1,2
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@serwer ~]$ cat osoby.txt | cut -d ' ' -f2,1
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

Podstawowe operacje na plikach

- przypadek 3 – wyświetlenie zawartości pliku **od drugiej kolumny do ostatniej**

```
[user@serwer ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
[user@serwer ~]$ cat osoby.txt | cut -c2-
ser1 grupa1
ser2 grupa2
ser3 grupa3
```

- przypadek 4 – wyświetlenie zawartości pliku **od drugiej do piątej kolumny**

```
[user@serwer ~]$ cat osoby.txt | cut -c2-5
ser1
ser2
ser3
```

Podstawowe operacje na plikach

- komenda head (wyświetlenie określonej ilości linii)
- przypadek 1

```
[user@serwer ~]$ cat osoby.txt
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@serwer ~]$ cat osoby.txt | head -n2
```

```
user1 grupa1
```

```
user2 grupa2
```

```
[user@serwer ~]$ cat -n osoby.txt | head -n2
```

```
1 user1 grupa1
```

```
2 user2 grupa2
```

```
[user@serwer ~]$ cat -n osoby.txt | head -2
```

```
1 user1 grupa1
```

```
2 user2 grupa2
```

} „n2” lub 2 → ten sam efekt

Podstawowe operacje na plikach

- przypadek 2

```
[user@serwer ~]$ cat -n liczby.txt | head
1 9344 3606 19119 17592 8989
2 18640
3 22821 11210
4 24425 30285 23554
5 2420 13655 26468 26018 11333 30833
6 5527 10346 23301 20038 29128 1457
7 15063
8 5976
9 11693 29556
10
```

Komenda **tail** wyświetla określoną ilość linii od końca, konstrukcja analogiczna do head

Podstawowe operacje na plikach

- komenda **grep** (wycięcie linii z pasującym słowem)

```
[user@serwer ~]$ cat osoby.txt
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@serwer ~]$ cat osoby.txt | grep 'user'
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@serwer ~]$ cat osoby.txt | grep 'user1'
```

```
user1 grupa1
```

Podstawowe operacje na plikach

- komenda sed (szeroko rozumiana edycja tekstu)
- przykład 1 – zmiana łańcucha user na USER → każde (parametr „g”) wystąpienie w linii łańcucha „user” zostanie zmienione na łańcuch „USER”. Wynik zamiany jest tylko wyświetlony, ale nie zapisany do pliku.

```
[user@server ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
[user@server ~]$ cat osoby.txt | sed -e 's/user/USER/g'
USER1 grupa1
USER2 grupa2
USER3 grupa3
[user@server ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
```

brak parametru g → zamiana tylko pierwszego wystąpienia łańcucha „user” w linii

Podstawowe operacje na plikach

- przykład 2 – zmiana łańcucha user na USER → każde (parametr „g”) wystąpienie w linii łańcucha „user” zostanie zmienione na łańcuch „USER”. Wynik zamiany **zostaje zapisany do pliku**.

```
[user@server ~]$ cat osoby.txt
user1 grupa1
user2 grupa2
user3 grupa3
[user@server ~]$ sed -e 's/user/USER/g' osoby.txt > osoby1.txt
[user@server ~]$
[user@server ~]$ cat osoby1.txt
USER1 grupa1
USER2 grupa2
USER3 grupa3
```

Podstawowe operacje na plikach

- przykład 3 – zamiana łańcucha „user” na „user_abc”. UWAGA! Znak & (czyt. appersand) odwołuje się do łańcucha wejściowego i może być wykorzystany do zamiany:

```
[user@server ~]$ cat osoby.txt
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@server ~]$ cat osoby.txt | sed -e 's/user/&_abc/g'
```

```
user_abc1 grupa1
```

```
user_abc2 grupa2
```

```
user_abc3 grupa3
```

user

- analogicznie:

```
[user@server ~]$ cat osoby.txt | sed -e 's/user/&&_abc/g'
```

```
useruseruser_abc1 grupa1
```

```
useruseruser_abc2 grupa2
```

```
useruseruser_abc3 grupa3
```

Podstawowe operacje na plikach

- przykład 4 – usunięcie „pustych linii”

`sed '/^$/d'`

d - parametr odpowiedzialny za usunięcie

łańcuch odpowiadający „pustej linii” - ^ to początek linii, a \$ to koniec linii

```
[user@server ~]$ cat osoby.txt
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```

```
[user@server ~]$ cat osoby.txt | sed '/^$/d'
```

```
user1 grupa1
```

```
user2 grupa2
```

```
user3 grupa3
```


Podstawowe operacje na plikach

- przykład 5 – filtrowanie linii (wliczane są również „puste” linie) → przedział zamknięty

```
[user@server ~]$ cat -n osoby.txt
```

```
1 user1 grupa1
```

```
2
```

```
3 user2 grupa2
```

```
4
```

```
5 user3 grupa3
```

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

```
[user@server ~]$ cat -n osoby.txt | sed -n "2,5"p
```

```
2
```

```
3 user2 grupa2
```

```
4
```

```
5 user3 grupa3
```

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

```
[user@server ~]$ cat -n osoby.txt | sed -n '2,5'p
```

```
2
```

```
3 user2 grupa2
```

```
4
```

```
5 user3 grupa3
```

Podstawowe operacje na plikach

- przykład 6 – wyświetlenie pierwszego elementu z pierwszej wiersza

```
3  echo ""
4  cat tab.txt
5
6  echo ""
7  ile_wierszy=$(cat tab.txt | wc -l)
8  echo "ilość wierszy: "$ile_wierszy
9
10 echo ""
11 koll=$(cat tab.txt | head -1)
12 echo "pierwszy wiersz: "$koll
13
14 echo ""
15 ell=$(cat tab.txt | head -1 | cut -d ',' -f1)
16 echo "pierwszy element z pierwszego wiersza: "$ell
17
18 echo ""
19 ile_separatorow=$(cat tab.txt | head -1 | sed 's/^[^,]//g')
20 echo "ilość separatorów: "$ile_separatorow
21
22 echo ""
23 ile_separatorow=$(cat tab.txt | head -1 | sed 's/^[^,]//g' | wc -l)
24 ile_kolumn=$((ile_separatorow+1))
25 echo "ilość kolumn: "$ile_kolumn
26
```

a1,a2
b1,b2
c1,c2

ilość wierszy: 3

pierwszy wiersz: a1,a2

pierwszy element z pierwszego wiersza: a1

ilość separatorów: ,

ilość kolumn: 2

Podstawowe operacje na plikach

- przykład 7 – wyświetlenie wybranego elementu z pliku

```
3 echo ""
4 cat tab.txt
5
6 # ----- szukany element
7 nr_wier=2
8 nr_kol=2
9 # ----- szukany element
10
11 echo ""
12 nr_kol=f$nr_kol
13 element=$(cat tab.txt | sed -n "$nr_wier"p | cut -d ',' -"$nr_kol")
14 echo "element: "$element
```

a1,a2
b1,b2
c1,c2

element: b2



Architektura Oracle