ProActive DBA

PDBA DB Monitoring Solution for  EDB POSTGRES

Enterprise Level DB Monitoring Solution for PostgresSQL & EnterpriseDB Postgres

2023.10

㈜ 데이터웍스

주요내용

- PDBA 개요
- PDBA 기능소개
  - Dashboard
  - SQL Capture
  - TimeLine
  - Alert / Report
- PDW 활용사례
- Demo & Q/A

PDBA 개요(1)

✓ PDBA DB 모니터링 솔루션 (기본)

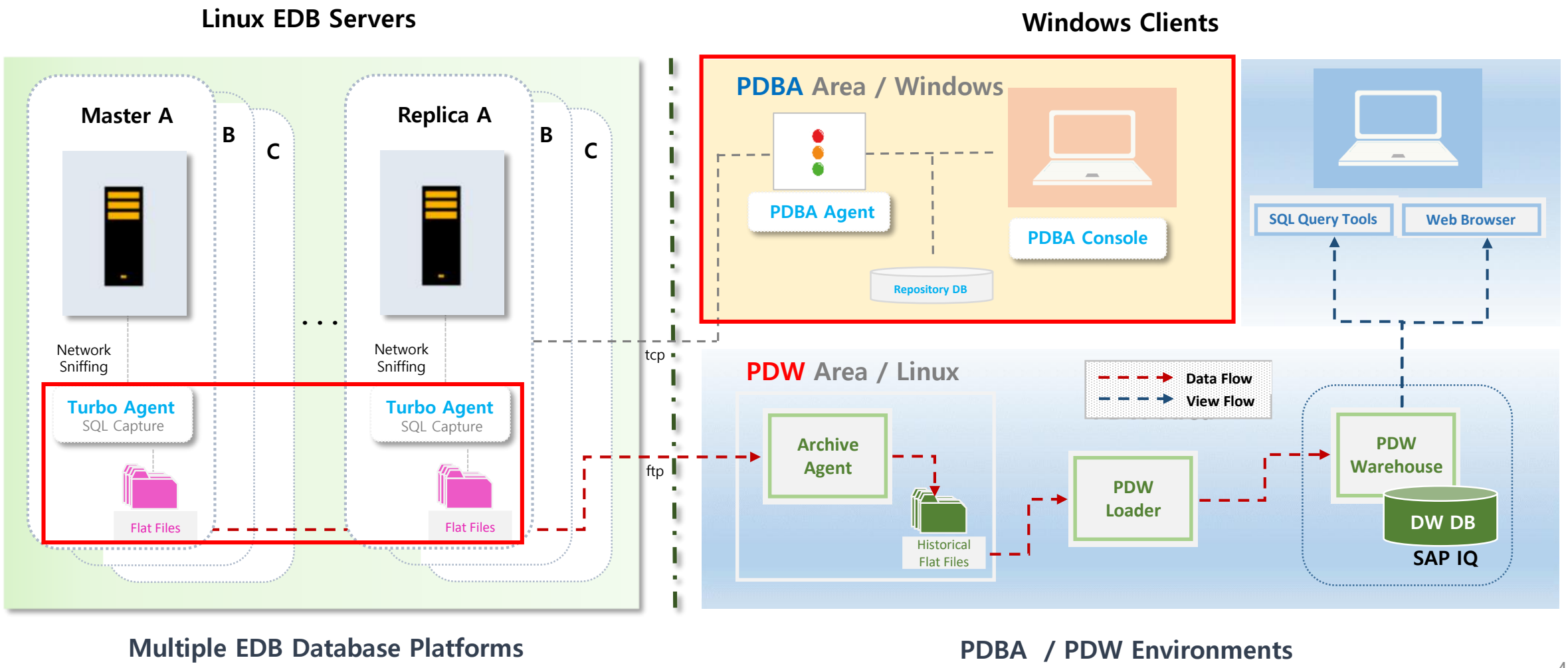
ProActive DBA(PDBA)은 **미국 White Sands Technology** 사에서 제작한 DB 모니터링 솔루션으로 PostgreSQL / EDB Postgres 를 비롯하여 SAP ASE, IQ, Rep. Server 그리고 MSSQL 과 Oracle에 대하여 수행되는 **SQL 실행정보**를 포함한 중요 **DB 운영정보**를 Network Sniffing에 의한 **최소한의 부하로 실시간 수집**하고 관리하여 **DB 운영관련 문제 발생시 즉시 대응**을 가능하게 지원합니다.

✓ PDW 시스템 구축 필요성 (확장)

ProActive DBA Warehouse(PDW)는 ProActive DBA의 확장옵션으로 **복수의 각 서버에서 수집되는 SQL**을 포함한 중요 DB 운영정보를 이관하고 **통합하여 장기적인 보관 및 검색**을 가능하게 함으로 **DB 운영 환경에 대한 효율화 증대 및 DB 접근이력에 대한 감시와 감사**를 가능하게 하는 **통합 목적용 DW 시스템**입니다.

PDBA 개요(2)

➢ PDBA 시스템 구성도



PDBA 기능별 상세

Dashboard/SQL Capture/TimeLine/Report

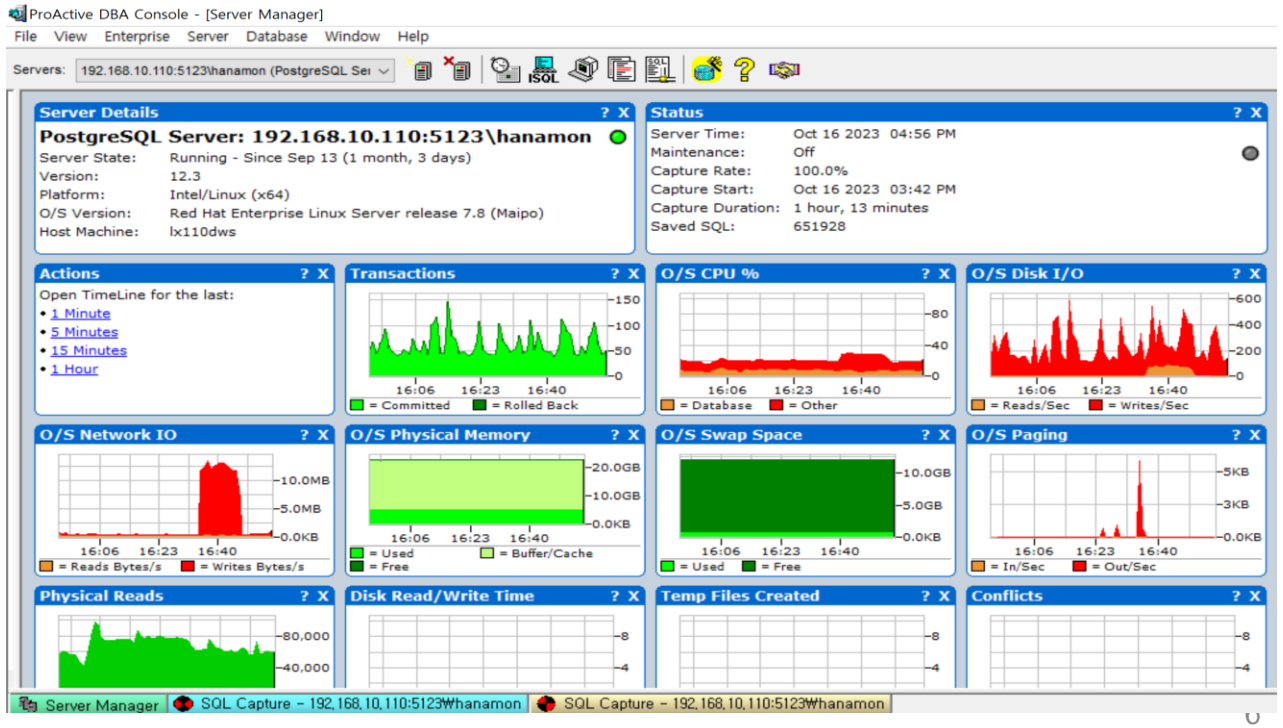
Dashboard

EDB Postgres 주요 KPI에 대한 실시간 대시보드 기능

CPU, Memory 및 Disk 등 인프라 자원의 운용현황 및 SQL 내용과 실제 수행되는 프로세스 등 실 운영상황을 나타내는 **Key Performance Indicators (KPI)에 대한 다양한 사용자 맞춤형 DIY 화면 구성**이 가능합니다.

PDBA Dashboard 기능

- EDB 서버의 주요 KPI 상황을 한눈에 모니터링하고, 각 KPI 간의 비교 검증이 가능하게 하는 기능
- DBA가 주요 KPI 지표에 추가 및 삭제가 가능하여 사용자별 DIY 맞춤형 구성이 가능함
- Host, Service, CPU, Memory Usage 현황 등, 주요 KPI에 대한 실시간 Dashboard 제공



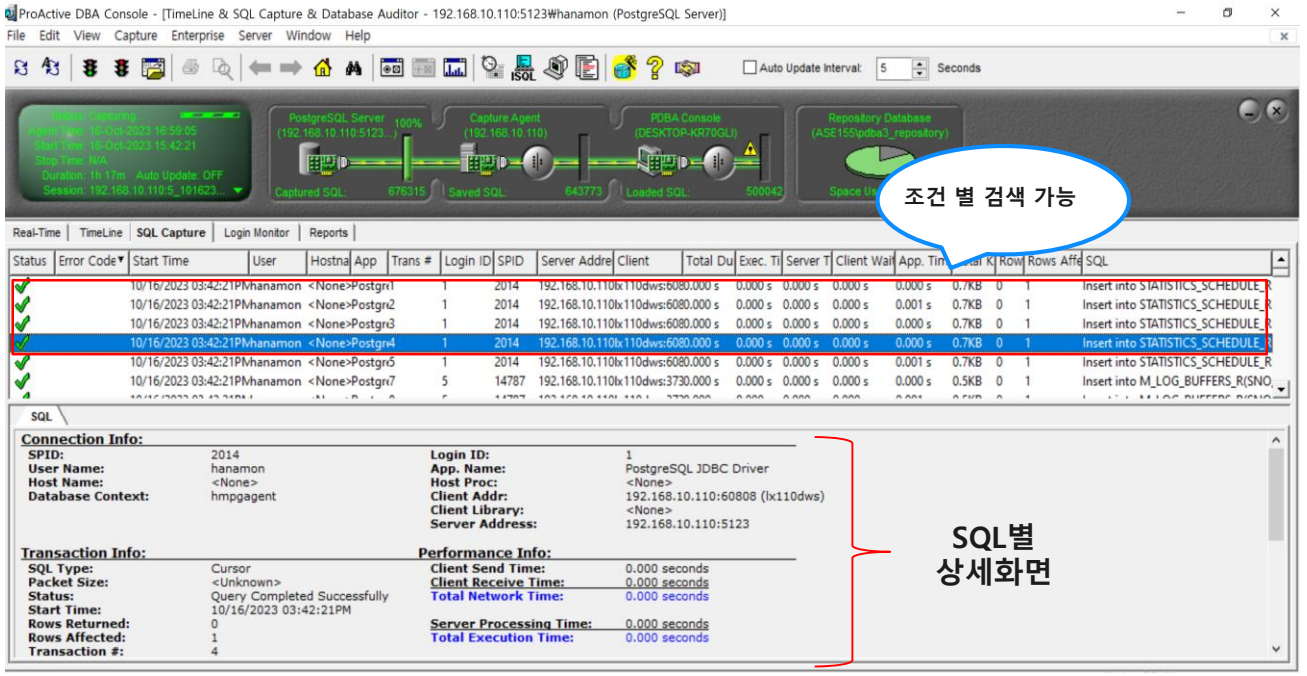
PDBA SQL Capture

수행되는 모든 SQL을 실시간 수집 기능

DB 서버에서 처리되는 모든 SQL 및 Login 정보를 성능에 성능 부하를 최소화하기 위하여 네트워크 상에서 스니핑 기능을 이용하여 실시간 수집하여 시계열 형태로 저장하는 기능입니다.

PDBA SQL Capture 기능

- PDBA 솔루션의 디폴트 기본 기능
- 수행되는 모든 SQL(입력 파라메터 포함) 실시간 수집하고 저장하여 성능 튜닝 및 감사 목적으로 사용
- 요청된 SQL Text, Procedure 정보 그리고 사용된 Client IP, 시작시간, 수행시간 등 모든 정보를 시계열 형태로 저장하고 검색을 가능하게 지원하는 기능
- Expensive SQL의 상세한 수행 이력에 대한 정보 제공
- SQL Capture Alerting Option - High Return Rows, Long Running 등, 다양한 Alerting 옵션이 제공됨



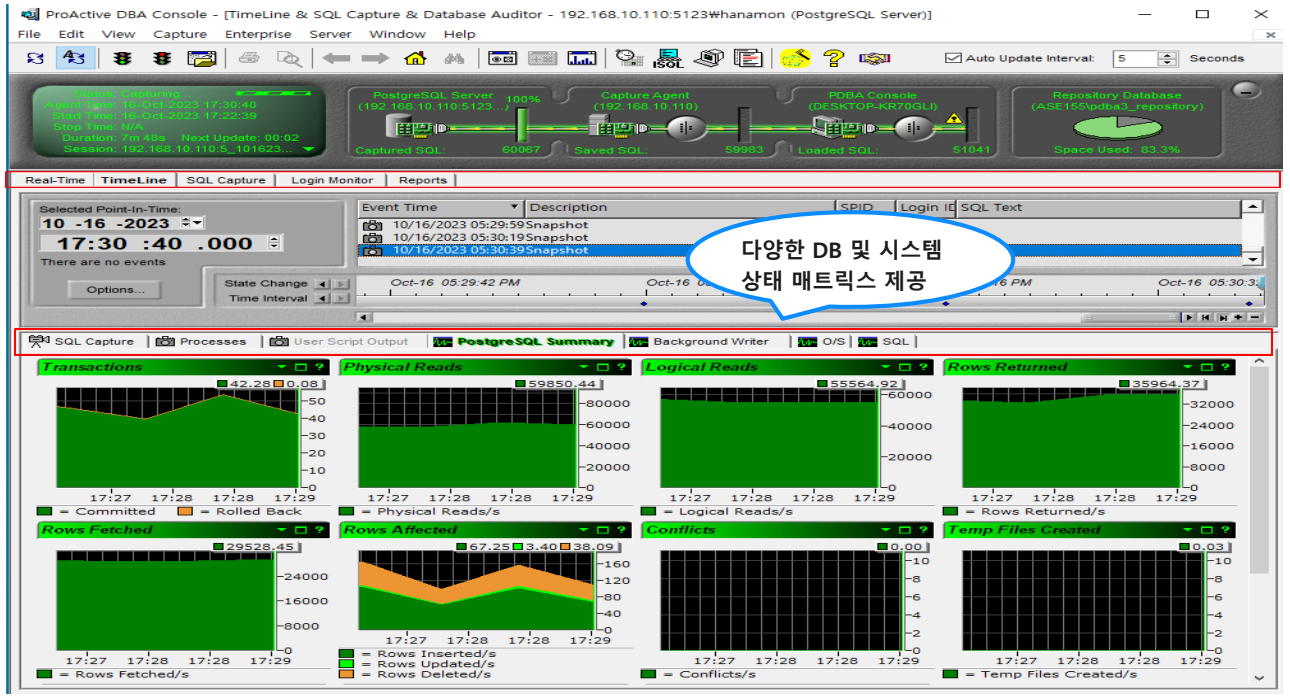
PDBA Timeline(1)

시점 별 장애/운영이력 분석용 타임라인 기능

장애 발생 시 실시간 또는 과거의 수행 이력에 대한 **심층분석**이 요구될 때 사용하는 기능으로 **특정된 시점 별 CPU, Cache, Disk, Network 등 인프라 자원의 사용현황** 및 **SQL**이 수행된 Connection, Process 등 트랜잭션 수행에 대한 연동 분석을 제공합니다.

PDBA Timeline 기능

- 성능부하 또는 장애 시점에 대한 Process Tracking, SQL Statement, Cache Usage 통합 분석 인프라 제공
- 과거의 운용 이력에 대한 시 / 분별로 re-play가 가능하여 시점 별 사용추이 비교 등, 심층 분석이 가능함
- Expensive SQL의 Process, CPU / Memory 등 자원 사용과 연계한 Tracking & Analytics 분석 지원
- TimeLine Alerting Option - OS, 서버 또는 프로세스에 임계 치를 설정하여 문제 또는 장애 상황을 사전에 인지하여 조치할 수 있게 하는 50여 개 이상의 다양한 Alerting 기능을 제공함



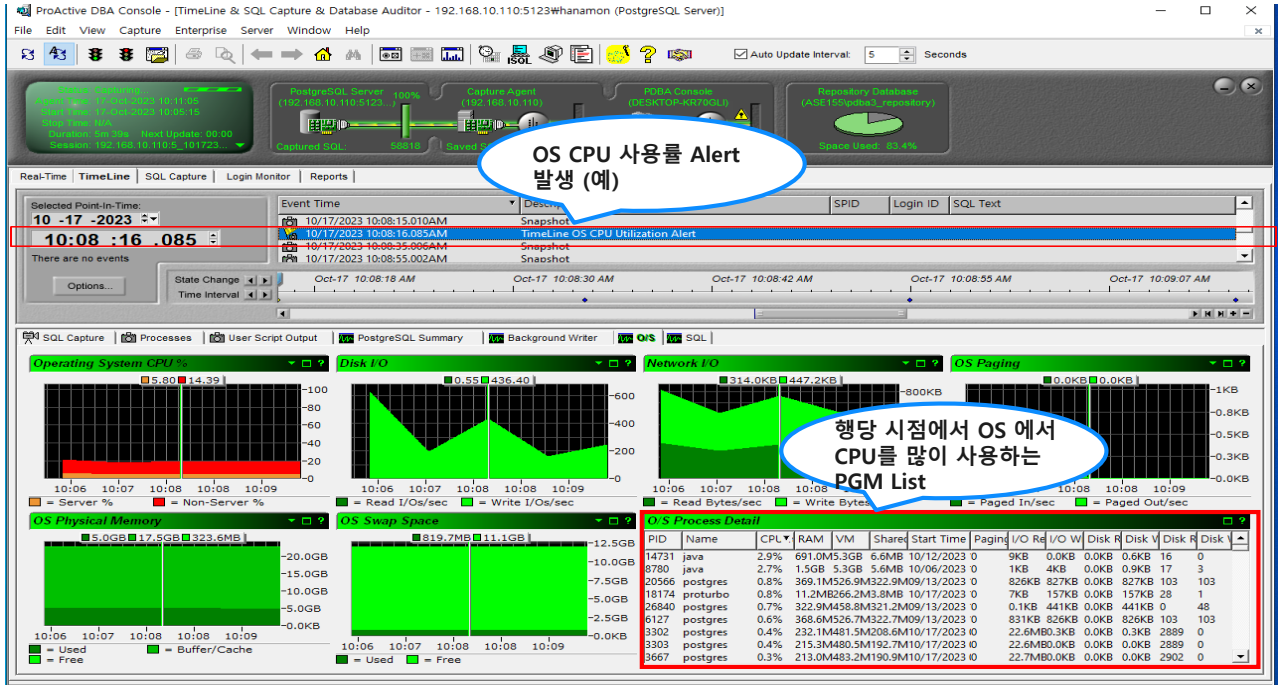
PDBA Timeline(2)

시점 별 장애/운영이력 분석용 타임라인 기능

장애 발생 시 실시간 또는 과거의 수행 이력에 대한 **심층분석**이 요구될 때 사용하는 기능으로 **특정된 시점 별 CPU, Cache, Disk, Network 등 인프라 자원의 사용현황** 및 **SQL**이 수행된 Connection, Process 등 트랜잭션 수행에 대한 연동 분석을 제공합니다.

PDBA Timeline 기능

- 성능부하 또는 장애 시점에 대한 Process Tracking, SQL Statement, Cache Usage 통합 분석 인프라 제공
- 과거의 운용 이력에 대한 시 / 분별로 re-play가 가능하여 시점 별 사용추이 비교 등, 심층 분석이 가능함
- Expensive SQL의 Process, CPU / Memory 등 자원 사용과 연계한 Tracking & Analytics 분석 지원
- TimeLine Alerting Option - OS, 서버 또는 프로세스에 임계 치를 설정하여 문제 또는 장애 상황을 사전에 인지하여 조치할 수 있게 하는 50여 개 이상의 다양한 Alerting 기능을 제공함



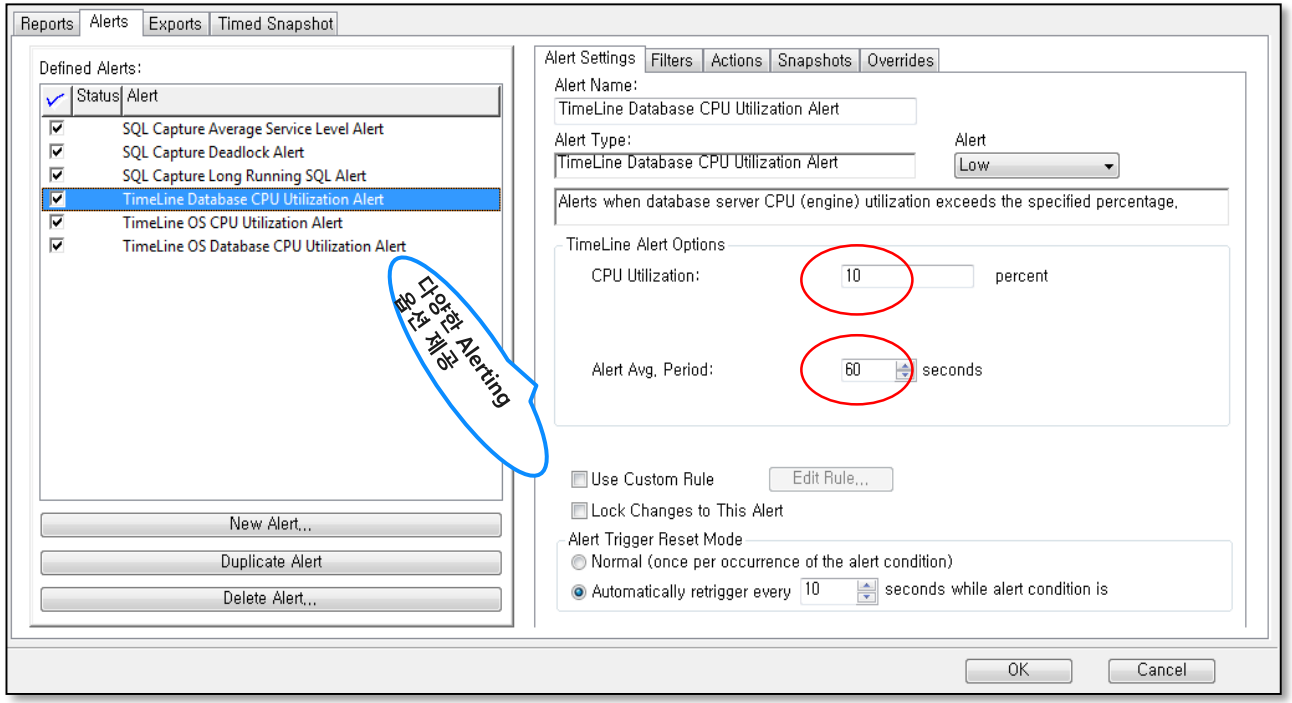
PDBA Alerting

DB서버의 운영상황을 사전 파악하는 Alert 기능

DB서버의 원활한 운영을 위하여 장애가 발생될 수 있는 다양한 운영 지표에 대하여 사전에 임계 치를 설정하여 DBA로 하여금 사전에 파악하고 조치할 수 있게 하는 시간적, 기능적 인프라를 제공하는 기능입니다.

PDBA Alerting 기능

- SQL 운영상황 및 Database, OS 등, 50여 항목 이상의 다양한 Alert 기능을 제공함
- Locking, SQL 수행시간 등 Processing 관련 기능 및 CPU, Disk 사용률 등 인프라 관련 그리고 Error No, Text 등 DB 서버와 관련된 다양한 Alert을 제공함
- 간단한 설정으로 사용이 가능하도록 스마트 스크린 기능을 제공함
- 설정에 의하여 Email, SMS 등 자동 통보 기능제공



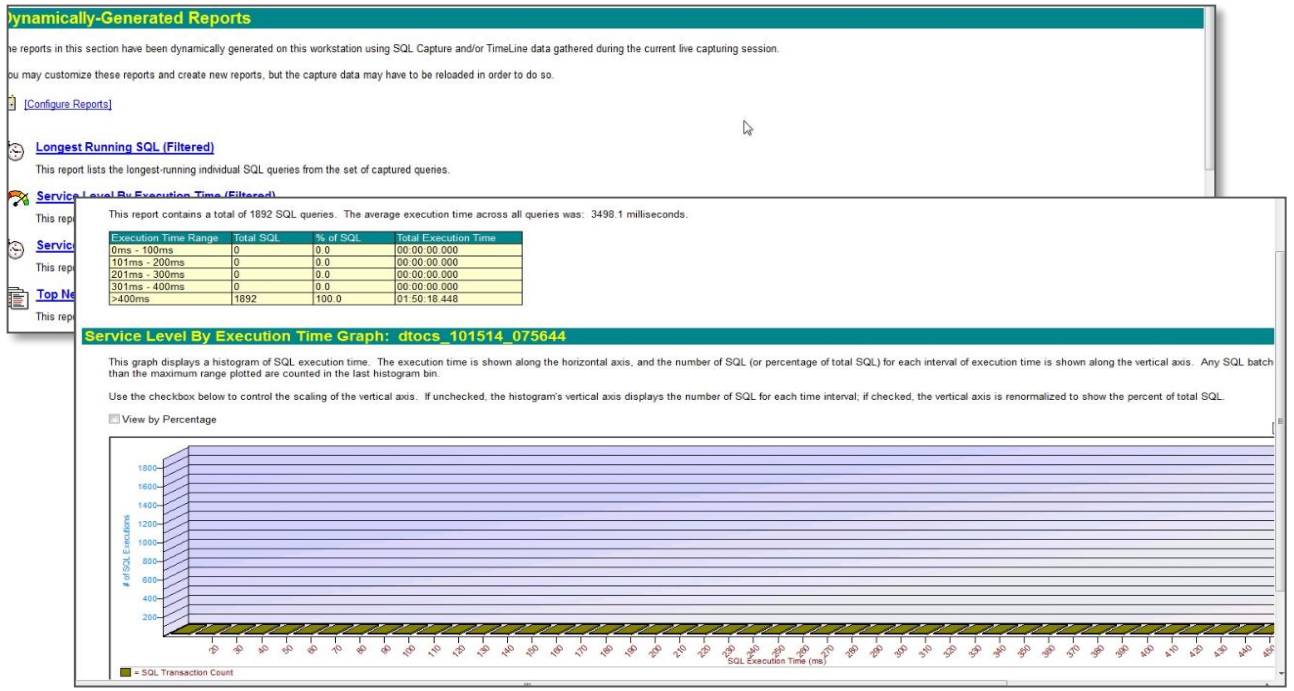
PDBA Reporting

운영상황에 대한 실시간 Report 기능

DB서버의 운영 현황에 대하여 각 기능별로 실시간 또는 시점 별로 다양한 운영 Report를 지원합니다.

PDBA Reporting 기능

- 다양한 TOP-N 형태의 다양한 운영 Report를 실시간 또는 과거 시점 별로 제공
- Long Running, SQL/SP 수행빈도 등 SQL에 대한 상세한 TOP-N Report 제공
- 가장 많이 수행된 Client / Host, AP 등 Report 제공
- CPU 부하, Network 부하 등에 대한 Report 제공
- 일 /시간당 SQL 수행 분포에 대한 Report 제공
- 수행된 Report에 대한 저장 및 검색 기능을 제공
- 총 20여 항목의 다양한 Report 기능을 제공함

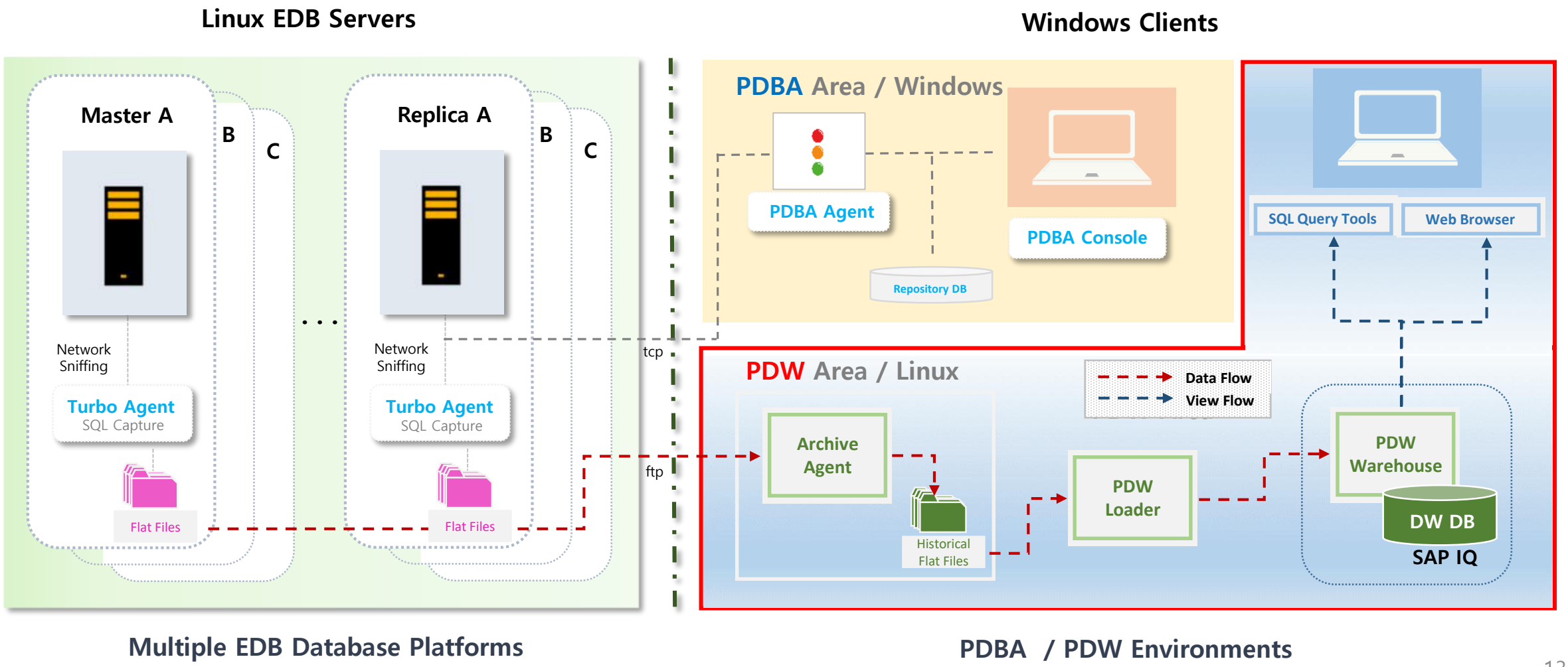


ProActive DBA Warehouse(PDW) 활용 사례

SQL Query Tool을 이용한 활용

PDW 개요

➢ PDW 시스템 구성도



PDW 활용

SQL Tool에 의한 활용 사례(1/4)

- DBA가 직접 PDW에 SQL Tool로 접속하여 원하는 데이터를 검색
- 일별 **가장 오래 수행된 SQL Text** 검색 – 하루 동안 10초 이상 수행된 SQL 모두 검색(예)

```
DECLARE @server_id int
        , @server_name VARCHAR(50)
        , @date VARCHAR(10)

SELECT @server_name = 'ASE160_205' --ASE160_205,   IQ160_207
       , @date = '20160906'

SELECT @server_id = server_id
FROM proact_pdw_servers
WHERE server_name = @server_name

SELECT a.server_id
       , a.source_file_id
       , a.sql_detail_id
       , a.login_id
       , a.start_time
       , a.end_time
       , a.server_processing_time
       , b.login_time
       , b.host_id
       , b.spid
INTO #test1
FROM proact_pdw_sql_detail a
       , proact_pdw_login_detail b
WHERE a.server_id = @server_id
AND a.server_processing_time > 10000
AND a.source_file_id = b.source_file_id
AND a.login_id = b.login_id


CREATE NONCLUSTERED INDEX IX_test1_server_id ON #test1(server_id, source_file_id, sql_detail_id)


select @server_name AS server_name
       , b.login_time
       , b.start_time
       , b.end_time
       , b.spid
       , convert(VARCHAR(255), a.raw_sql_text) AS long_query
       , b.server_processing_time
       , a.sql_text_id
       , a.chunk_no
       , a.b_continued
from proact_pdw_raw_sql_text a, #test1 b
WHERE b.server_id = a.server_id
AND b.source_file_id = a.source_file_id
AND b.sql_detail_id = a.sql_detail_id
```

server_name	login_time	start_time	end_time	spid	long_query	server_processing
ASE160_205	Sep 6 2016 3:10PM	Sep 6 2016 3:10PM	Sep 6 2016 3:10PM	44	insert into htest3_backselect top 10000 * from htest	10399
ASE160_205	Sep 6 2016 3:13PM	Sep 6 2016 3:13PM	Sep 6 2016 3:13PM	26	insert into htest3_backselect top 10000 * from htest	10403
ASE160_205	Sep 6 2016 3:15PM	Sep 6 2016 3:15PM	Sep 6 2016 3:15PM	15	insert into htest3_backselect top 10000 * from htest	12022
ASE160_205	Sep 6 2016 3:16PM	Sep 6 2016 3:16PM	Sep 6 2016 3:16PM	53	insert into htest3_backselect top 10000 * from htest	10285
ASE160_205	Sep 6 2016 3:17PM	Sep 6 2016 3:17PM	Sep 6 2016 3:17PM	57	insert into htest3_backselect top 10000 * from htest	10848
ASE160_205	Sep 6 2016 3:18PM	Sep 6 2016 3:18PM	Sep 6 2016 3:18PM	63	insert into htest3_backselect top 10000 * from htest	11389
ASE160_205	Sep 6 2016 3:19PM	Sep 6 2016 3:19PM	Sep 6 2016 3:19PM	67	insert into htest3_backselect top 10000 * from htest	11716
ASE160_205	Sep 6 2016 3:20PM	Sep 6 2016 3:20PM	Sep 6 2016 3:20PM	66	insert into htest3_backselect top 10000 * from htest	11294
ASE160_205	Sep 6 2016 3:21PM	Sep 6 2016 3:21PM	Sep 6 2016 3:21PM	19	insert into htest3_backselect top 10000 * from htest	10659
ASE160_205	Sep 6 2016 3:22PM	Sep 6 2016 3:22PM	Sep 6 2016 3:22PM	55	insert into htest3_backselect top 10000 * from htest	10896
ASE160_205	Sep 6 2016 3:23PM	Sep 6 2016 3:23PM	Sep 6 2016 3:23PM	20	insert into htest3_backselect top 10000 * from htest	12097
ASE160_205	Sep 6 2016 3:24PM	Sep 6 2016 3:24PM	Sep 6 2016 3:24PM	65	insert into htest3_backselect top 10000 * from htest	11347
ASE160_205	Sep 6 2016 3:25PM	Sep 6 2016 3:25PM	Sep 6 2016 3:25PM	34	insert into htest3_backselect top 10000 * from htest	12081
ASE160_205	Sep 6 2016 3:26PM	Sep 6 2016 3:26PM	Sep 6 2016 3:26PM	24	insert into htest3_backselect top 10000 * from htest	11199
ASE160_205	Sep 6 2016 3:27PM	Sep 6 2016 3:27PM	Sep 6 2016 3:27PM	33	insert into htest3_backselect top 10000 * from htest	11403
ASE160_205	Sep 6 2016 3:28PM	Sep 6 2016 3:28PM	Sep 6 2016 3:28PM	36	insert into htest3_backselect top 10000 * from htest	11952
ASE160_205	Sep 6 2016 3:28PM	Sep 6 2016 3:29PM	Sep 6 2016 3:29PM	68	insert bulk testdb..htest1_bcp with nodescribe	12301
ASE160_205	Sep 6 2016 3:28PM	Sep 6 2016 3:29PM	Sep 6 2016 3:29PM	43	insert bulk testdb..htest3_bcp with nodescribe	12079
ASE160_205	Sep 6 2016 3:29PM	Sep 6 2016 3:29PM	Sep 6 2016 3:29PM	49	insert into htest3_backselect top 10000 * from htest	12105
ASE160_205	Sep 6 2016 3:28PM	Sep 6 2016 3:29PM	Sep 6 2016 3:29PM	28	[Bulk Insert]	10163
ASE160_205	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	35	insert into htest1_backselect top 10000 * from htest	15675
ASE160_205	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	19	insert into htest2_backselect top 10000 * from htest	17251
ASE160_205	Sep 6 2016 3:28PM	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	28	insert bulk testdb..htest2_bcp with nodescribe	17741
ASE160_205	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	Sep 6 2016 3:30PM	22	insert into htest3_backselect top 10000 * from htest	17647
ASE160_205	Sep 6 2016 3:31PM	Sep 6 2016 3:31PM	Sep 6 2016 3:31PM	15	insert into htest3_backselect top 10000 * from htest	10306
ASE160_205	Sep 6 2016 3:31PM	Sep 6 2016 3:31PM	Sep 6 2016 3:31PM	65	insert into htest2_backselect top 10000 * from htest	10178
ASE160_205	Sep 6 2016 3:32PM	Sep 6 2016 3:32PM	Sep 6 2016 3:32PM	17	insert into htest3_backselect top 10000 * from htest	10448
ASE160_205	Sep 6 2016 3:33PM	Sep 6 2016 3:33PM	Sep 6 2016 3:33PM	67	insert into htest3_backselect top 10000 * from htest	10478
ASE160_205	Sep 6 2016 3:34PM	Sep 6 2016 3:34PM	Sep 6 2016 3:34PM	28	insert into htest3_backselect top 10000 * from htest	11221
ASE160_205	Sep 6 2016 3:35PM	Sep 6 2016 3:35PM	Sep 6 2016 3:35PM	25	insert into htest3_backselect top 10000 * from htest	10427
ASE160_205	Sep 6 2016 3:37PM	Sep 6 2016 3:37PM	Sep 6 2016 3:37PM	28	insert into htest3_backselect top 10000 * from htest	10191
ASE160_205	Sep 6 2016 3:38PM	Sep 6 2016 3:38PM	Sep 6 2016 3:38PM	12	insert into htest3_backselect top 10000 * from htest	12411
ASE160_205	Sep 6 2016 3:38PM	Sep 6 2016 3:38PM	Sep 6 2016 3:38PM	26	insert into htest2_backselect top 10000 * from htest	11803
ASE160_205	Sep 6 2016 3:39PM	Sep 6 2016 3:39PM	Sep 6 2016 3:39PM	57	insert into htest3_backselect top 10000 * from htest	11021
ASE160_205	Sep 6 2016 3:39PM	Sep 6 2016 3:39PM	Sep 6 2016 3:39PM	56	insert into htest2_backselect top 10000 * from htest	10507
ASE160_205	Sep 6 2016 3:40PM	Sep 6 2016 3:40PM	Sep 6 2016 3:40PM	63	insert into htest3_backselect top 10000 * from htest	10023
ASE160_205	Sep 6 2016 3:42PM	Sep 6 2016 3:42PM	Sep 6 2016 3:42PM	24	insert into htest3_backselect top 10000 * from htest	10846
ASE160_205	Sep 6 2016 3:43PM	Sep 6 2016 3:43PM	Sep 6 2016 3:43PM	67	insert into htest3_backselect top 10000 * from htest	10949
ASE160_205	Sep 6 2016 3:45PM	Sep 6 2016 3:45PM	Sep 6 2016 3:45PM	20	insert into htest3_backselect top 10000 * from htest	10694
ASE160_205	Sep 6 2016 3:46PM	Sep 6 2016 3:46PM	Sep 6 2016 3:46PM	41	insert into htest3_backselect top 10000 * from htest	10352
ASE160_205	Sep 6 2016 3:48PM	Sep 6 2016 3:48PM	Sep 6 2016 3:48PM	30	insert into htest3_backselect top 10000 * from htest	10919
ASE160_205	Sep 6 2016 3:49PM	Sep 6 2016 3:49PM	Sep 6 2016 3:49PM	43	insert into htest3_backselect top 10000 * from htest	11252
ASE160_205	Sep 6 2016 3:50PM	Sep 6 2016 3:50PM	Sep 6 2016 3:50PM	1	insert into htest3_backselect top 10000 * from htest	10625
ASE160_205	Sep 6 2016 3:52PM	Sep 6 2016 3:52PM	Sep 6 2016 3:52PM	68	insert into htest3_backselect top 10000 * from htest	10363
ASE160_205	Sep 6 2016 3:54PM	Sep 6 2016 3:54PM	Sep 6 2016 3:54PM	21	insert into htest3_backselect top 10000 * from htest	10085
ASE160_205	Sep 6 2016 3:55PM	Sep 6 2016 3:55PM	Sep 6 2016 3:55PM	37	insert into htest3_backselect top 10000 * from htest	11211
ASE160_205	Sep 6 2016 3:56PM	Sep 6 2016 3:56PM	Sep 6 2016 3:56PM	26	insert into htest2_backselect top 10000 * from htest	10655
ASE160_205	Sep 6 2016 3:56PM	Sep 6 2016 3:56PM	Sep 6 2016 3:56PM	41	insert into htest3_backselect top 10000 * from htest	13236
ASE160_205	Sep 6 2016 3:57PM	Sep 6 2016 3:57PM	Sep 6 2016 3:57PM	35	insert into htest3_backselect top 10000 * from htest	11537
ASE160_205	Sep 6 2016 3:58PM	Sep 6 2016 3:58PM	Sep 6 2016 3:58PM	35	insert into htest3_backselect top 10000 * from htest	10876
ASE160_205	Sep 6 2016 3:59PM	Sep 6 2016 3:59PM	Sep 6 2016 3:59PM	45	insert into htest3_backselect top 10000 * from htest	11607

Rows 41 of 860

PDW 활용

SQL Tool에 의한 활용 사례(2/4)

- 일별 **특정 사용자가 접근한 테이블 목록** 검색 – User 명이 "pdba" 인 사용자가 접근한 테이블 목록 검색(예)

```
Query | Result 1 | Result 2 | Result 3 | Result 4

    th.table_id,
    user_id


-- Build final report output
SELECT
    CONVERT(VARCHAR(40), db.database_name) database_name,
    CONVERT(VARCHAR(40), tbl.owner_name) owner_name,
    CONVERT(VARCHAR(128), tbl.table_name) table_name,
    CONVERT(VARCHAR(40), u.user_name) user_name,
    Sum(selects_sum) selects_sum,
    Sum(inserts_sum) inserts_sum,
    Sum(bulk_inserts_sum) bulk_inserts_sum,
    Sum(updates_sum) updates_sum,
    Sum(deletes_sum) deletes_sum,
    Sum(order_bys_sum) order_bys_sum,
    Sum(group_bys_sum) group_bys_sum,
    Sum(joins_sum) joins_sum,
    Sum(havings_sum) havings_sum

FROM
    #th th
    INNER JOIN proact_pdw_table tbl ON  th.server_id = tbl.server_id AND
        th.table_id = tbl.table_id

    INNER JOIN proact_pdw_databases db ON   tbl.server_id = db.server_id AND
        tbl.database_id = db.database_id

    INNER JOIN proact_pdw_user u ON     th.server_id = u.server_id AND
        th.user_id = u.user_id
where u.user_name="pdba"    --사용자 : pdba

GROUP BY
    CONVERT(VARCHAR(40), db.database_name),
    CONVERT(VARCHAR(40), tbl.owner_name),
    CONVERT(VARCHAR(128), tbl.table_name),
    CONVERT(VARCHAR(40), u.user_name)

ORDER BY
    CONVERT(VARCHAR(40), db.database_name),
    CONVERT(VARCHAR(40), tbl.owner_name),
    CONVERT(VARCHAR(128), tbl.table_name),
    CONVERT(VARCHAR(40), u.user_name)


go
DROP TABLE #sm
go
DROP TABLE #th
go
```

Query | Result 1 | Result 2 | Result 3 | Result 4

database_name	owner_nam	table_name	user_name	selects_sum	inserts_sum	bulk_inserts_sum	updates_sum	deletes_sum	order_l
master	dbo	sysdatabases	pdba	66	0	0	0	0	0
proact_pdw	dbo	proact_pdw_alert_detail	pdba	0	0	42	0	0	0
proact_pdw	dbo	proact_pdw_alert_name	pdba	22	0	4	0	0	0
proact_pdw	dbo	proact_pdw_alert_type	pdba	128	0	0	0	0	0
proact_pdw	dbo	proact_pdw_app	pdba	43	0	12	0	0	0
proact_pdw	dbo	proact_pdw_client_addr	pdba	43	0	10	0	0	0
proact_pdw	dbo	proact_pdw_client_appl_name	pdba	43	0	2	0	0	0
proact_pdw	dbo	proact_pdw_client_host_name	pdba	43	0	2	0	0	0
proact_pdw	dbo	proact_pdw_client_library	pdba	43	0	6	0	0	0
proact_pdw	dbo	proact_pdw_client_name	pdba	43	0	2	0	0	0
proact_pdw	dbo	proact_pdw_column	pdba	60	0	10	0	0	0
proact_pdw	dbo	proact_pdw_column_joins	pdba	20	0	8	0	0	1
proact_pdw	dbo	proact_pdw_columns_hit	pdba	39	0	10	0	0	1
proact_pdw	dbo	proact_pdw_database_users	pdba	20	0	2	0	0	0
proact_pdw	dbo	proact_pdw_databases	pdba	48	0	4	0	0	3
proact_pdw	dbo	proact_pdw_dbm_activity	pdba	42	0	9	0	0	1
proact_pdw	dbo	proact_pdw_dbm_activity_types	pdba	67	0	0	0	0	3
proact_pdw	dbo	proact_pdw_host	pdba	43	0	6	0	0	0
proact_pdw	dbo	proact_pdw_keys	pdba	3524	1762	0	1791	0	0
proact_pdw	dbo	proact_pdw_loader_options	pdba	132	0	0	0	0	0
proact_pdw	dbo	proact_pdw_lock_class	pdba	22	0	2	0	0	0
proact_pdw	dbo	proact_pdw_login_detail	pdba	0	0	41	0	0	0
proact_pdw	dbo	proact_pdw_login_summary	pdba	16	7	0	15	0	0
proact_pdw	dbo	proact_pdw_perf_entry_data	pdba	22	0	0	0	0	0
proact_pdw	dbo	proact_pdw_perf_entry_types	pdba	128	0	0	0	0	0
proact_pdw	dbo	proact_pdw_perf_instance_names	pdba	26	0	10	0	0	4
proact_pdw	dbo	proact_pdw_perf_metrics	pdba	52	8	54	32	0	0
proact_pdw	dbo	proact_pdw_process_cmd	pdba	22	0	12	0	0	0
proact_pdw	dbo	proact_pdw_process_misc_text	pdba	22	0	2	0	0	0
proact_pdw	dbo	proact_pdw_process_status	pdba	22	0	10	0	0	0
proact_pdw	dbo	proact_pdw_process_tran_name	pdba	22	0	10	0	0	0
proact_pdw	dbo	proact_pdw_raw_sql_text	pdba	0	0	66	0	0	0
proact_pdw	dbo	proact_pdw_role	pdba	21	0	2	0	0	0
proact_pdw	dbo	proact_pdw_role_user	pdba	21	0	2	0	0	0
proact_pdw	dbo	proact_pdw_servers	pdba	147	66	0	2	0	1
proact_pdw	dbo	proact_pdw_snapshot_locks	pdba	0	0	6	0	0	0
proact_pdw	dbo	proact_pdw_snapshot_processes	pdba	0	0	42	0	0	0
proact_pdw	dbo	proact_pdw_source	pdba	66	0	16	0	0	0
proact_pdw	dbo	proact_pdw_source_file	pdba	150	64	0	201	0	0
proact_pdw	dbo	proact_pdw_sql_detail	pdba	0	0	42	0	0	0
proact_pdw	dbo	proact_pdw_sql_summary	pdba	41	14	0	16	0	0
proact_pdw	dbo	proact_pdw_sql_text	pdba	62	0	36	0	0	0
proact_pdw	dbo	proact_pdw_sql_text_hash	pdba	49	18	0	0	0	3
proact_pdw	dbo	proact_pdw_table	pdba	47	0	6	9	0	3
proact_pdw	dbo	proact_pdw_table_defn_text	pdba	38	0	2	0	0	9
proact_pdw	dbo	proact_pdw_table_joins	pdba	20	0	10	0	0	1
proact_pdw	dbo	proact_pdw_tables_hit	pdba	42	0	10	0	0	1
proact_pdw	dbo	proact_pdw_time_window	pdba	70	0	16	0	0	0
proact_pdw	dbo	proact_pdw_user	pdba	90	0	4	0	0	6
tempdb	<Unknown>	<TempTable>	pdba	141	19	137	3	0	0

Rows 1 of 50

PDW 활용

SQL Tool에 의한 활용 사례(3/4)

- 일별 **CPU를 가장 많이 사용한 SQL** 검색 – 하루 동안 CPU time을 1초 이상 점유한 SQL 모두 검색(예)

Query | table_usage.sql | Result 1 | Result 2 | Result 3 | Result 4 | Result 5 | Result 6 | Result 7 | Result 8 | Result 9 | Result 10

```sql
raw_sql_text varchar(255)
)
go

DECLARE @server_id INT
SELECT @server_id = server_id FROM proact_pdw_servers WHERE server_name = 'LX04DWS_ASE157'

select server_id, snapshot_time, spid, user_id, cpu_usage_cumul, cpu_usage_new, mem_usage
into #sn
from proact_pdw_snapshot_processes
where cpu_usage_cumul > 10 and snapshot_time between '20161010 00:00:00' and '20161010 23:00:00'
go

declare cur1 cursor for select snapshot_time, spid, user_id, cpu_usage_cumul, cpu_usage_new, mem_usage
from #sn
go

declare @snapshot_time datetime

declare @spid int
declare @user_id int
declare @user_name varchar(255)
declare @cpu_usage_cumul int
declare @cpu_usage_new int
declare @mem_usgae float
declare @start_time datetime
declare @end_time datetime
declare @chunk_no int
declare @raw_sql_text varchar(255)


DECLARE @server_id INT
SELECT @server_id = server_id FROM proact_pdw_servers WHERE server_name = 'LX04DWS_ASE157'




open cur1

fetch cur1 into @snapshot_time, @spid, @user_id, @cpu_usage_cumul, @cpu_usage_new, @mem_usgae

while @@sqlstatus = 0
begin

insert into #result
select
@server_id,
@spid,
c.user_name,
a.start_time,
a.end_time,
```

Result 7 | Result 8 | Result 9 | Result 10 | Result 11 | Result 12 | Result 13 | Result 14 | Result 15 | Result 16 | Result 17

server_id	spid	ser_name		end_time	cpu_usage_cumul	cpu_usage_new	mem_usage	chunk_no	
16	1	sa	2016-10-10 15:27:33.	Oct 10 2016 3:41PM	65	0	23	1	SELECT b.user_name ,I
16	1	sa	2016-10-10 15:27:33.	Oct 10 2016 3:41PM	65	0	23	2	er_id = h.server_idAND b.us
16	29	pdba	2016-10-10 15:11:15.	Oct 10 2016 3:11PM	43	0	25	1	BEGIN TRAN SAVE_SUMM
16	29	pdba	2016-10-10 15:11:15.	Oct 10 2016 3:11PM	43	0	25	2	= 241,last_saved_summary_
16	26	pdba	2016-10-10 15:21:39.	Oct 10 2016 3:21PM	80	0	41	1	BEGIN TRAN SAVE_SUMM
16	26	pdba	2016-10-10 15:21:39.	Oct 10 2016 3:21PM	80	0	41	2	dest.metric_sumsq + src.met
16	26	pdba	2016-10-10 15:21:39.	Oct 10 2016 3:21PM	80	0	41	3	0 then dest.metric_max else
16	26	pdba	2016-10-10 15:21:39.	Oct 10 2016 3:21PM	80	0	41	4	btypeAND dest.instance_nu
16	26	pdba	2016-10-10 15:21:39.	Oct 10 2016 3:21PM	80	0	41	5	2,last_saved_summary_time
16	19	pdba	2016-10-10 15:30:29.	Oct 10 2016 3:30PM	50	0	13	1	BEGIN TRAN SAVE_SUMM
16	19	pdba	2016-10-10 15:30:29.	Oct 10 2016 3:30PM	50	0	13	2	dest.metric_sumsq + src.met
16	19	pdba	2016-10-10 15:30:29.	Oct 10 2016 3:30PM	50	0	13	3	0 then dest.metric_max else
16	19	pdba	2016-10-10 15:30:29.	Oct 10 2016 3:30PM	50	0	13	4	btypeAND dest.instance_nu
16	19	pdba	2016-10-10 15:30:29.	Oct 10 2016 3:30PM	50	0	13	5	2,last_saved_summary_time
16	19	pdba	2016-10-10 15:30:41.	Oct 10 2016 3:30PM	50	0	45	1	BEGIN TRAN SAVE_SUMM
16	19	pdba	2016-10-10 15:30:41.	Oct 10 2016 3:30PM	50	0	45	2	dest.metric_sumsq + src.met
16	19	pdba	2016-10-10 15:30:41.	Oct 10 2016 3:30PM	50	0	45	3	0 then dest.metric_max else
16	19	pdba	2016-10-10 15:30:41.	Oct 10 2016 3:30PM	50	0	45	4	btypeAND dest.instance_nu
16	19	pdba	2016-10-10 15:30:41.	Oct 10 2016 3:30PM	50	0	45	5	= 2,last_saved_summary_tim
16	12	pdba	2016-10-10 15:30:54.	Oct 10 2016 3:31PM	65	0	45	1	BEGIN TRAN SAVE_SUMM
16	12	pdba	2016-10-10 15:30:54.	Oct 10 2016 3:31PM	65	0	45	2	dest.metric_sumsq + src.met
16	12	pdba	2016-10-10 15:30:54.	Oct 10 2016 3:31PM	65	0	45	3	0 then dest.metric_max else
16	12	pdba	2016-10-10 15:30:54.	Oct 10 2016 3:31PM	65	0	45	4	btypeAND dest.instance_nu
16	12	pdba	2016-10-10 15:30:54.	Oct 10 2016 3:31PM	65	0	45	5	2,last_saved_summary_time
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	1	SELECT b.user_name ,I
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	2	rver_idAND b.user_name = '
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	1	SELECT b.user_name ,I
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	2	rver_idAND b.user_name = '
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	1	SELECT b.user_name ,I
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	2	rver_idAND b.user_name = '
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	1	SELECT b.user_name ,I
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	25	2	rver_idAND b.user_name = '
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	45	1	SELECT b.user_name ,I
16	23	sa	2016-10-10 15:33:41.	Oct 10 2016 3:36PM	11	0	45	2	rver_idAND b.user_name = '
16	19	pdba	2016-10-10 15:40:53.	Oct 10 2016 3:41PM	64	0	45	1	BEGIN TRAN SAVE_SUMM
16	19	pdba	2016-10-10 15:40:53.	Oct 10 2016 3:41PM	64	0	45	2	dest.metric_sumsq + src.met
16	19	pdba	2016-10-10 15:40:53.	Oct 10 2016 3:41PM	64	0	45	3	0 then dest.metric_max else
16	19	pdba	2016-10-10 15:40:53.	Oct 10 2016 3:41PM	64	0	45	4	btypeAND dest.instance_nu
16	19	pdba	2016-10-10 15:40:53.	Oct 10 2016 3:41PM	64	0	45	5	2,last_saved_summary_time
16	21	pdba	2016-10-10 15:50:53.	Oct 10 2016 3:51PM	61	0	45	1	BEGIN TRAN SAVE_SUMM
16	21	pdba	2016-10-10 15:50:53.	Oct 10 2016 3:51PM	61	0	45	2	dest.metric_sumsq + src.met
16	21	pdba	2016-10-10 15:50:53.	Oct 10 2016 3:51PM	61	0	45	3	0 then dest.metric_max else
16	21	pdba	2016-10-10 15:50:53.	Oct 10 2016 3:51PM	61	0	45	4	btypeAND dest.instance_nu
16	21	pdba	2016-10-10 15:50:53.	Oct 10 2016 3:51PM	61	0	45	5	2,last_saved_summary_time
16	13	sa	2016-10-10 15:54:18.	Oct 10 2016 3:54PM	117	0	45	1	SELECT b.user_name ,I
16	13	sa	2016-10-10 15:54:18.	Oct 10 2016 3:54PM	117	0	45	2	rver_id = h.server_idAND b.u
16	19	pdba	2016-10-10 16:01:06.	Oct 10 2016 4:01PM	68	0	45	1	BEGIN TRAN SAVE_SUMM
16	19	pdba	2016-10-10 16:01:06.	Oct 10 2016 4:01PM	68	0	45	2	dest.metric_sumsq + src.met
16	19	pdba	2016-10-10 16:01:06.	Oct 10 2016 4:01PM	68	0	45	3	0 then dest.metric_max else
16	19	pdba	2016-10-10 16:01:06.	Oct 10 2016 4:01PM	68	0	45	4	btypeAND dest.instance_nu
16	19	pdba	2016-10-10 16:01:06.	Oct 10 2016 4:01PM	68	0	45	5	2,last_saved_summary_time

Rows 1 of 154

PDW 활용

SQL Tool에 의한 활용 사례(4/4)

- **시간대별 프로세스 사용률** 검색 – 시간대별 평균 프로세스 사용률 검색(예)

```
Query | Result 1

46
47  DECLARE @server_id INT
48  SELECT @server_id = server_id FROM proact_pdw_servers WHERE server_name = 'ASE160_205'
49
50  SET FORCEPLAN ON
51
52  INSERT INTO #pm
53  SELECT
54      smr.server_id,
55      smr.time_window_time,
56      smr.instance_num,
57      smr.instance_name_id,
58      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 1 THEN smr.metric_sum ELSE NULL END) / SUM
59      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 2 THEN smr.metric_sum ELSE NULL END) / SUM
60      100.0 * ( ( Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 6 THEN smr.metric_sum ELSE NUL
61      100.0 * ( ( Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 4 THEN smr.metric_sum ELSE NUL
62      100.0 * ( ( Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 5 THEN smr.metric_sum ELSE NUL
63      100.0 * ( Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 7 THEN smr.metric_sum ELSE NULL
64      100.0 * ( Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 8 THEN smr.metric_sum ELSE NULL
65      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 9 THEN smr.metric_sum ELSE NULL END) / SUM
66      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 10 THEN smr.metric_sum ELSE NULL END) / SU
67      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 11 THEN smr.metric_sum ELSE NULL END) / SU
68      Sum(CASE WHEN smr.entry_type = 560 AND smr.entry_subtype = 12 THEN smr.metric_sum ELSE NULL END) / SU
69      100.0 * ( 0.010000 * (Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 3 THEN smr.metric_su
70      100.0 * ( 0.010000 * (Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 4 THEN smr.metric_su
71      100.0 * ( 0.010000 * (Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 9 THEN smr.metric_su
72      100.0 * ( 0.010000 * (Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 5 THEN smr.metric_su
73      Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 6 THEN smr.metric_sum ELSE NULL END) / SUM
74      Sum(CASE WHEN smr.entry_type = 570 AND smr.entry_subtype = 7 THEN smr.metric_sum ELSE NULL END) / SUM
75
76  FROM
77      proact_pdw_perf_metrics smr
78      INNER JOIN proact_pdw_time_window w ON  smr.server_id = w.server_id AND
79          smr.time_window_id = w.time_window_id
80
81
82  WHERE
83      smr.server_id = @server_id AND
84      (
85          smr.time_window_time >= '20161010 00:00:00.000' AND
86          smr.time_window_time <= '20161011 00:00:00.000'
87      ) AND
88      smr.entry_type IN (560, 570)
89
90  GROUP BY
91      smr.server_id,
92      smr.time_window_time,
93      smr.instance_num,
94      smr.instance_name_id
95
96
```

Query | Result 1

time_window_time	instance_name	tpool_num_threads_avg	tpool_tasks_per_sec_avg	tpool_busy_percent_avg	tpool_idle_percent_avg	tpool_sleep_pe
Oct 10 2016 12:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 12:30AM	syb_default_pool	4	4.9591666666666665	0.14861111111111111	99.851388888888891	0
Oct 10 2016 12:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 1:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 1:30AM	syb_default_pool	4	4.858888888888889	0.14791666666666667	99.852083333333326	0
Oct 10 2016 1:30AM	syb_system_pool	3	0	0.0055555555555555558	99.994444444444454	0
Oct 10 2016 2:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 2:30AM	syb_default_pool	4	4.8688888888888888	0.14722222222222223	99.852777777777774	0
Oct 10 2016 2:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 3:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 3:30AM	syb_default_pool	4	4.8594444444444447	0.14166666666666666	99.85833333333332	0
Oct 10 2016 3:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 4:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 4:30AM	syb_default_pool	4	4.9091666666666667	0.14861111111111111	99.851388888888891	0
Oct 10 2016 4:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 5:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 5:30AM	syb_default_pool	4	5.0049999999999999	0.14583333333333334	99.854166666666671	0
Oct 10 2016 5:30AM	syb_system_pool	3	0	0.0064814814814814813	99.993518518518528	0
Oct 10 2016 6:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 6:30AM	syb_default_pool	4	4.9836111111111112	0.15069444444444444	99.849305555555546	0
Oct 10 2016 6:30AM	syb_system_pool	3	0	0.0064814814814814813	99.993518518518528	0
Oct 10 2016 7:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 7:30AM	syb_default_pool	4	4.9572222222222226	0.14166666666666666	99.85833333333332	0
Oct 10 2016 7:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 8:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 8:30AM	syb_default_pool	4	4.9061111111111115	0.15138888888888888	99.848611111111126	0
Oct 10 2016 8:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 9:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 9:30AM	syb_default_pool	4	4.9227777777777781	0.13194444444444445	99.868055555555557	0
Oct 10 2016 9:30AM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 10:30AM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 10:30AM	syb_default_pool	4	5.0186111111111114	0.17430555555555557	99.825694444444451	0
Oct 10 2016 10:30AM	syb_system_pool	3	0	0.008333333333333332	99.991666666666666	0
Oct 10 2016 11:30AM	syb_blocking_pool	4	0	0.0007183908045977011 4	99.999281609195407	0
Oct 10 2016 11:30AM	syb_default_pool	4	2.7058333333333335	0.29525862068965514	99.704741379310349	0
Oct 10 2016 11:30AM	syb_system_pool	3	0	0.009578544061302683	99.990421455938687	0
Oct 10 2016 12:30PM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 12:30PM	syb_default_pool	4	5.1516666666666664	0.12777777777777778	99.872222222222234	0
Oct 10 2016 12:30PM	syb_system_pool	3	0	0.007407407407407077	99.992592592592587	0
Oct 10 2016 1:30PM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 1:30PM	syb_default_pool	4	36.679861111111109	0.28160919540229884	99.718390804597689	0
Oct 10 2016 1:30PM	syb_system_pool	3	0	0.047892720306513405	99.952107279693493	0
Oct 10 2016 2:30PM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 2:30PM	syb_default_pool	4	11.863055555555556	5.7494969818913475	94.25050301810866	0
Oct 10 2016 2:30PM	syb_system_pool	3	0	0.017246335153779824	99.982753664846214	0
Oct 10 2016 3:30PM	syb_blocking_pool	4	0	0	100	0
Oct 10 2016 3:30PM	syb_default_pool	4	27.572777777777777	9.2652777777777775	90.734722222222217	0
Oct 10 2016 3:30PM	syb_system_pool	3	0	0.010185185185185184	99.989814814814821	0

PDBA 솔루션 요약

➢ PDBA / PDW 솔루션 특징

구 분	솔루션 특징
✓ PDBA 솔루션 장점 (DB서버 단위 모니터링)	▪ DB 서버에서 실행되는 모든 SQL 실시간 수집 및 분석 ▪ 네트워크 스니핑 기능에 의한 최소한의 부하로 모든 SQL 수집 ▪ 실행된 Input 파라메터를 포함한 모든 SQL을 실시간 수집하고 검색 ▪ SQL의 단계별 처리시간 분석 및 성능지표 Data와 연동 검색 기능 ▪ 과거 DB 시스템의 운영 환경에 대한 Replay 기능(TimeLine) ▪ 서버 자원과 SQL에 대한 실시간 연동 모니터링 기능 ▪ CPU, Engine 및 Memory, Cache 사용량 모니터링 ▪ 프로세스 별 실시간 사용현황 모니터링 ▪ Top-N 형태의 다양한 리포트 및 다양한 실시간 Alert 제공
✓ PDW 솔루션 장점 (DB서버 통합 모니터링)	▪ 개별 단위 DB 서버의 운영 현황을 통합 관리하는 기능 ▪ 복수의 Mater, Replica 의 SQL 및 DB 운영 정보에 대한 통합관리 ▪ 대규모로 DB서버를 운용하는 고객 환경에 꼭 필요한 통합 솔루션 ▪ 과거 실행된 SQL(Input 파라메터 포함) 및 로그인 정보에 대한 장시간 보유가 요구되는 고객에 필요한 기능 ▪ SQL을 포함하여 수집된 Flat Files에 대한 자동 이관 및 DB 적재 지원 ▪ SAP IQ 고성능 DW 전문 DB로 DATA 압축 저장 및 조회 지원

Thank you!

주식회사 데이터웍스

http://www.dataworks.co.kr

서울특별시 성동구 뚝섬로 1길 31 서울숲 M타워 1308호

대표전화: 02-2292-5096, 팩스: 02-2292-5081