



Niezbędnik Administratora Baz Danych

**Zbiór niezbędnych zapytań SQL do analizy
konfiguracji i optymalizacji silników
bazodanowych!**

Witaj!

Niniejszy eBook powstał z myślą by **ułatwić Ci pracę z bazami danych**. Informacje, które tutaj zebrałem będą pomocne zarówno dla **nowicjuszy, jak i doświadczonych administratorów**.

Pracuję jako DBA już prawie 10 lat. Regularnie sprawdzam i wyciągam z silników bazodanowych różnorakie informacje na **temat ich konfiguracji, wydajności czy ogólnego stanu**.

Doświadczenie pokazuje, że każdorazowe pisanie zapytania, czy nawet wyszukiwanie go w czeluściach internetu nie jest optymalne czasowo :)

Właśnie z tego powodu zebrałem tutaj wszystkie najbardziej wartościowe i pomocne SQL'ki dla **Oracle Database i PostgreSQL, czyli dwóch najpopularniejszych silników bazodanowych**.

Otrzymujesz gotowe zapytania, z przykładowymi wynikami. Dodatkowo przy każdym z nich umieściłem opis, co możemy wywnioskować z otrzymanych informacji.

ORACLE®

DATABASE

Sprawdzenie nazwy i daty utworzenia bazy danych

select name, created from v\$database;

	NAME	CREATED
1	MAINDB	20/10/20

Zaczynamy od prostego zapytania sprawdzającego nazwę i datę utworzenia bazy danych.

Warto zauważyć że kolumna CREATED, podaje nie datę uruchomienia silnika bazodanowego, lecz dzień w którym dana baza została utworzona przy pomocy narzędzia DBCA.

Sprawdzenie wersji bazy danych

select name, created from v\$database;

BANNER	BANNER_FULL
1 Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production Version 19.3.0.0.0	

Szybkie zapytanie do sprawdzenia wersji baz danych Oracle, na której pracujemy. Dostarczy Ci ono informacji także o wersji zainstalowanego patcha bazodanowego.

Sprawdzenie zainstalowanych w bazie komponentów

select * from DBA_REGISTRY;

COMP_ID	COMP_NAME	VERSION	VERSION_FULL	STATUS
1	CATALOG	Oracle Database Catalog Views	19.0.0.0.0 19.3.0.0.0	VALID
2	CATPROC	Oracle Database Packages and Types	19.0.0.0.0 19.3.0.0.0	VALID
3	RAC	Oracle Real Application Clusters	19.0.0.0.0 19.3.0.0.0	OPTION OFF
4	JAVAVM	JSERVER JAVA Virtual Machine	19.0.0.0.0 19.3.0.0.0	VALID
5	XML	Oracle XDK	19.0.0.0.0 19.3.0.0.0	VALID
6	CATJAVA	Oracle Database Java Packages	19.0.0.0.0 19.3.0.0.0	VALID
7	APS	OLAP Analytic Workspace	19.0.0.0.0 19.3.0.0.0	VALID
8	XDB	Oracle XML Database	19.0.0.0.0 19.3.0.0.0	VALID
9	OWM	Oracle Workspace Manager	19.0.0.0.0 19.3.0.0.0	VALID
10	CONTEXT	Oracle Text	19.0.0.0.0 19.3.0.0.0	VALID
11	ORDIM	Oracle Multimedia	19.0.0.0.0 19.3.0.0.0	VALID
12	SDO	Spatial	19.0.0.0.0 19.3.0.0.0	VALID
13	XOQ	Oracle OLAP API	19.0.0.0.0 19.3.0.0.0	VALID
14	DV	Oracle Database Vault	19.0.0.0.0 19.3.0.0.0	VALID
15	OLS	Oracle Label Security	19.0.0.0.0 19.3.0.0.0	VALID

Odpytanie widoku DBA_REGISTRY pozwala na sprawdzenie, które z komponentów opcjonalnych są dostępne w naszej instalacji silnika bazodanowego.

Sprawdzenie podstawowych informacji o instancji

```
select instance_name, status, to_char(STARTUP_TIME, 'dd-mon-yy hh24:mi:ss') as  
"STARTUP_TIME", host_name  
from v$instance;
```

	INSTANCE_NAME	STATUS	STARTUP_TIME	HOST_NAME
1	maindb	OPEN	12-paź-22 15:51:15	oracle-lab

Zapytanie pokaże Ci jak nazywa się instancja bazodanowa, w jakim jest aktualnie stanie, a także kiedy została uruchomiona i na jakim hoście.

Sprawdzanie kto ostatnio logował się do bazy

```
select username, count(*) "SESSIONS", trunc(last_call_et/3600) "IDLE_HOURS", module  
from v$session group by username, trunc(last_call_et/3600), module order by 4, 3, 1;
```

	USERNAME	SESSIONS	IDLE_HOURS	MODULE
1	(null)	6	0	KTSJ
2	(null)	6	0	MMON_SLAVE
3	SYS	1	0	SQL Developer
4	(null)	24	0	Streams
5	SYS	1	0	(null)
6	(null)	37	0	(null)

Tym razem przyjrzymy się użytkownikom naszej bazy, szybko możesz zweryfikować kto ostatnio się do niej logował i od jakiego czasu jego sesja nic nie robi. Dzięki temu zapytaniu ustalisz, które sesje na bazie danych "wiszą" od dłuższej chwili.

Zauważ że procesy silnika bazodanowego, również mają swoje sesje.

Sprawdzenie listy sesji na bazie

```
SELECT s.username,  
       s.osuser,  
       s.sid,  
       s.serial#,  
       p.spid,  
       s.lockwait,  
       s.status,  
       s.service_name,  
       s.machine,  
       s.program,  
       TO_CHAR(s.logon_time,'DD-MON-YYYY HH24:MI:SS') AS logon_time,  
       s.last_call_et AS last_call_et_secs,  
       s.module,  
       s.action,  
       s.client_info,  
       s.client_identifier  
FROM   v$session s,  
       v$process p  
WHERE  s.paddr = p.addr  
AND    s.username IS NOT NULL  
ORDER BY s.username, s.osuser;
```

	USERNAME	OSUSER	SID	SERIAL#	SPID	LOCKWAIT	STATUS	SERVICE_NAME	MACHINE	PROGRAM	LOGON_TIME	LAST_CALL_ET_SECS	MODULE
1	SYS	lukas	18	64431	2319	(null)	ACTIVE	maindb	MacBook-Air-Lukas.local	SQL Developer	12-PAZ-2022 20:40:52		SQL Developer
2	SYS	oracle	4	64741	1946	(null)	ACTIVE	SYS\$BACKGROUND	oracle-lab	oracle@oracle-lab (OFSD)	12-PAZ-2022 20:39:51	1016	(null)

Super przydatne zapytanie, pokazujące informacje o sesjach zalogowanych aktualnie do bazy danych Oracle. Znajdziesz tutaj info takie jak, nazwa użytkownika DB, numery namiarowe na sesje(SID, SERIAL#), nazwe serwisu przez który łączy się dana sesja, czy nazwę maszyny z której przychodzi połączenie.

To zapytanie to często pierwsza rzecz, której powinienes użyć aby zobaczyć kto aktualnie korzysta z Twojej bazy danych.

Top sesje po liczbie zapytań

```
SELECT NVL(a.username, '(oracle)') AS username,
       a.osuser,
       a.sid,
       a.serial#,
       c.value ,
       a.lockwait,
       a.status,
       a.module,
       a.machine,
       a.program,
       TO_CHAR(a.logon_Time,'DD-MON-YYYY HH24:MI:SS') AS logon_time
FROM   v$session a,
       v$sesstat c,
       v$statname d
WHERE  a.sid      = c.sid
AND    c.statistic# = d.statistic#
AND    d.name      = DECODE(c.value, 'READS', 'session logical reads',
                           'EXECS', 'execute count',
                           'CPU', 'CPU used by this session',
                           'CPU used by this session')
ORDER BY c.value DESC;
```

USERNAME	OSUSER	SID	SERIAL#	VALUE	LOCKWAIT	STATUS	MODULE	MACHINE	PROGRAM	LOGON_TIME
1 (oracle)	oracle	277	47183	624 (null)		ACTIVE	MMON_SLAVE	oracle-lab	oracle@oracle-lab (M004)	12-PAŻ-2022 20:40:12
2 (oracle)	oracle	37	56649	381 (null)		ACTIVE	MMON_SLAVE	oracle-lab	oracle@oracle-lab (M003)	12-PAŻ-2022 20:40:12
3 (oracle)	oracle	252	37774	298 (null)		ACTIVE	(null)	oracle-lab	oracle@oracle-lab (MMON)	12-PAŻ-2022 20:39:51

Powyższe zapytanie pomoże Ci w znalezieniu sesji które generują największą liczbę zapytań na bazie.

Pamiętaj, że tego typu informacja jest tylko jedną z składowych opisujących co dzieje się na bazie. Sesja generująca setki zapytań może delikatnie obciążać bazę, podczas gdy sesja z jednym dużym zapytaniem obciąży ją bardzo.

Lista parametrów które dziedziczy każda nowa sesja

```
SELECT sp.name,  
       sp.type,  
       sp.value,  
       sp.isises_modifiable,  
       sp.issys_modifiable,  
       sp.isinstance_modifiable  
FROM   v$system_parameter sp  
ORDER BY sp.name;
```

NAME	TYPE	VALUE	ISSES_MODIFIABLE	ISSYS_MODIFIABLE	ISINSTANCE_MODIFIABLE
1 active_instance_count	3	(null)	FALSE	FALSE	FALSE
2 adg_account_info_tracking	2	LOCAL	FALSE	FALSE	FALSE
3 adg_redirect_dml	1	FALSE	FALSE	IMMEDIATE	TRUE
4 allow_global_dblinks	1	FALSE	FALSE	IMMEDIATE	TRUE
5 allow_group_access_to_sga	1	FALSE	FALSE	FALSE	FALSE

Odpytując widok V\$SYSTEM_PARAMETER szybko ustalisz jakie parametry na start będzie miała ustawione każda nowa sesja tworzona na bazie danych Oracle.

Parametry które mają inną wartość niż w SPFILE

```
SELECT p.name,  
       i.instance_name AS sid,  
       p.value AS current_value,  
       sp.sid,  
       sp.value AS spfile_value  
FROM   v$spparameter sp,  
       v$parameter p,  
       v$instance i  
WHERE  sp.name = p.name  
AND    sp.value != p.value;
```

NAME	SID	CURRENT_VALUE	SID_1	SPFILE_VALUE
1 nls_language	maindb	POLISH	*	AMERICAN
2 nls_territory	maindb	POLAND	*	AMERICA
3 memory_target	maindb	1593835520	*	1583349760
4 control_files	maindb	/u01/oradata/MAINDB/controlfile/o1_mf_hryj655v_.ctl, /u01/oradata/MAINDB/control_MAINDB.ctl *		/u01/oradata/MAINDB/controlfile/o1_mf_hryj655v_.ctl
5 control_files	maindb	/u01/oradata/MAINDB/controlfile/o1_mf_hryj655v_.ctl, /u01/oradata/MAINDB/control_MAINDB.ctl *		/u01/oradata/MAINDB/control_MAINDB.ctl

Dobrze skonfigurowana baza danych powinna wykorzystywać plik SPFILE. Przychodzisz do nowego serwera, baza pracuje od dawna, chcesz sprawdzić czy konfiguracja z SPFILE pokrywa się z parametrami ustawionymi aktualnie w tzw. MEMORY instancji. Użycie powyższego zapytania rozwieje Twoje wątpliwości.

Lista parametrów z wartościami nondefault

```
SELECT name,  
       value  
FROM v$parameter  
WHERE isdefault = 'FALSE';
```

NAME	VALUE
1 processes	300
2 nls_language	POLISH
3 nls_territory	POLAND
4 service_names	maindb,APP_SERVICE
5 memory_target	1593835520
6 control_files	/u01/oradata/MAINDB/controlfile/01_mf_hryj655v_.ctl, /u01/oradata/MAINDB/control_MAINDB.ctl
7 control_file_record_keep_time	14
8 db_block_size	8192

Proste zapytanie pokaże Ci wszystkie parametry, których wartości odbiegają od ustawień domyślnych.

Lista tabel należących do użytkownika wykonującego zapytanie

```
SELECT  
  *  
FROM  
  user_tables  
ORDER BY  
  table_name;
```

TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS	PCT_FREE	PCT_USED	INI_TRANS
1 ACCESS\$	SYSTEM	(null)	(null)	VALID	10	40	1
2 ACLMV\$	SYSTEM	(null)	(null)	VALID	10	40	1
3 ACLMV\$_REFLOG	SYSTEM	(null)	(null)	VALID	10	40	1
4 ACLMVREFSTAT\$	SYSTEM	(null)	(null)	VALID	10	40	1
5 ACLMVSUBTBL\$	SYSTEM	(null)	(null)	VALID	10	40	1
6 ADMINAUTH\$	SYSTEM	(null)	(null)	VALID	10	40	1
7 ADO_IMPARAM\$	SYSAUX	(null)	(null)	VALID	10	(null)	1
8 ADO_IMSEGSTAT\$	SYSAUX	(null)	(null)	VALID	10	(null)	1

Chcesz sprawdzić do jakich tabel masz dostęp? Widok USER_TABLES Ci w tym pomoże!

Lista wszystkich tabel w bazie

```
SELECT
*
FROM
dba_tables
ORDER BY
owner, table_name;
```

OWNER	TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS	PCT_FREE	PCT_USED	IN
1 APPQOSSYS	WLM_CLASSIFIER_PLAN	SYSAUX	(null)	(null)	VALID	10	(null)	
2 APPQOSSYS	WLM_FEATURE_USAGE	SYSAUX	(null)	(null)	VALID	10	(null)	
3 APPQOSSYS	WLM_METRICS_STREAM	SYSAUX	(null)	(null)	VALID	10	(null)	
4 APPQOSSYS	WLM_MPA_STREAM	SYSAUX	(null)	(null)	VALID	10	(null)	
5 APPQOSSYS	WLM_VIOLATION_STREAM	SYSAUX	(null)	(null)	VALID	10	(null)	
6 AUDSYS	AUD\$UNIFIED	(null)	(null)	(null)	VALID	(null)	(null)	
7 C##DEVONE	TABONE	USERS	(null)	(null)	VALID	10	(null)	
8 C##DEVTHREE	TABTHREE	USERS	(null)	(null)	VALID	10	(null)	

Główne zapytanie dla DBA szukającego jakiejś tabelki. Spis wszystkich tabel w bazie, wykorzystując warunek w WHERE i LIKE możemy poszukać np. tabeli której znamy tylko kawałek nazwy.

Wyszukanie informacji o indeksach należących do danego użytkownika

```
SELECT table_owner,
       table_name,
       owner AS index_owner,
       index_name,
       tablespace_name,
       num_rows,
       status,
       index_type
FROM dba_indexes
WHERE table_owner = 'SYS'
ORDER BY table_owner, table_name, index_owner, index_name;
```

TABLE_OWNER	TABLE_NAME	INDEX_OWNER	INDEX_NAME	TABLESPACE_NAME	NUM_ROWS	STATUS	INDEX_TYPE
1 SYS	ACCESS\$	SYS	I_ACCESS1	SYSTEM	104875	VALID	NORMAL
2 SYS	ACLMV\$	SYS	I_ACLMV\$_1	SYSTEM	0	VALID	NORMAL
3 SYS	ACLMVREFSTAT\$	SYS	I_ACLMVREFSTAT\$_1	SYSTEM	0	VALID	NORMAL
4 SYS	ACLMVSUBTBL\$	SYS	I_ACLMVSUBTBL\$_1	SYSTEM	0	VALID	NORMAL
5 SYS	ADMINAUTH\$	SYS	I_ADMINAUTH1	SYSTEM	0	VALID	NORMAL
6 SYS	ADO_IMPARAM\$	SYS	C_ADO_IMPARAM	SYSAUX	1	VALID	NORMAL

Powyższe zapytanie pozwala lokalizować indeksy należące do danego użytkownika. W przykładzie wykonałem wyszukiwanie dla użytkownika administracyjnego SYS.

Sprawdzenie sesji wykorzystujących CPU najbardziej

```
SELECT se.username, ss.sid, ROUND (value/100) "CPU Usage"
FROM v$session se, v$sesstat ss, v$statname st
WHERE ss.statistic# = st.statistic#
AND name LIKE '%CPU used by this session%'
AND se.sid = ss.SID
AND se.username IS NOT NULL
ORDER BY value DESC;
```

	USERNAME	SID	CPU Usage
1	SYS	18	5
2	SYS	4	0
3	SYS	4	0
4	SYS	18	0
5	SYS	18	0
6	SYS	4	0

Szybki sposób na sprawdzenie, która sesja konsumuje najwięcej CPU.

Sprawdzenie listy obiektów INVALID w bazie

```
SELECT owner,
       object_type,
       object_name,
       status
FROM   dba_objects
WHERE  status = 'INVALID'
ORDER BY owner, object_type, object_name;
```

Zapytanie to pokaże Ci które obiekty w bazie danych wymagają kompilacji. Output nie ma, ponieważ na mojej bazie testowej wszystko jest VALID :)

Rozłożenie pamięci po różnych cache

```
SELECT component,  
       ROUND(current_size/1024/1024) AS current_size_mb,  
       ROUND(min_size/1024/1024) AS min_size_mb,  
       ROUND(max_size/1024/1024) AS max_size_mb  
FROM   v$sga_dynamic_components  
WHERE  current_size != 0  
ORDER BY component;
```

COMPONENT	CURRENT_SIZE_MB	MIN_SIZE_MB	MAX_SIZE_MB
1 DEFAULT buffer cache	400	400	400
2 java pool	16	16	16
3 large pool	16	16	16
4 Shared IO Pool	48	48	48
5 shared pool	336	336	336
6 streams pool	16	16	16

Powyższe zapytanie pokaże Ci jak rozkładana jest pamięć RAM pomiędzy poszczególne komponenty SGA. W sytuacji wysokiego obciążenia, zapytanie to potrafi naprowadzić Nas na to co aktualnie obciąża nam bazę i jakiego typu operacjami.

Top zapytania po odczytach

```
SELECT *  
FROM   (SELECT Substr(a.sql_text,1,50) sql_text,  
          Trunc(a.disk_reads/Decode(a.executions,0,1,a.executions))  
        reads_per_execution,  
          a.buffer_gets,  
          a.disk_reads,  
          a.executions,  
          a.sorts,  
          a.address  
        FROM   v$sqlarea a  
        ORDER BY 2 DESC);
```

SQL_TEXT	READS_PER_EXECUTION	BUFFER_GETS	DISK_READS	EXECUTIONS	SORTS
1 select count(*) num_enabled, sum(case optimizer_st	864	2637	864	1	1
2 select count(*) num_enabled, sum(case optimizer_st	864	2637	864	1	1
3 select owner,object_name from all_objects where ob	598	2961	1197	2	0
4 SELECT table_owner, table_name, owne	560	26619	560	1	1
5 select owner,object_name from all_objects where ob	467	2421	934	2	0
6 BEGIN sys.dbms_auto_report_internal.i_save_report	373	34897	747	2	0

Bardzo cenne zapytanie, pozwoli Ci odczytać które SQL wyciągają z bazy najwięcej danych, z podziałem na odczyty z dysku i z buffer cache. Bardzo pomocne przy diagnostyce problemów wydajnościowych i lokalizowaniu problematycznych zapytań.

Sprawdzenie kodu źródłowego dla obiektów

```
SELECT type, line, text
FROM dba_source
WHERE owner='SYS'
AND name='DBMS_ADR'
ORDER BY line;
```

TYPE	LINE	TEXT
1 PACKAGE	1	PACKAGE dbms_adr AS
2 PACKAGE BODY	1	PACKAGE BODY dbms_adr wrapped a000000 1 abcd abcd abcd abcd abcd abcd abcd abcd abcd abcd abc
3 PACKAGE	2	
4 PACKAGE	3	-- *****
5 PACKAGE	4	-- ODL message type constants
6 PACKAGE	5	-- These are for use in adr_home_t.log(msg_type => ...)
7 PACKAGE	6	-- *****
8 PACKAGE	7	
9 PACKAGE	8	log_msg_type_unknown CONSTANT INTEGER := 1; /* Unknown */
10 PACKAGE	9	log_msg_type_incident CONSTANT INTEGER := 2; /* Incident */
11 PACKAGE	10	log_msg_type_error CONSTANT INTEGER := 3; /* Error */

Mega przydatny widok bazodanowy DBA_SOURCE, w przykładzie wyciągam sobie definicje jednego z pakietów należących do użytkownika SYS. Przy pomocy tego widoku możesz przeszukiwać cały kod PL/SQL, który masz w bazie. W połączeniu z warunkiem WHERE można wyszukiwać na przykład czy dana funkcja odwołuje się do jakiejś tabelki. Można też sprawdzić czy jakiś widok jest używany przez jakikolwiek kod składowany w bazie.

Lokalizowanie plików trace dla sesji

```
SELECT s.sid,
       s.serial#,
       pa.value || '/' || LOWER(SYS_CONTEXT('userenv','instance_name')) ||
       '_ora_' || p.spid || '.trc' AS trace_file
FROM   v$session s,
       v$process p,
       v$parameter pa
WHERE  pa.name = 'user_dump_dest'
AND    s.paddr = p.addr
AND    s.audsid = SYS_CONTEXT('USERENV', 'SESSIONID');
```

SID	SERIAL#	TRACE_FILE
1	32	25178 /u01/app/oracle/product/19.0.0/dbhome_1/rdbms/log/maindb_ora_2332.trc

Szybkie odpytanie powyższych widoków, pozwoli Ci na sprawdzenie gdzie logują procesy które mają aktualnie włączony trace sesji.

Sprawdzenie przyrostu danych w bazie

```
SELECT
(select min(creation_time) from v$datafile) "Create Time",
(select name from v$database) "Database Name",
ROUND((SUM(USED.BYTES) / 1024 / 1024 ),2) || ' MB' "Database Size",
ROUND((SUM(USED.BYTES) / 1024 / 1024 ) - ROUND(FREE.P / 1024 / 1024 ),2) || ' MB' "Used Space",
ROUND(((SUM(USED.BYTES) / 1024 / 1024 ) - (FREE.P / 1024 / 1024 )) / ROUND(SUM(USED.BYTES) / 1024 / 1024
,2)*100,2) || '% MB' "Used in %",
ROUND((FREE.P / 1024 / 1024 ),2) || ' MB' "Free Space",
ROUND(((SUM(USED.BYTES) / 1024 / 1024 ) - ((SUM(USED.BYTES) / 1024 / 1024 ) - ROUND(FREE.P / 1024 / 1024
))))/ROUND(SUM(USED.BYTES) / 1024 / 1024,2)*100,2) || '% MB' "Free in %",
ROUND(((SUM(USED.BYTES) / 1024 / 1024 ) - (FREE.P / 1024 / 1024 ))/(select sysdate-min(creation_time) from
v$datafile),2) || ' MB' "Growth DAY",
ROUND(((SUM(USED.BYTES) / 1024 / 1024 ) - (FREE.P / 1024 / 1024 ))/(select sysdate-min(creation_time) from
v$datafile)/ROUND((SUM(USED.BYTES) / 1024 / 1024 ),2)*100,3) || '% MB' "Growth DAY in %",
ROUND(((SUM(USED.BYTES) / 1024 / 1024 ) - (FREE.P / 1024 / 1024 ))/(select sysdate-min(creation_time) from
v$datafile)*7,2) || ' MB' "Growth WEEK",
ROUND((((SUM(USED.BYTES) / 1024 / 1024 ) - (FREE.P / 1024 / 1024 ))/(select sysdate-min(creation_time) from
v$datafile)/ROUND((SUM(USED.BYTES) / 1024 / 1024 ),2)*100)*7,3) || '% MB' "Growth WEEK in %"
FROM (SELECT BYTES FROM V$DATAFILE
UNION ALL
SELECT BYTES FROM V$TEMPFILE
UNION ALL
SELECT BYTES FROM V$LOG) USED,
(SELECT SUM(BYTES) AS P FROM DBA_FREE_SPACE) FREE
GROUP BY FREE.P;
```

Create Time	Database Name	Database Size	Used Space	Used in %	Free Space	Free in %	Growth DAY	Growth DAY in %	Growth WEEK	Growth WEEK in %
1 19/04/17	MAINDB	6224 MB	5946 MB	95,53% MB	278,06 MB	4,47% MB	4,66 MB	,075% MB	32,63 MB	,524% MB

Zapytanie to pokaże Ci ile zajmuje Twoja baza danych, a także jak szybko przyrasta.

Sprawdzenie konfiguracji DATAFILE

```
set linesize 200
set pages 999
col "TBS" for a20
col "FILE" for a70
col "SIZE(M)" for 999999.99
col "USED(M)" for 999999.99
col "FREE(M)" for 999999.99
col "Used" for 999999.99
col "TotalUsed" for 999999.99
col "MaxSize(M)" for 999999.99
col "INC_MB" for 999999.99
```

```
SELECT df.tablespace_name "TBS",
       df.file_id "ID",
       df.file_name "FILE",
       df.STATUS,
       df.ONLINE_STATUS,
       Round(df.bytes / 1024 / 1024, 2) "SIZE(M)",
       Round(e.used_bytes / 1024 / 1024, 2) "USED(M)",
       Round(f.free_bytes / 1024 / 1024, 2) "FREE(M)",
       df.AUTOEXTENSIBLE as "AUTOEXT",
       df.increment_by*8/1024 inc_mb,
       Round(df.MAXBYTES / 1024 / 1024, 2) "MaxSize(M)",
       round(decode(df.MAXBYTES, 0, 0, (e.used_bytes / df.bytes) * 100),2) "%Used",
       round(decode(df.MAXBYTES, 0, 0, (e.used_bytes / df.MAXBYTES) * 100),2) "%TotalUsed"
FROM DBA_DATA_FILES DF,
     (SELECT file_id,
              Sum(Decode(bytes, NULL, 0, bytes)) used_bytes
      FROM dba_extents
      GROUP by file_id) E,
     (SELECT Max(bytes) free_bytes,
              file_id
      FROM dba_free_space
      GROUP BY file_id) f
WHERE e.file_id (+) = df.file_id
      AND df.file_id = f.file_id (+)
ORDER BY df.tablespace_name;
```

TBS	ID	FILE	STATUS	ONLINE_STATUS	SIZE(M)	USED(M)	FREE(M)	AUTOEXT	INC_MB	MaxSize(M)	%Used	%TotalUsed
1 SYSAUX	3	/u01/oradata/MAINDB/datafile/o1_mf_sysaux_jd7ho36m_.dbf	AVAILABLE	ONLINE	590	558,06	28	YES	10	32767,98	94,59	1,7
2 SYSTEM	1	/u01/oradata/MAINDB/datafile/o1_mf_system_jd7ho35b_.dbf	AVAILABLE	SYSTEM	910	904,75	4	YES	10	32767,98	99,42	2,76
3 UNDOTBS1	4	/u01/oradata/MAINDB/datafile/o1_mf_undotbs1_jd7ho37v_.dbf	AVAILABLE	ONLINE	280	38,25	55	YES	5	32767,98	13,66	0,12
4 USERS	7	/u01/oradata/MAINDB/datafile/o1_mf_users_jd7ho39g_.dbf	AVAILABLE	ONLINE	5	1,88	2,13	YES	1,25	32767,98	37,5	0,01

Tym razem spore zapytanie, całkiem skomplikowane, ale świetnie pokazujące wielkość poszczególnych DATAFILE i ich zajętość. Pierwsza linia diagnostyki, kiedy dostajesz info o zapchanym TABLESPACE.

Sprawdzenie konfiguracji TABLESPACE

SET PAGESIZE 140

COLUMN used_pct FORMAT A11

```
SELECT tablespace_name,  
       size_mb,  
       free_mb,  
       max_size_mb,  
       max_free_mb,  
       TRUNC((max_free_mb/max_size_mb) * 100) AS free_pct,  
       RPAD(' ' || RPAD('X',ROUND((max_size_mb-max_free_mb)/max_size_mb*10,0),  
'X'),11,'-') AS used_pct  
FROM (   
  SELECT a.tablespace_name,  
         b.size_mb,  
         a.free_mb,  
         b.max_size_mb,  
         a.free_mb + (b.max_size_mb - b.size_mb) AS max_free_mb  
  FROM (SELECT tablespace_name,  
               TRUNC(SUM(bytes)/1024/1024) AS free_mb  
        FROM dba_free_space  
        GROUP BY tablespace_name) a,  
        (SELECT tablespace_name,  
               TRUNC(SUM(bytes)/1024/1024) AS size_mb,  
               TRUNC(SUM(GREATEST(bytes,maxbytes))/1024/1024) AS max_size_mb  
        FROM dba_data_files  
        GROUP BY tablespace_name) b  
  WHERE a.tablespace_name = b.tablespace_name  
 )  
ORDER BY tablespace_name;
```

	TABLESPACE_NAME	SIZE_MB	FREE_MB	MAX_SIZE_MB	MAX_FREE_MB	FREE_PCT	USED_PCT
1	SYSAUX	590	29	32767	32206	98	-----
2	SYSTEM	910	4	32767	31861	97	-----
3	UNDOTBS1	280	240	32767	32727	99	-----
4	USERS	5	2	32767	32764	99	-----

Tutaj uproszczona wersja poprzedniego zapytania, tym razem pozyskasz informacje o zajętości miejsca, bez wchodzenia w szczegóły. Szybkie sprawdzenie co dzieje się w TABLESPACE.

Statystyka produkcji REDO

```
SELECT A.*,  
Round(A.NUMBER_of_LOGS*B.AVG#/1024/1024) Daily_Avg_Mb  
FROM  
(  
SELECT  
To_Char(First_Time,'YYYY-MM-DD') DAY,  
Count(1) NUMBER_OF_LOGS,  
Min(RECID) MIN_RECID,  
Max(RECID) MAX_RECID  
FROM  
v$log_history  
GROUP  
BY To_Char(First_Time,'YYYY-MM-DD')  
ORDER  
BY 1 DESC  
) A,  
(  
SELECT  
Avg(BYTES) AVG#,  
Count(1) NUMBER_of_LOGS,  
Max(BYTES) Max_Bytes,  
Min(BYTES) Min_Bytes  
FROM  
v$log  
) B  
;
```

DAY	NUMBER_OF_LOGS	MIN_RECID	MAX_RECID	DAILY_AVG_MB
1 2022-10-12	1	41	41	300
2 2021-06-11	15	26	40	4500
3 2021-06-09	9	17	25	2700
4 2021-03-13	6	11	16	1800
5 2021-03-10	1	10	10	300
6 2020-11-04	1	9	9	300
7 2020-10-29	1	8	8	300
8 2020-10-20	7	1	7	2100

Powyżej widzisz bardzo często używane zapytanie, do codziennej obserwacji wydajności bazy danych. Zakłada się, że poprawnie skonfigurowane REDO nie powinny zmieniać się częściej niż raz na 20 minut. Łatwo można wyliczyć, że ilość switch REDO większa niż 72 w ciągu doby jest sygnałem do zwiększenia ich wielkości.

Miejsce zajmowane przez tabele danego użytkownika

```
COLUMN TABLE_NAME FORMAT A32  
COLUMN OBJECT_NAME FORMAT A32  
COLUMN OWNER FORMAT A10
```

```
SELECT  
  owner,  
  table_name,  
  TRUNC(sum(bytes)/1024/1024) Meg,  
  ROUND( ratio_to_report( sum(bytes) ) over () * 100) Percent  
FROM  
(SELECT segment_name table_name, owner, bytes  
  FROM dba_segments  
  WHERE segment_type IN ('TABLE', 'TABLE PARTITION', 'TABLE SUBPARTITION')  
  UNION ALL  
  SELECT i.table_name, i.owner, s.bytes  
  FROM dba_indexes i, dba_segments s  
  WHERE s.segment_name = i.index_name  
  AND s.owner = i.owner  
  AND s.segment_type IN ('INDEX', 'INDEX PARTITION', 'INDEX SUBPARTITION')  
  UNION ALL  
  SELECT l.table_name, l.owner, s.bytes  
  FROM dba_lobs l, dba_segments s  
  WHERE s.segment_name = l.segment_name  
  AND s.owner = l.owner  
  AND s.segment_type IN ('LOBSEGMENT', 'LOB PARTITION')  
  UNION ALL  
  SELECT l.table_name, l.owner, s.bytes  
  FROM dba_lobs l, dba_segments s  
  WHERE s.segment_name = l.index_name  
  AND s.owner = l.owner  
  AND s.segment_type = 'LOBINDEX')  
WHERE owner in UPPER('&owner')  
GROUP BY table_name, owner  
HAVING SUM(bytes)/1024/1024 > 10 /* Ignore really small tables */  
ORDER BY SUM(bytes) desc  
;
```

	OWNER	TABLE_NAME	MEG	PERCENT
1	SYS	IDL_UB1\$	394	59
2	SYS	WRI\$_EMX_FILES	49	7
3	SYS	SOURCE\$	44	7
4	SYS	OBJ\$	31	5
5	SYS	IDL_UB2\$	30	5
6	SYS	AW\$AWXML	24	4
7	SYS	ARGUMENT\$	23	3
8	SYS	JAVA\$MC\$	19	3
9	SYS	DEPENDENCY\$	19	3
10	SYS	METASTYLESHEET	13	2
11	SYS	COL\$	13	2
12	SYS	WRH\$_SQL_PLAN	11	2

Zapytanie wymaga podania nazwy użytkownika, którego chcesz sprawdzić. Pokaże Ci ono listę jego obiektów i ich wielkość. Zauważ, że do wielkości danej tabeli dodane są także jej indeksy i LOB'y.

Zapotrzebowanie na UNDO

```
col "ACTUAL UNDO SIZE [MByte]" for 9999999999
col "UNDO RETENTION [Sec]" for a20
col "OPTIMAL UNDO RETENTION [Sec]" for 9999999999
SELECT d.undo_size/(1024*1024) "ACTUAL UNDO SIZE [MB]",
SUBSTR(e.value,1,25) "UNDO RETENTION [Sec]",
(TO_NUMBER(e.value) * TO_NUMBER(f.value) *
g.undo_block_per_sec) / (1024*1024)
"NEEDED UNDO SIZE [MB]"
FROM (
SELECT SUM(a.bytes) undo_size
FROM v$datafile a,
v$tablespace b,
dba_tablespaces c
WHERE c.contents = 'UNDO'
AND c.status = 'ONLINE'
AND b.name = c.tablespace_name
AND a.ts# = b.ts#
) d,
v$parameter e,
v$parameter f,
(
SELECT MAX(undoblks/((end_time-begin_time)*3600*24))
undo_block_per_sec
FROM v$undostat
) g
WHERE e.name = 'undo_retention'
AND f.name = 'db_block_size';
```

	ACTUAL UNDO SIZE [MB]	UNDO RETENTION [Sec]	NEEDED UNDO SIZE [MB]
1	3400 900		3,7734375

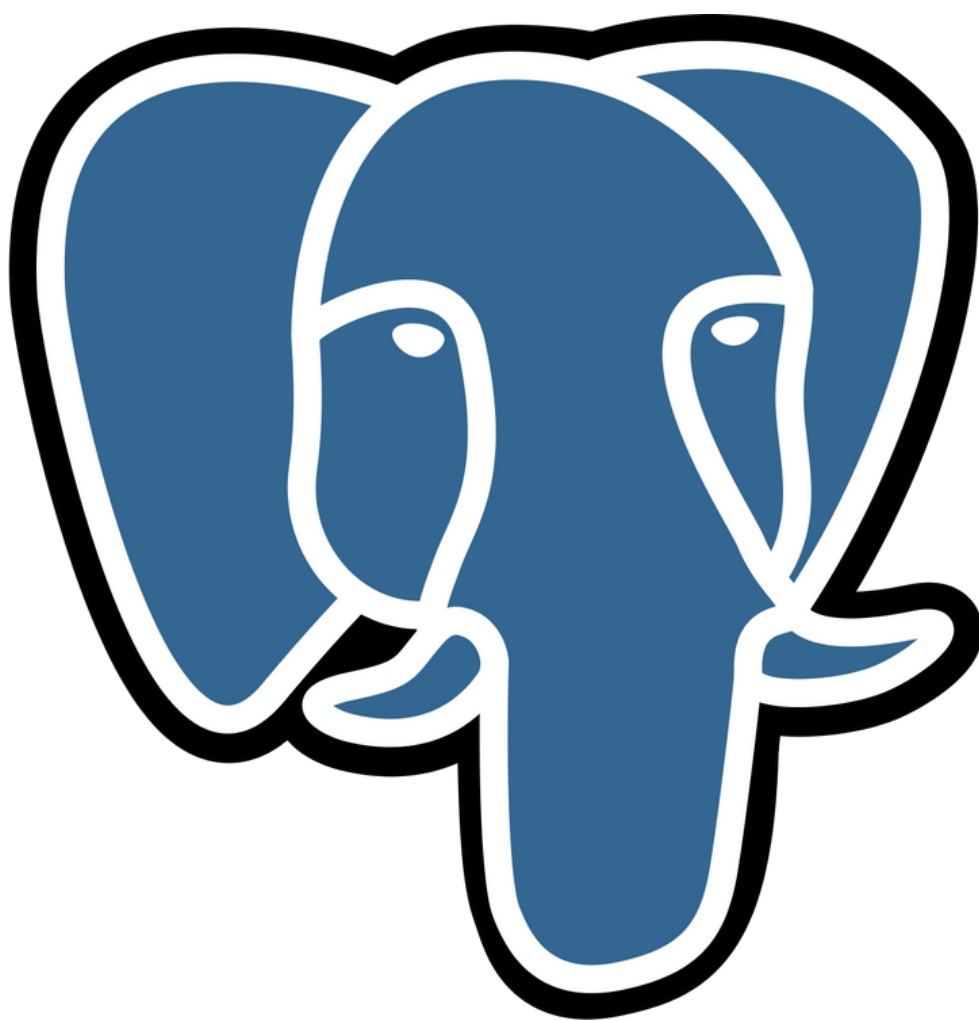
Szybki SELECT, którym zweryfikujesz sobie jakie jest zapotrzebowanie na przestrzeń UNDO w Twojej bazie. Będziesz wiedział dzięki niemu czy aktualne ustawienia są wystarczające.

Komplet uprawnień danego użytkownika

```
SELECT
PRIVILEGE,
OBJ_OWNER,
OBJ_NAME,
USERNAME,
LISTAGG(GRANT_TARGET, ',') WITHIN GROUP (ORDER BY GRANT_TARGET) AS GRANT_SOURCES, -- Lists the sources of the permission
MAX(ADMIN_OR_GRANT_OPT) AS ADMIN_OR_GRANT_OPT, -- MAX acts as a Boolean OR by picking 'YES' over 'NO'
MAX(HIERARCHY_OPT) AS HIERARCHY_OPT -- MAX acts as a Boolean OR by picking 'YES' over 'NO'
FROM (
-- Gets all roles a user has, even inherited ones
WITH ALL_ROLES_FOR_USER AS (
SELECT DISTINCT CONNECT_BY_ROOT GRANTEE AS GRANTED_USER, GRANTED_ROLE
FROM DBA_ROLE_PRIVS
CONNECT BY GRANTEE = PRIOR GRANTED_ROLE
)
SELECT
PRIVILEGE,
OBJ_OWNER,
OBJ_NAME,
USERNAME,
REPLACE(GRANT_TARGET, USERNAME, 'Direct to user') AS GRANT_TARGET,
ADMIN_OR_GRANT_OPT,
HIERARCHY_OPT
FROM (
-- System privileges granted directly to users
SELECT PRIVILEGE, NULL AS OBJ_OWNER, NULL AS OBJ_NAME, GRANTEE AS USERNAME, GRANTEE AS GRANT_TARGET, ADMIN_OPTION AS ADMIN_OR_GRANT_OPT,
NULL AS HIERARCHY_OPT
FROM DBA_SYS_PRIVS
WHERE GRANTEE IN (SELECT USERNAME FROM DBA_USERS)
UNION ALL
-- System privileges granted users through roles
SELECT PRIVILEGE, NULL AS OBJ_OWNER, NULL AS OBJ_NAME, ALL_ROLES_FOR_USER.GRANTED_USER AS USERNAME, GRANTEE AS GRANT_TARGET, ADMIN_OPTION AS
ADMIN_OR_GRANT_OPT, NULL AS HIERARCHY_OPT
FROM DBA_SYS_PRIVS
JOIN ALL_ROLES_FOR_USER ON ALL_ROLES_FOR_USER.GRANTED_ROLE = DBA_SYS_PRIVS.GRANTEE
UNION ALL
-- Object privileges granted directly to users
SELECT PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME AS OBJ_NAME, GRANTEE AS USERNAME, GRANTEE AS GRANT_TARGET, GRANTABLE, HIERARCHY
FROM DBA_TAB_PRIVS
WHERE GRANTEE IN (SELECT USERNAME FROM DBA_USERS)
UNION ALL
-- Object privileges granted users through roles
SELECT PRIVILEGE, OWNER AS OBJ_OWNER, TABLE_NAME AS OBJ_NAME, GRANTEE AS USERNAME, ALL_ROLES_FOR_USER.GRANTED_ROLE AS GRANT_TARGET,
GRANTABLE, HIERARCHY
FROM DBA_TAB_PRIVS
JOIN ALL_ROLES_FOR_USER ON ALL_ROLES_FOR_USER.GRANTED_ROLE = DBA_TAB_PRIVS.GRANTEE
) ALL_USER_PRIVS
-- Adjust your filter here
WHERE USERNAME = 'SYS'
) DISTINCT_USER_PRIVS
GROUP BY
PRIVILEGE,
OBJ_OWNER,
OBJ_NAME,
USERNAME
;
```

	PRIVILEGE	OBJ_OWNER	OBJ_NAME	USERNAME	GRANT_SOURCES
1	SELECT	DVSYS	DBA_DV_STATUS	SYS	Direct to user
2	SELECT	OUTLN	OL\$	SYS	Direct to user
3	SELECT	OUTLN	OL\$HINTS	SYS	Direct to user
4	SELECT	OUTLN	OL\$NODES	SYS	Direct to user

Podstawiasz pod WHERE USERNAME nazwę użytkownika, który Cię interesuje... i dostajesz spis wszystkich uprawnień przez niego posiadanych.



Listowanie baz

```
select oid as database_id,  
       datname as database_name,  
       datallowconn as allow_connect,  
       datconnlimit as connection_limit  
from pg_database  
order by oid;
```

database_id oid	database_name name	allow_connect boolean	connection_limit integer
1	template1	true	-1
14396	template0	false	-1
16399	uatdb1	true	-1
16400	devdb1	true	-1
16439	postgres	true	-1
16644	webappdev	true	-1
25387	pgbench	true	-1

Proste zapytanie do sprawdzenia jakie bazy mamy dostępne w danym PGDATA.

Listowanie schematów

```
select s.nspname as table_schema,  
       s.oid as schema_id,  
       u.username as owner  
from pg_catalog.pg_namespace s  
join pg_catalog.pg_user u on u.usesysid = s.nspowner  
order by table_schema;
```

table_schema name	schema_id oid	owner name
information_s...	14097	postgres
jacek	16505	jacek
jakis_schemat	16504	postgres
pg_catalog	11	postgres
pg_temp_1	12314	postgres
pg_toast	99	postgres
pg_toast_tem...	12315	postgres
public	2200	postgres

Proste zapytanie do sprawdzenia jakie schematy mamy dostępne w danej bazie.

Listowanie tabel - z pominięciem systemowych

```
select table_schema,  
       table_name  
from information_schema.tables  
where table_schema not in ('information_schema', 'pg_catalog')  
      and table_type = 'BASE TABLE'  
order by table_schema,  
       table_name;
```

table_schema name	table_name name
jacek	jakas_tabela
public	after_backup
public	new
public	zarobki

Ten SELECT pozwoli pozyskać Ci informacje na temat tabel dostępnych w danej bazie.

Listowanie tabel w schemacie

```
select t.table_name  
from information_schema.tables t  
where t.table_schema = 'jacek'  
      and t.table_type = 'BASE TABLE'  
order by t.table_name;
```

table_name name
jakas_tabela

Można również wylistować tabelki tylko z konkretnego schematu.

Listowanie 10 największych tabel w bazie

```
select schemaname as table_schema,  
       relname as table_name,  
       pg_size_pretty(pg_total_relation_size(relid)) as total_size,  
       pg_size_pretty(pg_relation_size(relid)) as data_size,  
       pg_size_pretty(pg_total_relation_size(relid) - pg_relation_size(relid))  
       as external_size  
from pg_catalog.pg_statio_user_tables  
order by pg_total_relation_size(relid) desc,  
        pg_relation_size(relid) desc  
limit 10;
```

table_schema name	table_name name	total_size text	data_size text	external_size text
public	new	40 kB	8192 bytes	32 kB
public	after_backup	8192 bytes	8192 bytes	0 bytes
public	zarobki	8192 bytes	8192 bytes	0 bytes
jacek	jakas_tabela	0 bytes	0 bytes	0 bytes

Zapytanie dzięki któremu dowiesz się jakie są największe tabelki, z dokładnym rozbićciem na ich składowe.

Podliczenie ilości tabel o poszczególnych wielkościach

```
select row_count,  
       count(*) as tables  
from (  
  select c.relname as table_name,  
         n.nspname as table_schema,  
         case when c.reltuples > 1000000000 then '1b rows and more'  
              when c.reltuples > 1000000 then '1m - 1b rows'  
              when c.reltuples > 1000 then '1k - 1m rows'  
              when c.reltuples > 100 then '100 - 1k rows'  
              when c.reltuples > 10 then '10 - 100 rows'  
              else '0 - 10 rows' end as row_count,  
         c.reltuples as rows  
  from pg_class c  
  join pg_namespace n on n.oid = c.relnamespace  
  where c.relkind = 'r'  
  -- and n.nspname not in ('pg_catalog', 'information_schema')  
) itv  
group by row_count  
order by max(rows);
```

row_count text	tables bigint
0 - 10 rows	47
10 - 100 rows	6
100 - 1k rows	16
1k - 1m rows	5

Tym razem sprawdzamy wielkość tabel, po ilości przetrzymywanych wierszy. Całość jest grupowana tak, żebyśmy widzieli jakiego "gabarytu" tabelki występują w naszej bazie.

Listowanie pustych tabel w bazie

```
select n.nspname as table_schema,  
       c.relname as table_name  
from pg_class c  
join pg_namespace n on n.oid = c.relnamespace  
where c.relkind = 'r'  
      and n.nspname not in ('information_schema','pg_catalog')  
      and c.reltuples = 0  
order by table_schema,  
         table_name;
```

table_schema name	table_name name
jacek	jakas_tabela
public	after_backup
public	zarobki

Prosty SELECT do lokalizowania pustych tabel. Pomaga utrzymywać ogólny porządek w strukturze danych.

Listowanie tabel posortowanych po ilości wierszy

```
select n.nspname as table_schema,  
       c.relname as table_name,  
       c.reltuples as rows  
from pg_class c  
join pg_namespace n on n.oid = c.relnamespace  
where c.relkind = 'r'  
      and n.nspname not in ('information_schema','pg_catalog')  
order by c.reltuples desc;
```

table_schema name	table_name name	rows real
public	new	3
public	after_backup	0
jacek	jakas_tabela	0
public	zarobki	0

Lista tabel, posortowana od największych po liczbie wierszy.

Sprawdzenie najczęściej używanych typów danych

```
with stats as (  
  select count(distinct col.table_schema || '.' || col.table_name)  
    as sum_tables,  
    count(col.column_name) as sum_columns  
  from information_schema.columns col  
  join information_schema.tables tab  
    on col.table_schema = tab.table_schema  
    and col.table_name = tab.table_name  
  where tab.table_schema not in ('information_schema', 'pg_catalog')  
    and tab.table_type = 'BASE TABLE'  
)  
select col.udt_name as typename,  
  count(distinct col.table_schema || '.' || col.table_name) as tables,  
  round(100.0*  
    count(distinct col.table_schema || '.' || col.table_name)  
    / (select sum_tables from stats)  
    , 2) as percent_tables,  
  count(col.column_name) as columns,  
  round(100.0*count(col.column_name)  
    / (select sum_columns from stats)  
    , 2) as percent_column  
from information_schema.columns col  
join information_schema.tables tab  
  on col.table_schema = tab.table_schema  
  and col.table_name = tab.table_name  
where tab.table_schema not in ('information_schema', 'pg_catalog')  
  and tab.table_type = 'BASE TABLE'  
group by col.udt_name  
order by columns desc,  
  tables desc;
```

typename name	tables bigint	percent_tables numeric	columns bigint	percent_column numeric
int4	4	100.00	6	100.00

Przydatne zapytanie przy planowaniu migracji baz danych. Pozwoli Ci ono sprawdzić jakie typy danych występują i w jakiej ilości w bazie.

Szukanie tabeli po konkretnych znakach w nazwie

```
select table_schema,  
       table_name  
from information_schema.tables  
where table_name like '%after%'  
       and table_schema not in ('information_schema', 'pg_catalog')  
       and table_type = 'BASE TABLE'  
order by table_name,  
       table_schema;
```

table_schema name	table_name name
public	after_backup

Wyszukiwanie tabel po kawałku nazwy. Przydatne kiedy nie masz dokładnych namiarów na tabelkę.

Listowanie user w bazie

```
select usesysid as user_id,  
       username as username,  
       usesuper as is_superuser,  
       passwd as password_md5,  
       valuntil as password_expiration  
from pg_shadow  
order by username;
```

user_id oid	username name	is_superuser boolean	password_md5 text
16650	developer_user...	false	md5047fadd4947f85725dc20b20c0268e...
16651	developer_user...	false	md51e71d79c9de76c1b71219b7e78603...
16401	jacek	false	md5848536d2e77e92dae5900a783b850f...
16512	magda	false	[null]

Dzięki temu sprawdzić jakich użytkowników masz na bazie. Pokazuje również zakodowane w md5 hasło, danego użytkownika.

Listowanie sesji na bazie danych

```
select pid as process_id,  
       username as username,  
       datname as database_name,  
       client_addr as client_address,  
       application_name,  
       backend_start,  
       state,  
       state_change  
from pg_stat_activity;
```

process_id integer	username name	database_name name	client_address inet	application_name text	backend_start timestamp with time zone
217	[null]	[null]	[null]		2022-10-13 11:43:10.686...
220	postgres	[null]	[null]		2022-10-13 11:43:10.688...
221	postgres	postgres	192.168.1.15	pgAdmin 4 - DB:p...	2022-10-13 11:43:15.707...
223	postgres	postgres	192.168.1.15	pgAdmin 4 - CON...	2022-10-13 11:43:47.980...
215	[null]	[null]	[null]		2022-10-13 11:43:10.684...
214	[null]	[null]	[null]		2022-10-13 11:43:10.687...
216	[null]	[null]	[null]		2022-10-13 11:43:10.685...

Bardzo przydatne zapytanie i często używane. Pokazuje Ci aktualne sesje zalogowane, dostajesz info na temat hostname/IP klienta, a także nazwę programu którym podpiną się do bazy danych.

Sprawdzenie ostatniego zapytania danej sesji

```
select pid,  
       username as username,  
       datname as database_name,  
       query,  
       application_name,  
       backend_start,  
       state,  
       state_change  
from pg_stat_activity  
where pid = '221';
```

pid integer	username name	database_name name	query text	application_name text
221	postgres	postgres	/*pga4dash*/ SELECT 'session_stats' A...	pgAdmin 4 - DB:postgres

W tym zapytaniu możemy sprawdzić co ostatnio robiła dana sesja. Przykład uzupełniłem id swojej sesji i widać w wyniku zapytanie z przykładu. To ono dla mnie będzie tym ostatnio wykonywanym.

Jestem przekonany, że powyższe zapytania pomogą
Tobie w pracy z bazami danych!

Jeżeli potrzebujesz więcej ciekawych informacji z
zakresu administracji baz danych zajrzyj na:
<https://dbadmin.net.pl>

Do usłyszenia,
Łukasz



<https://dbadmin.net.pl>