



EINFÜHRUNG VON POSTGRESQL

Eine Betrachtung auf der Zeitachse
Bernd Patolla

Inhaltsverzeichnis

Kurzvorstellung In & Out AG	3
Reise durch die Zeit	6
Detailkonzept	7
Tools	11
Offene Punkte	24

Kurzvorstellung

Kennzahlen und Branchen Schwerpunkte



Gründungsjahr

1993



Anzahl Mitarbeitende:

14



Umsatz:

CHF 7 Mio.



Schwerpunkt Branchen:
(alphabetische Reihenfolge)

Energie
Finanzinstitute
Gesundheitswesen
Handel
Industrie
Informatik
Öffentliche Verwaltung
Transport & Logistik
Versicherungen

Unsere Dienstleistungen im Überblick

Von der Analyse bis zur erfolgreichen Einführung



STRATEGIE &
TECHNOLOGIE



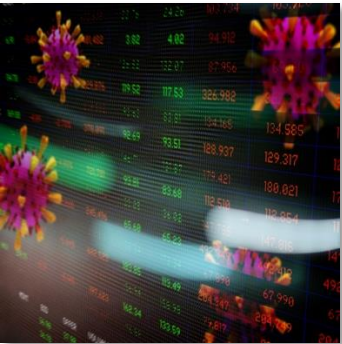
AUSSCHREIBUNGEN &
EVALUATIONEN



PERFORMANCE &
BENCHMARKING



KOSTEN &
WIRTSCHAFTLICHKEIT



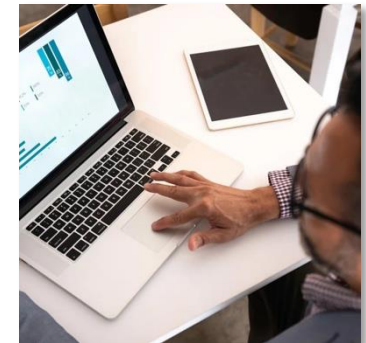
BUSINESS & IT
SERVICE CONTINUITY



IT- & SERVICE
MANAGEMENT



CLOUD &
DATA CENTER



PROJEKTBERATUNG &
-LEITUNG

Eine Reise in der Zeit

Start 2021

Start der Reise: Anfang 2021

- Allererste PostgreSQL Datenbank beim Kunden
- PoC einer neuen Applikation
- Anbieter verlangt PostgreSQL 12
- Installation und Konfiguration durch einen Applikations-Menschen der Oracle-DBA damals hatte kein Interesse an PGS
- Installation mit PostgreSQL-Repo und RPM
- Default-Verzeichnisse und Einstellungen
- Keine Integration in Umsysteme (Monitoring, Backup)
- Keine Security (pg_hba.conf, postgresql.conf)

Eine Reise in der Zeit

Jahr 2 (2022)

- Strategische Neu-Ausrichtung Technische Infrastrukturen
- März: Auftrag für Detailkonzept
- Mai: Start mit der Automatisierung
- Ende Mai: Erste Version vom Cookbook fertig
- CIS Security Settings / pgaudit
- Juli: Erste VM mit PGS automatisiert aufgebaut
- September: pgAdmin Einführung mit AD-Anbindung
- Weitere VMs

Eine Reise in der Zeit

Jahr 3 (2023)

- Einführung Pass-Utility (erst zentral, dann dezentral)
- `pg_stat_statements`
- `pg_profile`
- Weitere Applikationen auf PostgreSQL
- Cookbook weiter entwickelt
- Qlik Replicate (GG „Ersatz“ für PGS)
- Minor Updates (Patching) automatisiert

Eine Reise in der Zeit

Jahr 4 (2024)

- Backups und Datenbank-Cluster-Clones mittels Commvault
- Einführung pgBadger
- Ca. 40 VMs mit PGS, Tendenz stark wachsend
- Datenbank-Cluster und DBs werden mittels Scripts “automatisiert“ erstellt pro Applikation ein Script
- Migration von RHEL-7 auf RHEL-9

Eine Reise in der Zeit

Jahr 5 (2025)

- Ca. 58 VMs mit PGS, Tendenz stark wachsend
- PostgreSQL 14 bis 16 im Einsatz
- Alle VMs auf RHEL-9

Eine Reise in die Zukunft

Jahr 5 (2025 und später)

Planungen:

- GUI-Tool für Performance-Management anschauen
- Weitere Applikationen kommen (z.B. Archiv, Control-M)
- Grundsätzlich (so weit wie möglich): Ablösung von Oracle durch PostgreSQL
- DB-Datenmigrationen
- Planung und Engineering Major Update, Ziel: PGS 17 für alle

Detailkonzept

Ein paar wichtige Punkte

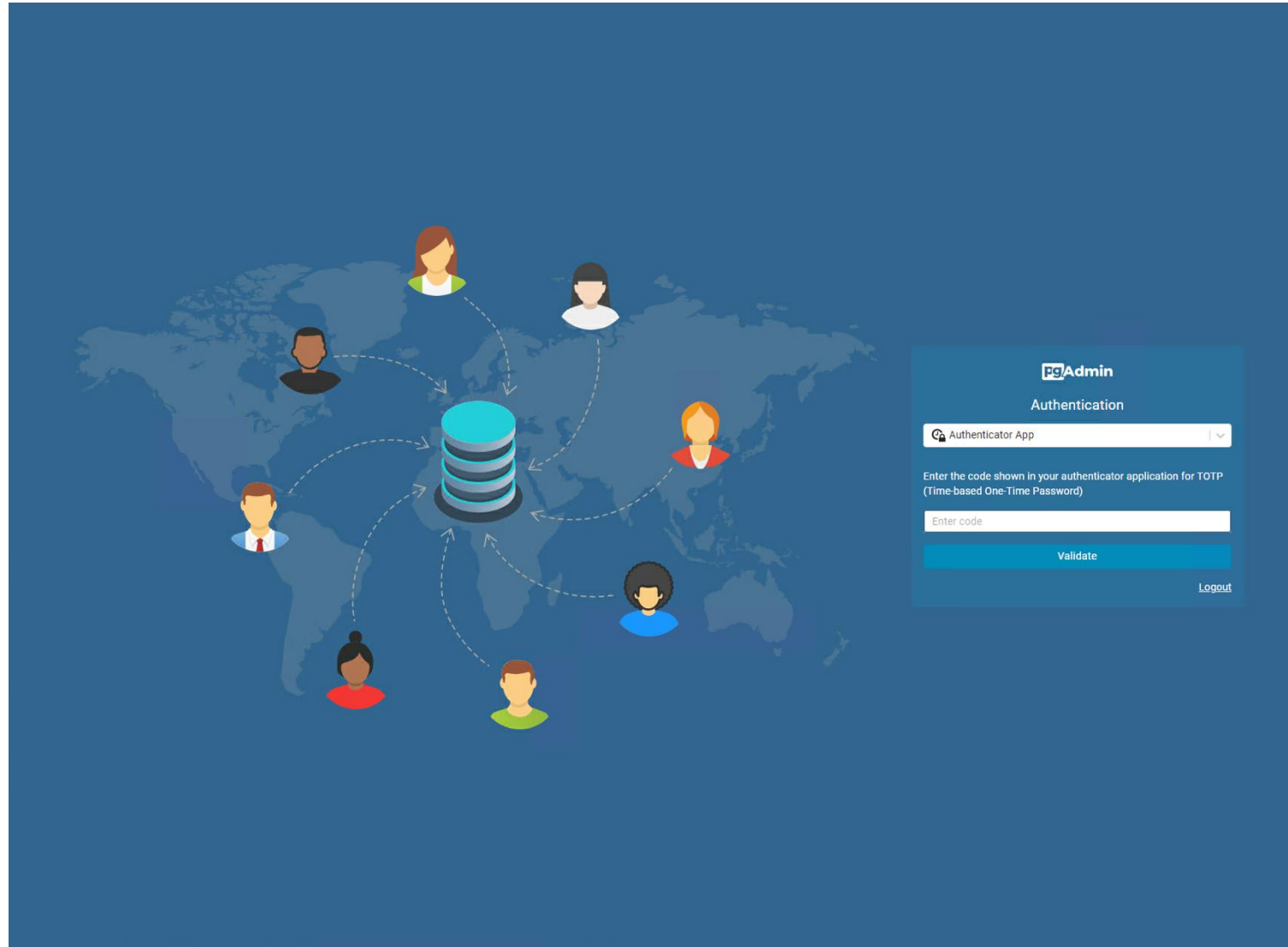
- Welche “Distribution“ soll benutzt werden?
- Virtualisierung
- Replikation
- Storage
 - Disks (-> Striping ja/nein?)
 - LVM?
 - Filesystem(e)
 - Mount-Optionen
- OS-User auf allen Servern einheitlich (gleiche UID, GID)
- PostgreSQL-Versionen?
- Namenskonventionen DBC, Databases, User, Schema
- Security-Einstellungen (hostssl, CIS, Auditing)
- Logging
- Backup / Restore

Tools

- pgAdmin
- pg_profile
- pgBadger
- Backup / Commvault
- Replikation

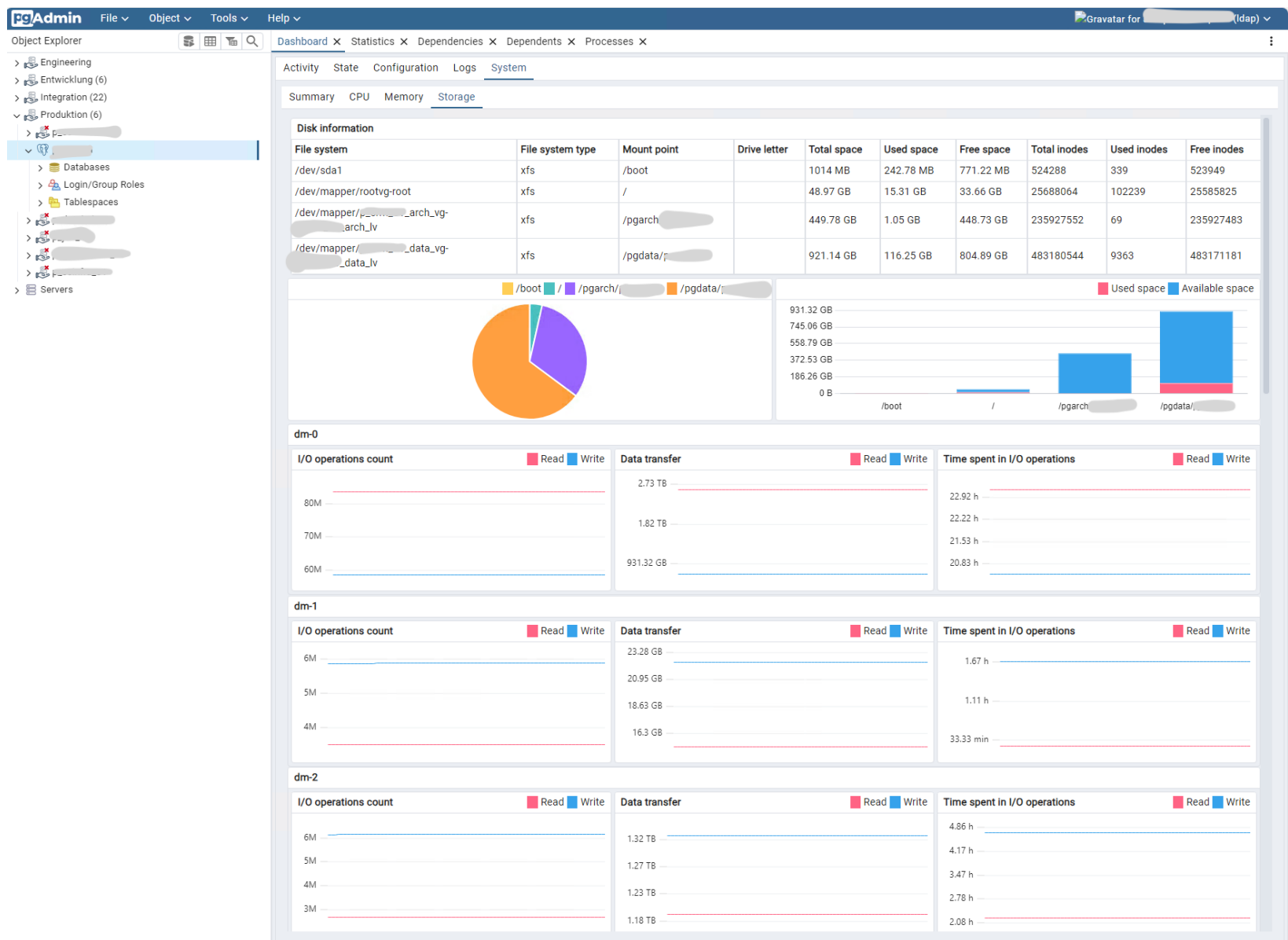
Tools

pgAdmin



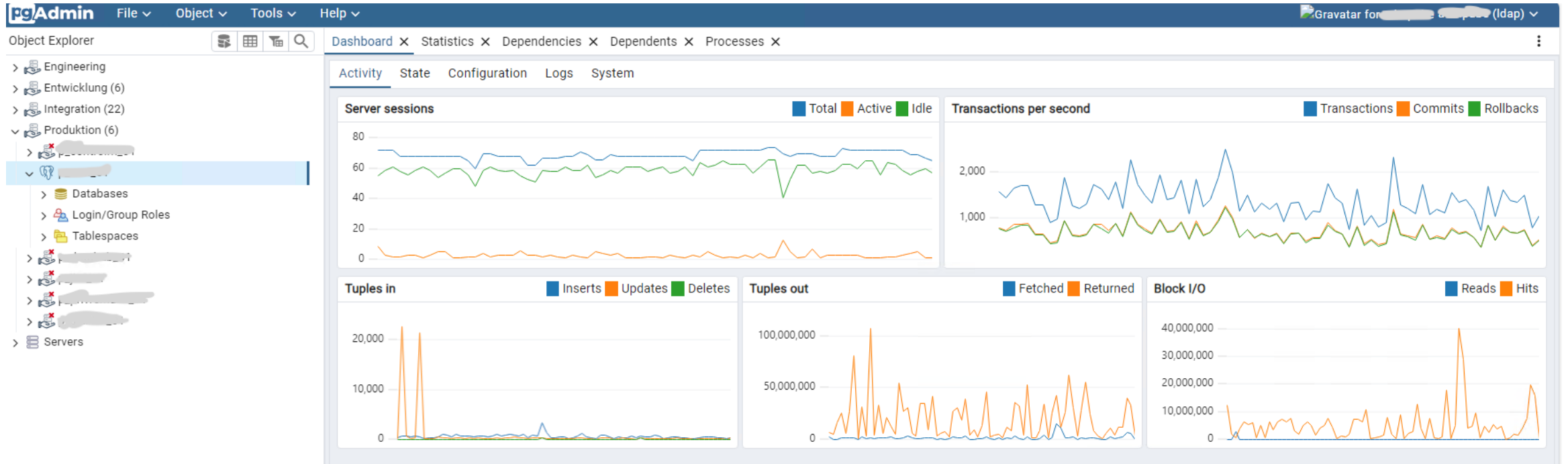
Tools

pgAdmin



Tools

pgAdmin



Tools

pg_profile

Filter...

Everywhere

Report details

Server statistics

SQL query statistics

- Top SQL by execution time
- Top SQL by executions
- Top SQL by I/O wait time
- Top SQL by shared blocks fetched
- Top SQL by shared blocks read
- Top SQL by shared blocks dirtied
- Top SQL by shared blocks written
- Top SQL by WAL size
- Top SQL by temp usage
- Complete list of SQL texts

Schema object statistics

- Top tables by estimated sequentially scanned volume
- Top tables by blocks fetched
- Top tables by blocks read
- Top DML tables
- Top tables by updated/deleted tuples
- Top growing tables
- Top indexes by blocks fetched
- Top indexes by blocks read
- Top growing indexes
- Unused indexes

User function statistics

- Top functions by total time
- Top functions by executions
- Top trigger functions by total time

Vacuum-related statistics

- Top tables by vacuum operations
- Top tables by analyze operations
- Top indexes by estimated vacuum load
- Top tables by dead tuples ratio
- Top tables by modified tuples ratio

Cluster settings during the report interval

Postgres profile report

Report details

Version	Server name	Interval (sample)		Interval (time)	
		start	end	start	end
4.6	local	4526	4549	2024-09-03 00:00:01+02	2024-09-03 23:00:02+02

Setting

Value

version	PostgreSQL 14.12 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-44), 64-bit
---------	--

Server statistics

Database statistics

Database	Transactions			Block statistics			Block I/O times		Tuples					Temp files		Size	Growth
	Commits	Rollbacks	Deadlocks	Hit(%)	Read	Hit	Read	Write	Ret	Fet	Ins	Upd	Del	Size	Files		
db_01	4319122	4270005		99.47	268858981	50708672375	3528.28	1.92	97148516858	19317708511	4684956	4995130	2505671	155 GB	10374	104 GB	499 MB
db_01_mq	145476			99.18	106241	12854734	3.16		17520120	5356095	10789	17505	9157			122 MB	1568 kB
postgres	36573			99.89	54141	49257629	18.89	0.01	57525390	29639331	221708	186029	220739	43 MB	7	130 MB	-4928 kB
Total	4501171	4270005		99.47	269019363	50770784738	3550.32	1.94	97223562368	19352703937	4917453	5198664	2735567	155 GB	10381	105 GB	496 MB

Cluster SLRU statistics

Name	Zeroed	Hits	Reads	%Hit	Writes	Checked	Flushes	Truncates
MultiXactMember		422		100				
MultiXactOffset		422		100				
Subtrans	768	46		100				
Xact	48	5471055	6491	99.88	2			
Total	816	5471945	6491	99.88	2			

Session statistics by database

Database	Timings			Sessions			
	Total	Active	Idle	Established	Abandoned	Fatal	Killed
db_01	3487324.41	51520.5	51927.12	10059	45		
db_01_mq	323196.29	103.11	3.53	2728	3		
postgres	3007.13	524.28	46.02	16774	1		
Total	3813527.83	52147.89	51976.68	29561	49		



Tools

pg_profile

Filter... Everywhere ▾												
- Report details - Server statistics - SQL query statistics - Schema object statistics - Top tables by estimated sequentially scanned volume - Top tables by blocks fetched - Top tables by blocks read - Top DML tables - Top tables by updated/deleted tuples - Top growing tables - Top indexes by blocks fetched - Top indexes by blocks read - Top growing indexes - Unused indexes - User function statistics - Top functions by total time - Top functions by executions - Top trigger functions by total time - Vacuum-related statistics - Top tables by vacuum operations - Top tables by analyze operations - Top indexes by estimated vacuum load - Top tables by dead tuples ratio - Top tables by modified tuples ratio - Cluster settings during the report interval												
SQL query statistics												
Top SQL by execution time												
Query ID	Database	User	Exec (s)	%Total	I/O time (s)		Rows	Execution times (ms)				Executions
					Read	Write		Mean	Min	Max	StdErr	
5a9b2e3c400e5fb4 [e40b82654f]	db_01	db_01_app	7076.39	14.56	12.59		1488	5127.82	1872.64	53226.24	4954.2	1380
31e1689d18bac1d4 [a489b6cda4]	db_01	db_01_app	6520.73	13.42	20.55		8	4728.6	1757.09	9764.92	1482.72	1379
344fc832a039455b [b58d4b8907]	db_01	db_01_app	5696.31	11.72	1.84		281147	2558.99	1476.53	9018.75	1338.85	2226
f0a88a5f8e4ad536 [bac045b903]	db_01	db_01_app	3776.54	7.77	1096.91	0.03	2013	20636.84	16099.15	46235.35	6086.59	183
9e56868f34326ae2 [760c09eaa5]	db_01	db_01_app	3562.54	7.33	258.63		179974	2240.59	98.16	6175.28	913.31	1590
a9a95b919e6a8221 [34ab45e534]	db_01	db_01_app	2502.79	5.15	0.51		1171	2115.63	1914.93	3026.94	119.44	1183
9cd42e54962c18ba [0fe75cbd14]	db_01	db_01_app	2471.54	5.09			1172	2108.82	1912.42	3724.19	146.88	1172
a94f467c79f85001 [4b9a4debb3]	db_01	db_01_app	1992.15	4.1	1.84		4292	4194	3810.7	7068.4	249.5	475
8df41952dde5012 [efca88b1ef]	db_01	db_01_app	769.92	1.58	12.45		7802	98.68	0.02	8604.86	329.03	7802
5b881e469a181dd9 [9d9715cffd]	db_01	db_01_app	485.56	1	462.34		50195	430.08	0.03	6818.53	831.1	1129
37a395c86a474a41 [472acc656d]	db_01	db_01_app	450.19	0.93	1.28		93015	4.84	0.09	12432.17	166.92	93015
da7fbfb4a2bcd732 [f0a18323e9]	db_01	db_01_app	432.82	0.89	0.07		2747	8015.18	1700.42	38077.03	11438.03	54

Tools

pg_profile

Filter... Everywhere Report details Server statistics SQL query statistics Schema object statistics - Top tables by estimated sequentially scanned volume - Top tables by blocks fetched - Top tables by blocks read - Top DML tables - Top tables by updated/deleted tuples - Top growing tables - Top indexes by blocks fetched - Top indexes by blocks read - Top growing indexes - Unused indexes User function statistics - Top functions by total time - Top functions by executions - Top trigger functions by total time Vacuum-related statistics - Top tables by vacuum operations - Top tables by analyze operations - Top indexes by estimated vacuum load - Top tables by dead tuples ratio - Top tables by modified tuples ratio Cluster settings during the report interval	5a9b2e3c400e5fb4	<pre>SELECT table_0_.CASE_NR AS col_0_ FROM app_CASE table_0_ WHERE table_0_.PRIVACY_STATUS_UID=\$7 AND EXISTS (SELECT \$8 AS col_1_ FROM app_CASE_INPUT table_1_ JOIN app_CUSTOMER table_2_ ON table_1_.ITEM_KEY0_NR=table_2_.CUSTOMER_NR WHERE table_0_.CASE_NR=table_1_.CASE_NR AND \$9=table_1_.ITEM_TYPE_ID AND table_1_.PROCESS_INPUT_UID = \$1 AND table_2_.PRIVACY_STATUS_UID=\$10 AND EXISTS (SELECT \$11 AS col_9_ FROM app_CUSTOMER_ADVISOR table_3_ JOIN LATERAL (SELECT table_4_.USER_NR AS col_2_, table_4_.USER_TYPE_UID AS col_3_, cast(\$12 as int8) AS col_4_, cast(\$13 as int8) AS col_5_, table_4_.OWNER_ORGANIZATION_UNIT_UID AS col_6_, table_4_.IS_ACTIVE AS col_7_, \$14 table_4_.USERNAME AS col_8_ FROM app_USER table_4_ WHERE table_4_.USER_TYPE_UID=\$15 AND table_4_.USER_NR=table_3_.ADVISOR_USER_NR AND \$16=table_3_.ADVISOR_TEAM_UID AND \$17 IN (table_3_.ADVISOR_TEAM_ROLE_UID, cast(\$18 as int8)) UNION ALL SELECT cast(\$19 as int8) AS col_2_, cast(\$20 as int8) AS col_3_, table_5_.TEAM_UID AS col_4_, table_7_.UC_UID AS col_5_, table_5_.OWNER_ORGANIZATION_UNIT_UID AS col_6_, CASE (table_7_.UC_UID) WHEN (\$21) THEN (cast(\$22 as boolean)) ELSE (table_6_.IS_DEFAULT_ACTI_ON_PARTITION) END AS col_7_, \$23 table_6_.GLOBAL_KEY \$24 table_7_.GLOBAL_KEY AS col_8_ FROM app_UC_TEAM table_5_ JOIN app_UC table_6_ ON table_6_.UC_UID=table_5_.TEAM_UID JOIN app_UC table_7_ ON table_7_.CODE_TYPE_NR=\$25 WHERE \$26=table_3_.ADVISOR_USER_NR AND table_6_.UC_UID=table_3_.ADVISOR_TEAM_UID AND table_7_.UC_UID IN (table_3_.ADVISOR_TEAM_ROLE_UID, cast(\$27 as int8)) UNION ALL SELECT cast(\$28 as int8) AS col_2_, cast(\$29 as int8) AS col_3_, table_8_.TEAM_UID AS col_4_, \$30 AS col_5_, table_8_.OWNER_ORGANIZATION_UNIT_UID AS col_6_, table_9_.IS_DEFAULT_ACTI_ON_PARTITION AS col_7_, \$31 table_9_.GLOBAL_KEY AS col_8_ FROM app_UC_TEAM table_8_ JOIN app_UC table_9_ ON table_9_.UC_UID=table_8_.TEAM_UID WHERE \$32=table_3_.ADVISOR_USER_NR AND table_9_.UC_UID=table_3_.ADVISOR_TEAM_UID AND \$33 IN (table_3_.ADVISOR_TEAM_ROLE_UID, cast(\$34 as int8))) table_10_ ON table_10_.col_2_=table_3_.ADVISOR_USER_NR AND table_10_.col_4_=table_3_.ADVISOR_TEAM_UID AND table_10_.col_5_ IN (table_3_.ADVISOR_TEAM_ROLE_UID, cast(\$35 as int8)) WHERE table_3_.ADVISOR_UID=\$2 AND EXISTS (SELECT \$36 AS col_10_ FROM app_UC table_11_ JOIN app_UC table_12_ ON table_11_.UC_UID=table_12_.UC_UID WHERE table_12_.UC_UID = any(\$3) AND table_11_.CODE_TYPE_NR=\$37 AND table_10_.col_4_=table_11_.UC_UID) AND table_2_.CUSTOMER_NR=table_3_.CUSTOMER_NR) AND NOT (table_2_.H_SYRIUS_PARTNER_PROT_UID = any(\$4))) AND NOT EXISTS (SELECT \$38 AS col_11_ FROM app_RULE_PENDING_PROCESSING table_13_ JOIN app_RULE_RUN table_14_ ON table_13_.RULE_RUN_NR=table_14_.RULE_RUN_NR WHERE table_13_.ITEM_TYPE_ID=\$39 AND table_13_.ITEM_KEY0_NR=table_0_.CASE_NR AND table_14_.RULE_UID=\$5) AND EXISTS (SELECT \$40 AS col_12_ FROM app_CASE_STEP table_15_ WHERE table_15_.RESPONSIBLE_TEAM_UID=\$6 AND table_0_.LAST_CASE_STEP_NR=table_15_.CASE_STEP_NR) AND table_0_.IS_PRIVATE_CASE = cast(\$41 as boolean)</pre>
	5b881e469a181dd9	<pre>SELECT COALESCE((table_1_.IS_READ), (cast(\$12 as boolean))) AS col_0_, table_1_.EVT_READ AS col_1_, table_0_.NEWSFEED_NR AS col_2_, table_0_.ITEM_TYPE_ID AS col_3_, table_0_.ITEM_KEY0_NR AS col_3_1, table_0_.ITEM_TYPE_ID AS col_4_, table_0_.ITEM_KEY0_NR AS col_4_1, table_0_.DESCRIPTION_SUBJECT AS col_5_, table_0_.DESCRIPTION_INFO AS col_6_, table_0_.DESCRIPTION_UID AS col_7_, table_0_.CHANGES_JSON AS col_8_, table_0_.CHANGE_ITEM_TYPE_ID AS col_9_, table_0_.CHANGE_ITEM_KEY0_NR AS col_9_1, table_0_.CHANGE_UID AS col_10_, table_0_.EVT_CHANGE AS col_11_, table_0_.CHANGED_BY_ITEM_TYPE_ID AS col_12_, table_0_.CHANGED_BY_ITEM_KEY0_NR AS col_12_1, table_0_.IS_DRILLDOWN AS col_13_ FROM app_NEWSFEED table_0_ LEFT OUTER JOIN app_NEWSFEED_USER table_1_ ON table_1_.NEWSFEED_NR=table_0_.NEWSFEED_NR AND table_1_.USER_NR=\$1 WHERE NOT EXISTS (SELECT \$13 AS col_14_ FROM app_GLOBAL_KEY_STATE table_2_ WHERE table_2_.ITEM_TYPE_ID=table_0_.ITEM_TYPE_ID AND table_2_.ITEM_KEY0_NR=table_0_.ITEM_KEY0_NR AND table_2_.PRIVACY_STATUS_UID<=\$14) AND table_0_.CHANGED_BY_ITEM_TYPE_ID in (\$2,\$3) AND NOT (table_0_.CHANGED_BY_ITEM_KEY0_NR IN (SELECT table_3_.USER_NR AS col_15_ FROM app_USER table_3_ WHERE table_3_.USERNAME=\$4)) AND EXISTS (SELECT table_4_.NEWSFEED_NR AS col_16_ FROM app_NEWSFEED_RESPONSIBLE table_4_ WHERE table_4_.CUSTOMER_NR IN (SELECT table_5_.USER_NR AS col_17_ FROM app_USER SUBSTITUTE_FLAT table_5_ WHERE table_5_.SUBSTITUTE_USER_NR=\$5) AND table_4_.NEWSFEED_NR=table_0_.NEWSFEED_NR) AND NOT (table_0_.CHANGED_BY_ITEM_TYPE_ID=\$6 AND table_0_.CHANGED_BY_ITEM_KEY0_NR=\$7) AND NOT (EXISTS (SELECT \$15 AS col_18_ FROM app_NEWSFEED_USER table_6_ WHERE table_0_.NEWSFEED_NR=table_6_.NEWSFEED_NR AND table_6_.USER_NR=\$8 AND table_6_.IS_READ=\$9)) AND table_0_.EVT_CHANGE>=\$10 limit \$11</pre>

Tools

pgBadger

Global Index - pgBadger

Year 2024

August

	Su	Mo	Tu	We	Th	Fr	Sa
31					01	02	03
32	04	05	06	07	08	09	10
33	11	12	13	14	15	16	17
34	18	19	20	21	22	23	24
35	25	26	27	28	29	30	31

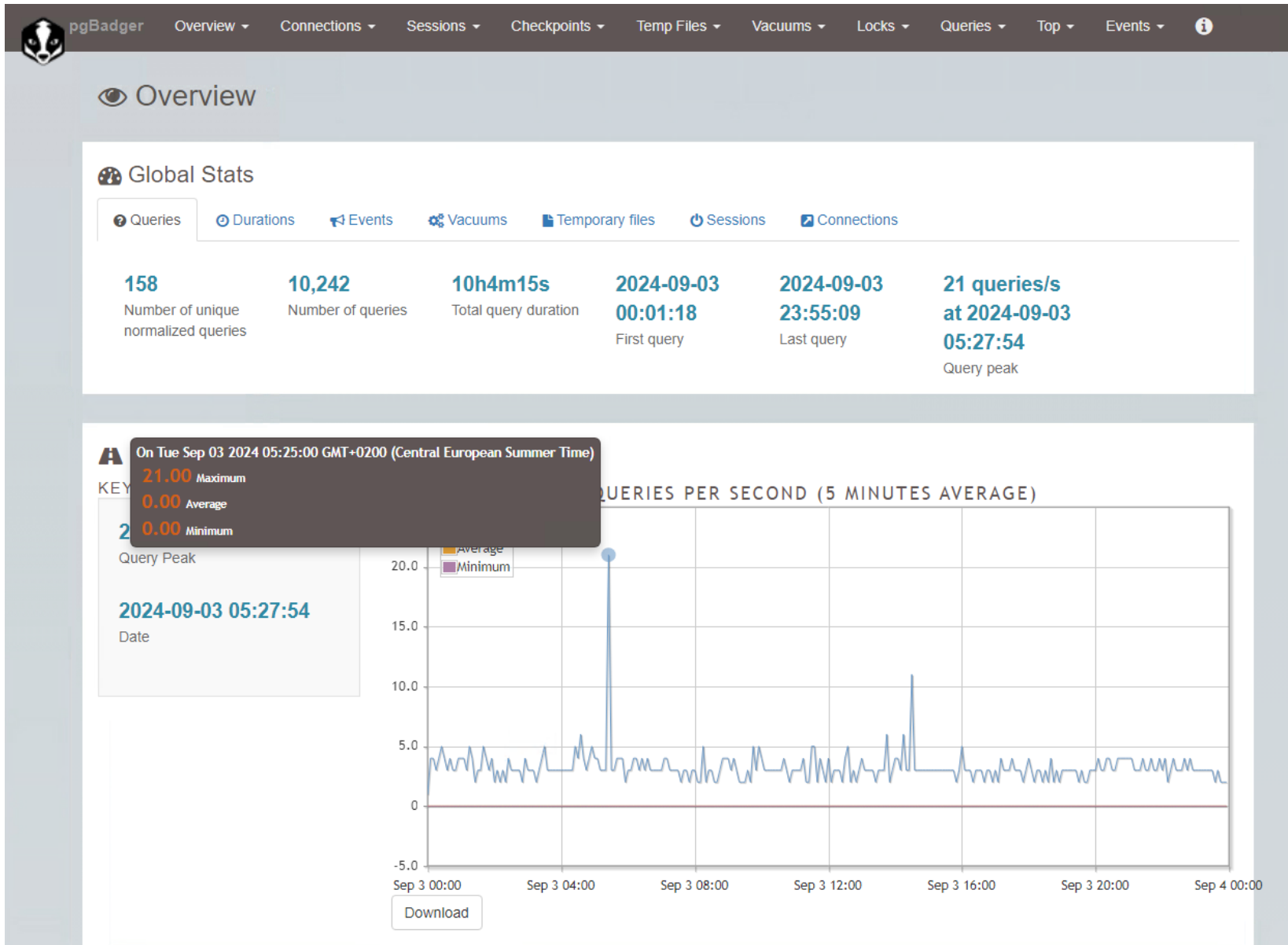
September

	Su	Mo	Tu	We	Th	Fr	Sa
36	01	02	03	04	05	06	07
37	08	09	10	11	12	13	14
38	15	16	17	18	19	20	21
39	22	23	24	25	26	27	28
40	29	30					

Report generated by [pgBadger 12.4](#).

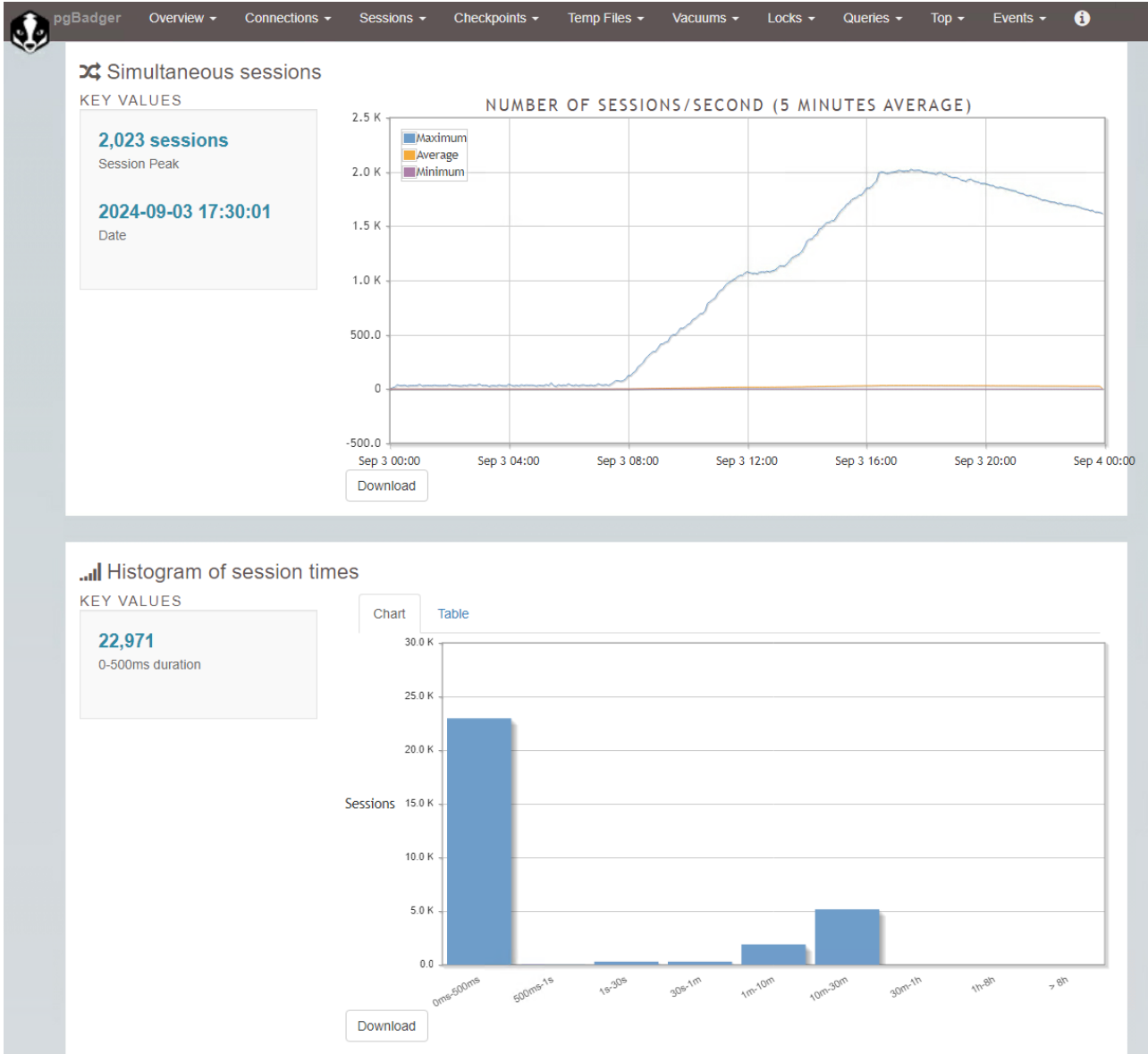
Tools

pgBadger



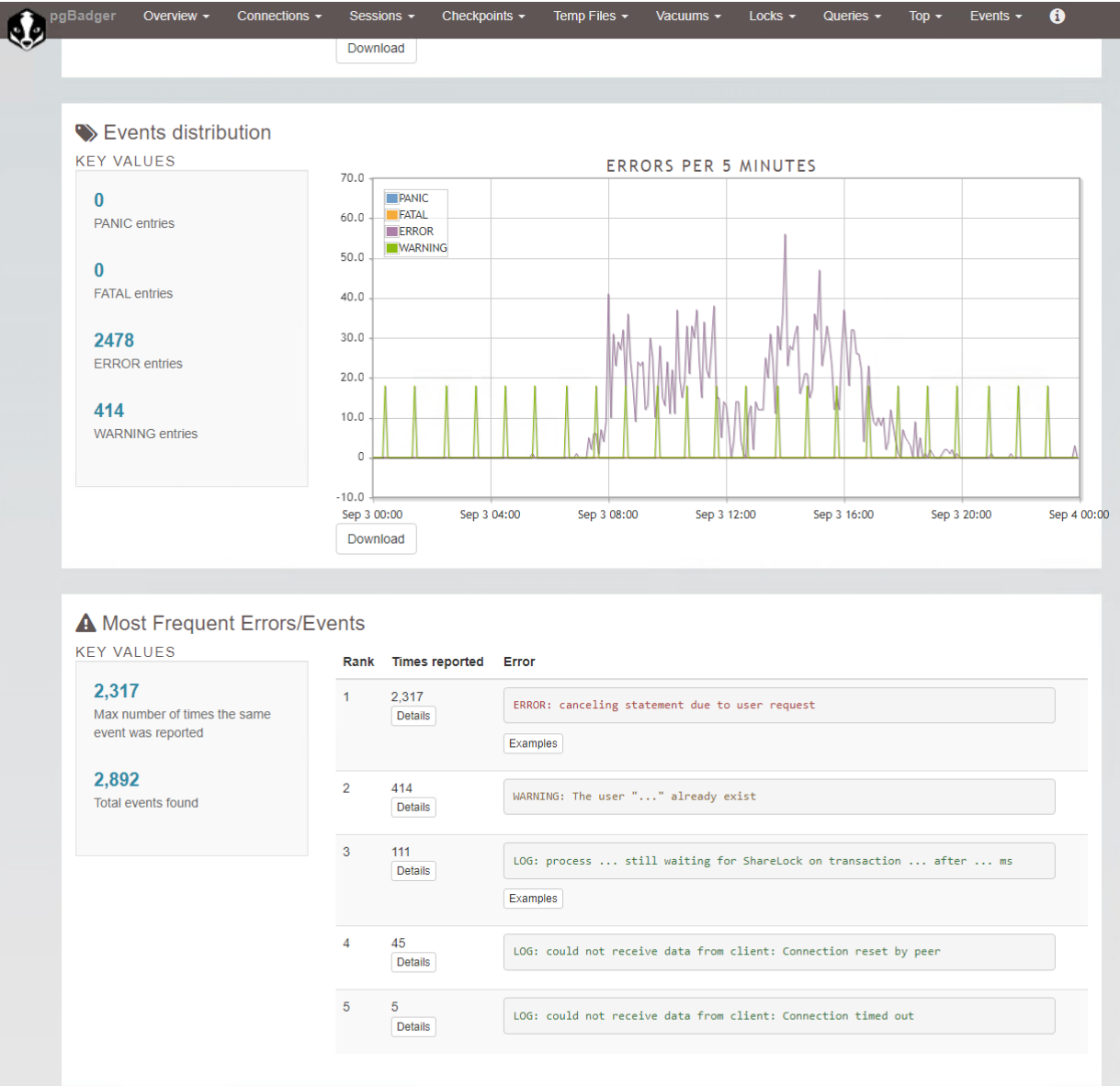
Tools

pgBadger



Tools

pgBadger



Offene Punkte

- Connection Pooling wie in Oracle
- Rman-like Tool / zentrales Backup-Repository
- pgAdmin automatisch neue Server / DBs hinzufügen
- Replikation (aktuell nicht geplant, da VMware-Failover)
- Direct und Async IO, Storage Management
- In-Place Migrationen Major-Versionen



WIR MACHEN IHNEN DAS LEBEN EINFACHER

Ganzheitliche und lösungsorientierte Beratung.
Von der Analyse bis zur erfolgreichen Einführung.