



MAXGAUGE
Practical Guide



Contents

1 Configuration

- | Alert 설정 가이드
- | Repeat / Interval 에 따른 발송 규칙
- | SMS / Mail 연동

2 SQL Analysis | Access Statistics

- | 테이블 내 미사용 인덱스를 확인하고 싶어요.
- | 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요.

3 SQL Analysis | Application Call Tree

- | 식별자를 사용하는 업무에서 SQL의 수정이 빈번하게 발생합니다.
- | 해당 업무 성능에 문제가 생겼다고 의심이 될 때, 업무 단위 성능추이, SQL 이력등 관련 정보를 확인하고 싶어요.

4 SQL Analysis | Elapsed Time Scatter

- | 업무시간에 5초 이상 수행되는 SQL을 추출하고 싶어요.
- | 그리고, SQL이 느린 이유를 알고 싶어요

5 SQL Analysis | Long-Term Analysis

- | 특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요.
- | SQL 개선 후, 상태를 체크하고 싶어요.

Contents

6 SQL Analysis | Plan Diff

- | SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요.
- | 통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요.

7 SQL Analysis | Wait Time Analysis

- | Real-Time Monitor에서 SQL Elapsed Time이 증가하였습니다.
- | Wait Time이 긴 SQL들을 확인하고 싶어요.

8 Trend Analysis | Performance

- | CPU가 급격히 증가한 원인을 알고 싶어요 .
- | RAC 간의 Load Balance 및 성능을 확인 하고 싶어요.
- | 한 달간 가장 큰 부하의 원인을 알고 싶어요.

9 Trend Analysis | 1-day Summary

- | 어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)
- | Peak 시간대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)

10 Trend Analysis | Parameter

- | 배치업무가 느려진 이유를 알고 싶어요.

Contents

- 11 Data Visualization | Data Path View**
 - | 어떤 노드가 업무 비중도가 높은가요?
 - | 자꾸 GC 이벤트가 발생해요. 업무 분산이 잘 돼 있는지 확인할 수 있는 방법을 알고 싶어요.
 - | 특정 SQL이 각 노드에서 수행되는 비중을 쉽게 알 수 있는 방법을 알고 싶어요.

- 12 Data Visualization | Hotspot view**
 - | 출장 기간 동안 리소스에 부하가 언제 발생 했는지 확인하고 싶어요.

- 13 Power Comparison | Event Comparison**
 - | 전 주에 비해 늘어난 이벤트와 그 원인을 알고 싶어요. (N:M 분석)
 - | 신규로 추가 된 업무가 발생 시키는 대기 현상이 궁금해요.(1:M 분석)

- 14 Power Comparison | Top SQL Comparison**
 - | 신규로 추가 된 업무 성능이 궁금해요.
 - | 지난 달과 이번 달의 월 단위 작업을 비교하고 싶어요.

- 15 Power Comparison | Trend Comparison**
 - | Storage 이전 작업 전후의 성능을 비교하고 싶어요.
 - | 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요.

Contents

16 Capacity Planning

| TABLESPACE ALARM이 발생 했는데 원인을 알고 싶어요.

| 특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요.

17 Script | Script Alert

| 업무시간에 Backup이 수행되는 것을 방지하기 위해 Alert을 등록하고 싶어요.

| 평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요.

18 Script | User Script

| Process 와 Session 의 사용량(Usage %)을 알고 싶어요.

19 Script | Maxgauge Script

| Real-Time Monitor에서 Tablespace Alert 이 발생하였습니다.

현재 시점의, Tablespaces 사용정보를 확인하고 싶어요.

Contents

Alert Configuration

개요 / 실시간 감지 / 사후 분석

Alert 사용자 가이드

1. Alert 설정 가이드
2. Repeat / Interval 에 따른 발송 규칙
3. SMS / Mail 연동

Alert Configuration

모든 성능 지표에 대한 **Alert** 설정 가능

효율적인 임계 치 설정을 통한 장애 징후
사전 감지 «



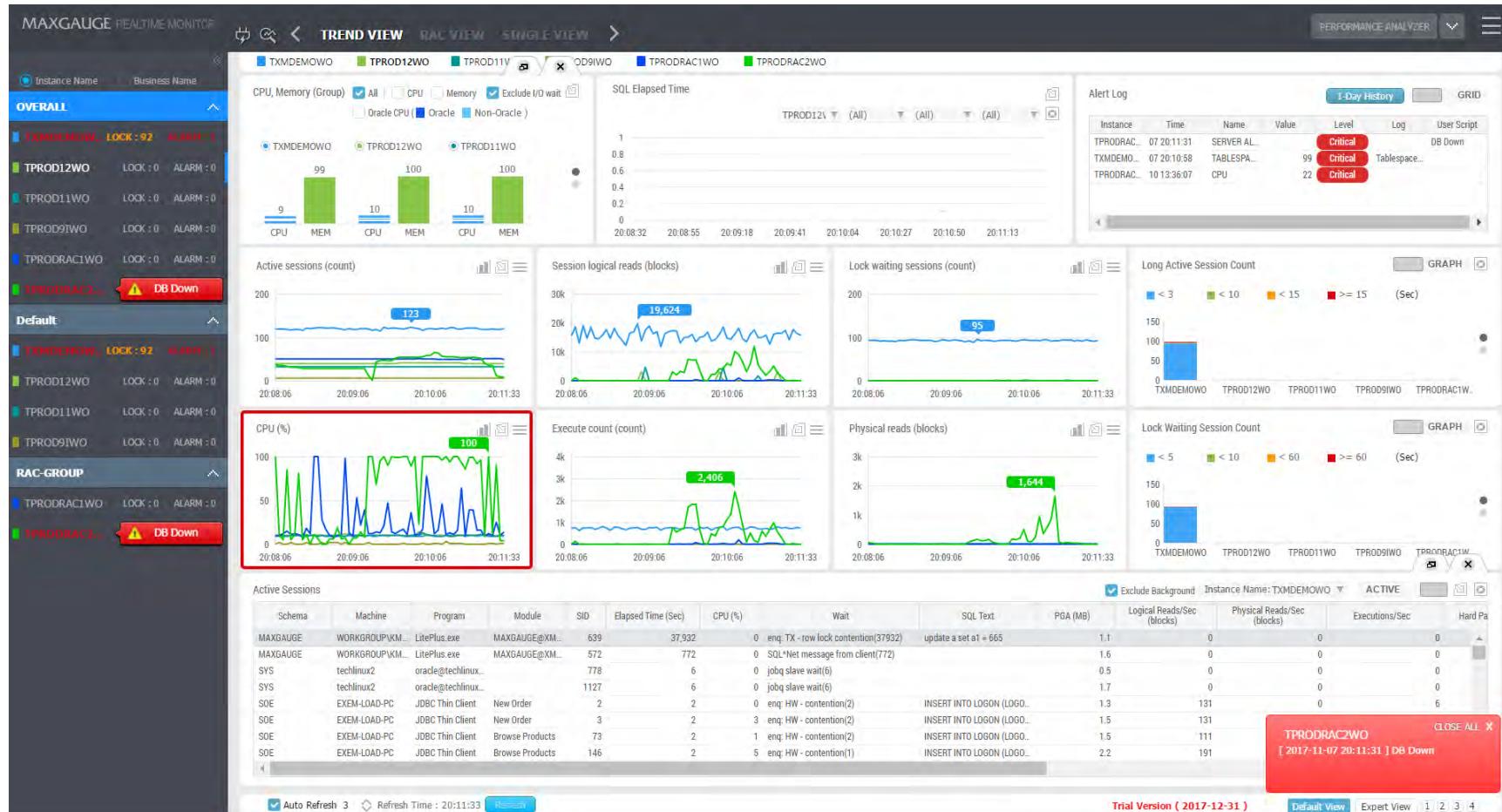
- ✓ Instance 별로 임계치 설정 가능
- ✓ Stat, Wait Event, Resource, Oracle Alert Log, File System, Tablespace 임계치 설정 가능
- ✓ Repeat / Interval 설정에 따라 Check 주기 변경 가능
- ✓ 모든 성능 지표에 대하여 SMS / Mail 연계



Alert Configuration?

Config에서 설정 된 Alert에 대하여 실시간 감지

- ✓ 현재 모니터링 중인 모든 instance들 중에서 DB Down, MaxGauge Agent (RTS) Down, Listener Stop시에 실시간으로 Alarm을 제공합니다.
- ✓ 설정한 지표 별 임계 값을 만족하는 Alarm 발생 시, Trend Chart에서는 화면 색상이 변경 됩니다. (Warning - 노란색 / Critical - 빨간색)



Alert Configuration?

발생 되었던 Alert에 대하여 **사후 분석**

- ✓ Oracle Alert Log 발생 이력을 제공해 줍니다.
- ✓ Configuration 화면에서 설정한 지표 별 임계 값 (threshold value)을 조건에 만족하는 횟수를 시간 별 추이 그래프 및 그리드로 제공합니다.

The screenshot displays the MAXGAUGE Performance Analyzer interface, specifically the Alert Log module. The top navigation bar includes tabs for 1-Day Summary View, Performance Trend, Session List, SQL List, Long-Term Trend, Parameter, Alert Log (which is selected), and Realtime Monitor. The main content area shows the following details:

- Alert Log:** Displays a list of 14 log entries from November 3, 2017, at 10:08:28. Each entry is a repeating ORA-1653 error message indicating that the table SOE.LOGON could not be extended by 128 bytes in the SOE tablespace.
- Alarm:** A chart showing the number of critical alerts over time. The Y-axis represents the count of alerts (0 to 1.3k), and the X-axis shows dates from November 1 to November 7, 2017. The chart shows a significant spike on November 7, 2017, reaching approximately 871 critical alerts.
- Alarm Grid:** A table showing the daily count of critical and warning alerts for each day from November 1 to November 7, 2017.
- Detailed Alert History:** A table listing specific alert events with columns for Alert Time, Event Name, Value, Level, and Description. The events listed are all related to physical read bytes (bytes) and are marked as Critical.

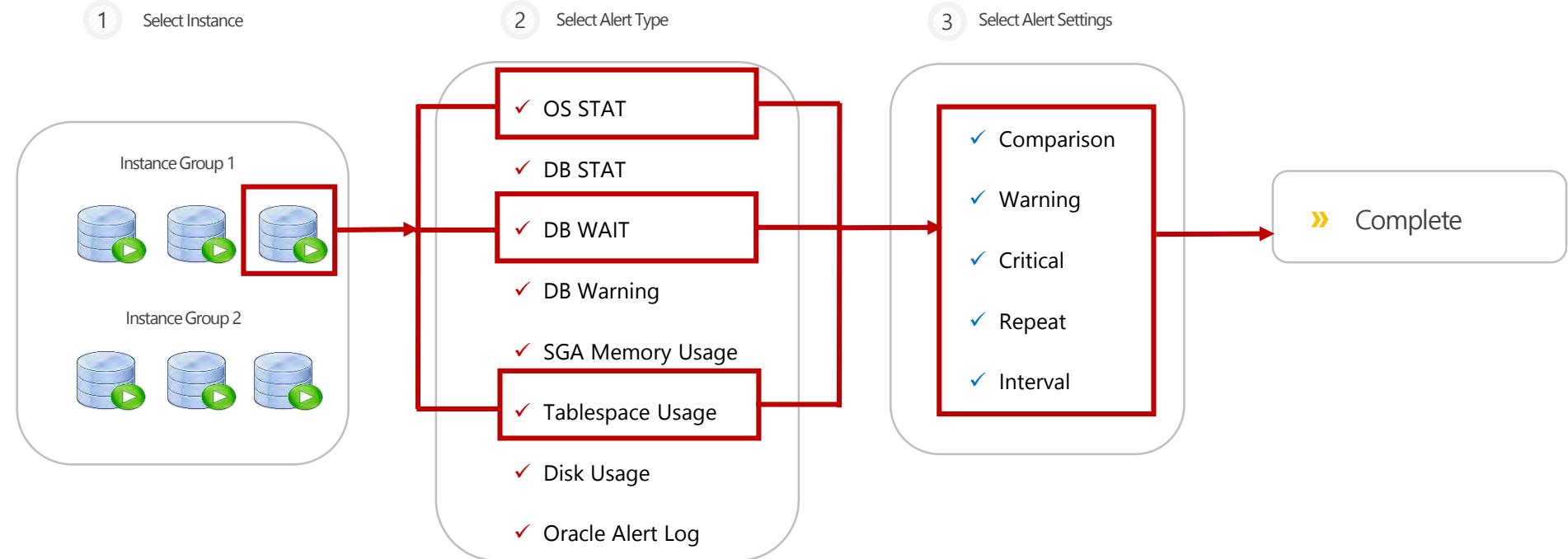
Alert 사용자 가이드

Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

» Alert 등록 순서



- 1 Alert 을 적용할 Instance 를 선택합니다.
- 2 Alert Type 과 Alert Name을 선택 합니다.
- 3 임계 값과 Alert Setting 상세에 대하여 설정 하여 줍니다.

Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

» Alert Type 구분



1 OS STAT / DB STAT / DB WAIT



2 DB Warning



3 SGA Memory Usage



4 Disk Usage



5 Tablespace Usage



6 Oracle Alert Log

1 DB STAT, Wait Event, OS 영역에 대한 항목.

2 Database Alert 영역

-Lock Wait Duration : Lock의 지속시간을 초단위로 설정

-Latch Wait Session : Latch 이벤트를 대기하는 세션들의 개수

-Library Cache Pin/Lock : Library Cache Pin/Lock 두 이벤트의 합

3 SGA 의 Free Memory 영역에 대한 Alert 설정

-Shared pool / Large pool / Java pool free memory의 low limit 설정

4 OS Disk Volume 임계 설정

5 DB Tablespace Usage 임계 설정

6 Oracle Alert Log 발생 이력의 임계 설정

Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

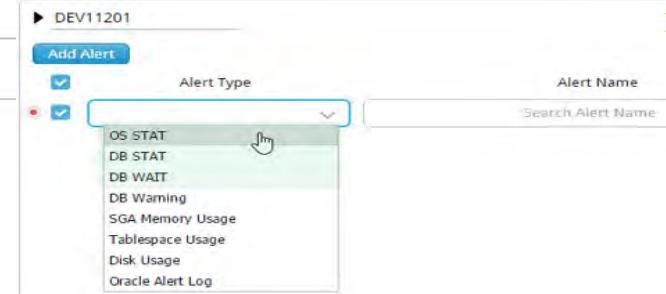
» Instance Alert 설정

OS STAT/DB STAT/DB Wait

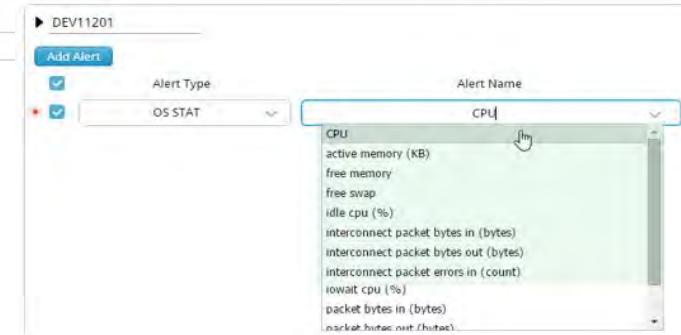
1 Alert 대상 Instance를 선택한다.



2 Alert Type의 OS STAT/DB STAT/DB Wait 중 필요 Type 선택한다.



3 Alert 대상을 선택한다.(Alert Name)



4 Alert Settings – 임계 값 / 부등호 / 발생 주기를 설정 해 준다.

Warning	Critical	Comparison	Repeat	Interval
80	90	>=	3	5
		<=		

5 Save All 버튼 클릭



Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
 - Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
 - SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

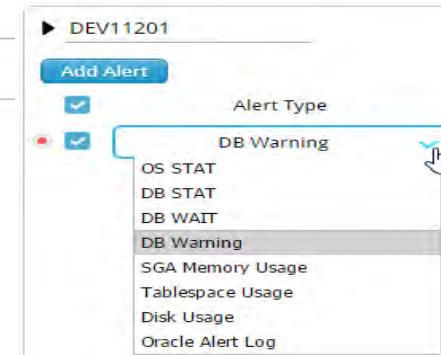
» Instance Alert 설정

DB Warning

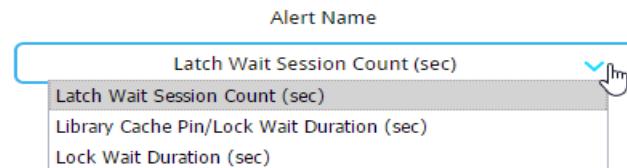
1 Alert 대상 Instance를 선택한다.



2 Alert Type 의 "DB Warning" 을 선택한다.



3 Alert 대상을 선택한다.(Alert Name)



4 Alert Settings – 임계 값을 설정 해 준다.



5 Save All 버튼 클릭



항목	설명
Alert Type	<ul style="list-style-type: none"> ● DB Warning
Alert Name	<ul style="list-style-type: none"> ● Lock Wait Duration (sec) : Lock의 지속시간을 초단위로 설정. ● Latch Wait Session Count (sec) : Latch 이벤트를 대기하는 세션들의 개수. ● Library Cache Pin / Lock (sec) : Library Cache Pin/Lock의 두 이벤트를 합한 시간.

Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

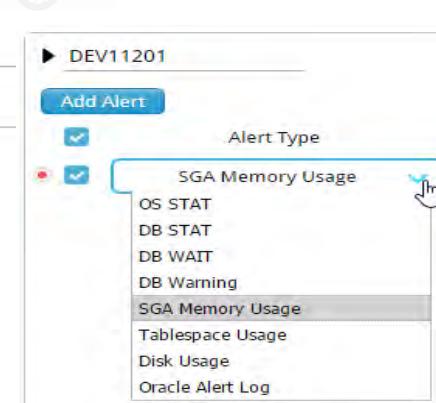
» Instance Alert 설정

SGA Memory Usage

1 Alert 대상 Instance를 선택한다.



2 Alert Type 의 "SGA Memory Usage" 를 선택한다.



3 Alert 대상을 선택한다.(Alert Name)



4 Alert Settings – 임계 값 / 부등호/발생 주기를 설정 해 준다.

Warning	Critical
80	90

5 Save All 버튼 클릭



항목	설명
Alert Type	<ul style="list-style-type: none"> ● SGA Memory Usage
Alert Name	<ul style="list-style-type: none"> ● Shared pool free memory: Shared pool free memory의 low limit을 설정 ● Large pool free memory: Large pool free memory의 low limit을 설정 ● Java pool free memory: Java pool free memory의 low limit을 설정

Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정

- Repeat / Interval에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙

- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

» Instance Alert 설정

Tablespace Usage

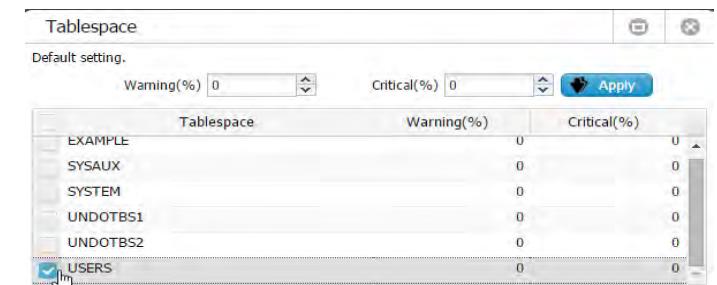
1 Alert 대상 Instance를 선택한다.



2 Alert Type의 "Tablespace Usage"를 선택한다.



3 Alert 대상을 선택한다.(Alert Name) – 팝업 창



4 Alert Settings – 임계 값을 설정 해 준다.

Tablespace	Warning(%)	Critical(%)
EXAMPLE	0	91
SYSAUX	0	0
SYSTEM	0	0
UNDOTBS1	0	0
UNDOTBS2	0	0
<input checked="" type="checkbox"/> USERS	90	95

5 OK 버튼 클릭한다.



6 발생 주기를 설정한다.

Repeat	Interval
<input type="text" value="3"/>	<input type="text" value="5"/>

7 Save All 버튼 클릭한다.



Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
 - Repeat / Interval에 따른 발생 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

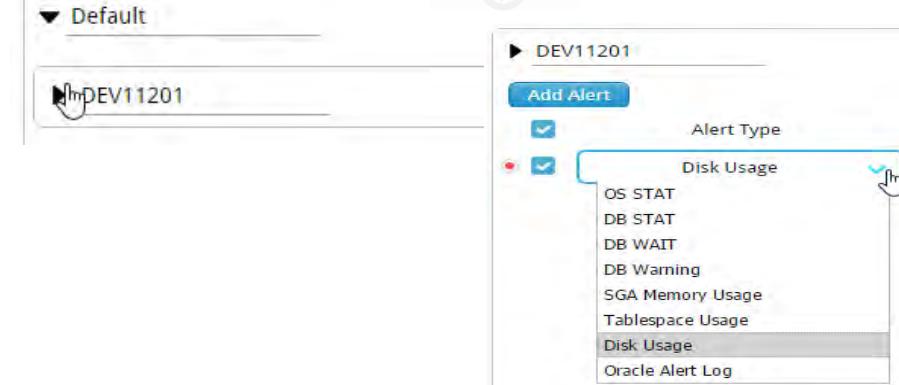
» Instance Alert 설정

Disk Usage

1 Alert 대상 Instance를 선택한다.



2 Alert Type의 "Disk Usage"를 선택한다.



3 Alert 대상을 선택한다.(Alert Name) – 팝업 창

Disk Usage

Default setting.

	Disk	Mount	Warning(%)	Critical(%)
<input type="checkbox"/>	tmpfs	/dev/shm	0	91
<input type="checkbox"/>	/dev/mapper/VolGroup00-L...	/	0	0
<input checked="" type="checkbox"/>	/dev/sda1	/boot	90	95

4 Alert Settings – 임계 값을 설정 해 준다.

	Disk	Mount	Warning(%)	Critical(%)
<input type="checkbox"/>	tmpfs	/dev/shm	0	91
<input type="checkbox"/>	/dev/mapper/VolGroup00-L...	/	0	0
<input checked="" type="checkbox"/>	/dev/sda1	/boot	90	95

5 OK 버튼 클릭한다.



6 발생 주기를 설정한다.

Repeat	Interval
3	5

7 Save All 버튼 클릭한다.



Alert 설정 가이드

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
 - Repeat / Interval에 따른 발생 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

» Instance Alert 설정

Oracle Alert Log

1 Alert 대상 Instance를 선택한다.



2 Alert Type의 "Oracle Alert Log"를 선택한다.



3 Alert 대상을 선택한다. (Alert Name) – 팝업 창

Oracle Alert Log /u01/app/oracle/rdbms/RAC1/udump/alert_RAC1.lo

Alert Filtering Options

Log Switch	Include	Critical
DDL	Include	None
Checkpoint	Include	Warning
DDL	Include	Critical
Archiving	Include	None
Others	Include	None

4 Add Custom Option을 설정한다. (선택사항)

Add Custom Options (Exclude Priority)

Add Alert	Text	Pattern	Alert
<input checked="" type="radio"/>	ORA-04031	Exclude	None

5 OK 버튼 클릭한다.



6 발생 주기를 설정한다.

Repeat	Interval
3	5

7 Save All 버튼 클릭한다.



Repeat / Interval에 따른 발송 규칙

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정

- Repeat / Interval에 따른 발송 규칙

- Alert 발생 관련 파라미터
- Alert 발생 규칙

- SMS / Mail 연동

- SMS / Mail Process
- SMS / Mail 등록 순서

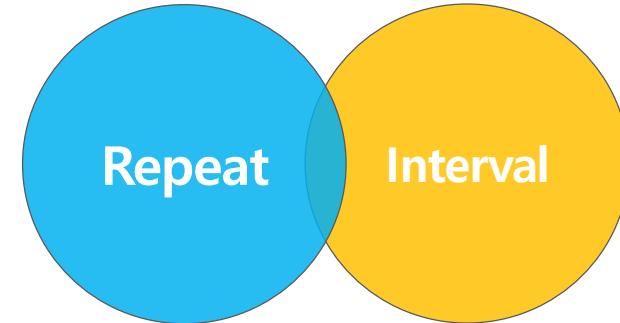
» Alert 발생 관련 파라미터

▼ TPROD11WO

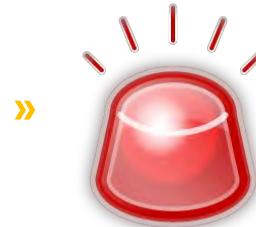
Add Alert		Alert Name	Warning	Critical	Comparison	Repeat	Interval (Sec)
<input checked="" type="checkbox"/>	Alert Type	lock waiting sessions (count)	2	5	>=	3	3
<input checked="" type="radio"/>	DB STAT						
<input type="button" value="Save All"/> <input type="button" value="Delete"/> <input type="button" value="Copy To"/>							



Instance Alert 설정



Repeat Interval



일람 발생

- ✓ DG는 Alert 등록 시, 파라미터 설정 값을 이용하여 Alarm 발생 여부를 결정한다. (알람발생 기준)
- ✓ Repeat :연속적으로 Repeat 설정된 횟수 이상 alarm 발생 조건을 만족해야 alarm 이 발생 된다.
- ✓ Interval : DG가 성능 데이터에 대한 Alarm 발생 조건 만족 여부를 체크하는 주기를 설정한다. (단위 : sec)

Repeat / Interval에 따른 발송 규칙

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval에 따른 발송 규칙

» Alert 발생 규칙

1. Repeat count가 0으로 초기화 되는 기준

- Repeat count가 지정한 수치에 도달하는 순간
- Alert(N/W/C)이 발생하는 순간 (W->C, C->W는 제외)
- Normal이 체크되는 순간

※ W->C, C->W Level이 변하는 경우 Repeat Count가 초기화되지 않음

따라서, W,C의 경계를 왔다 갔다 하는 경우 Alert Check 시점의 Level로 Alert이 발생함

2. 알람 발생 기준

- repeat count 도달 시 레벨이 이전 알람 발생 레벨과 다를 경우
- Normal은 repeat count에 도달하지 않아도 이전 Alert 레벨(W/C)과 다르면 무조건 발생
- Normal은 Alert log history 창에 표현되지 않으며, ora_alarm_history 테이블에만 기록됨



Alert Log

Instance	Time	Name	Value	Level	Log
3 TWEH11WP	29 11:07:30	LOCK WAIT...	3	Warning	
2 TWEH11WP	29 11:07:06	LOCK WAIT...	4	Critical	
1 TWEH11WP	29 11:06:27	LOCK WAIT...	2	Warning	

✓ 시나리오 (Alert 설정 파라미터)

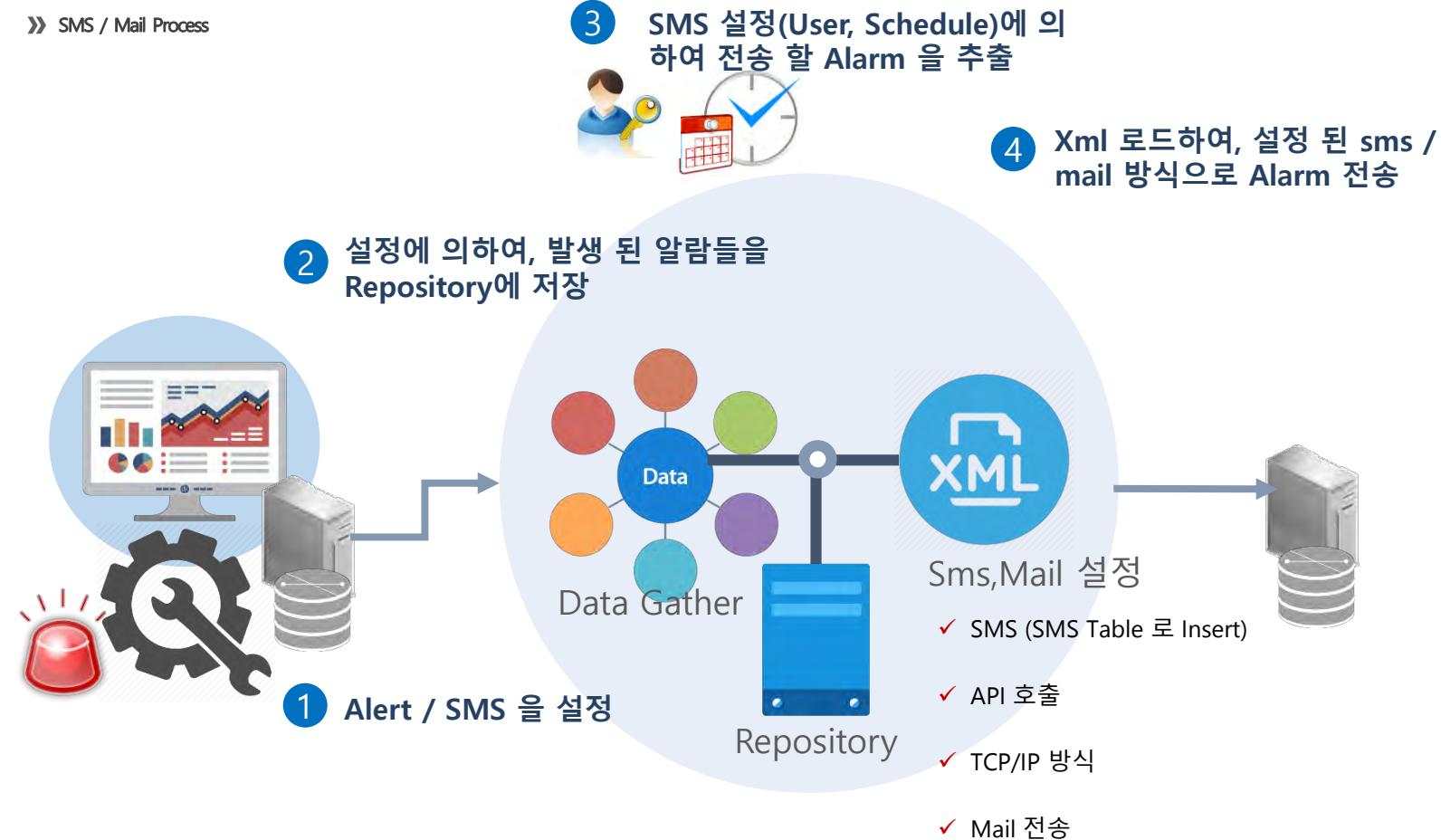
INSTANCE	: TWEH11WP
Alert Name	: DB STAT – Lock Waiting Sessions (count)
Warning / Critical	: 2 / 5
Repeat / Interval	: 3 / 3

SMS / Mail 연동

Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
- Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동
 - SMS / Mail Process
 - SMS / Mail 등록 순서

» SMS / Mail Process

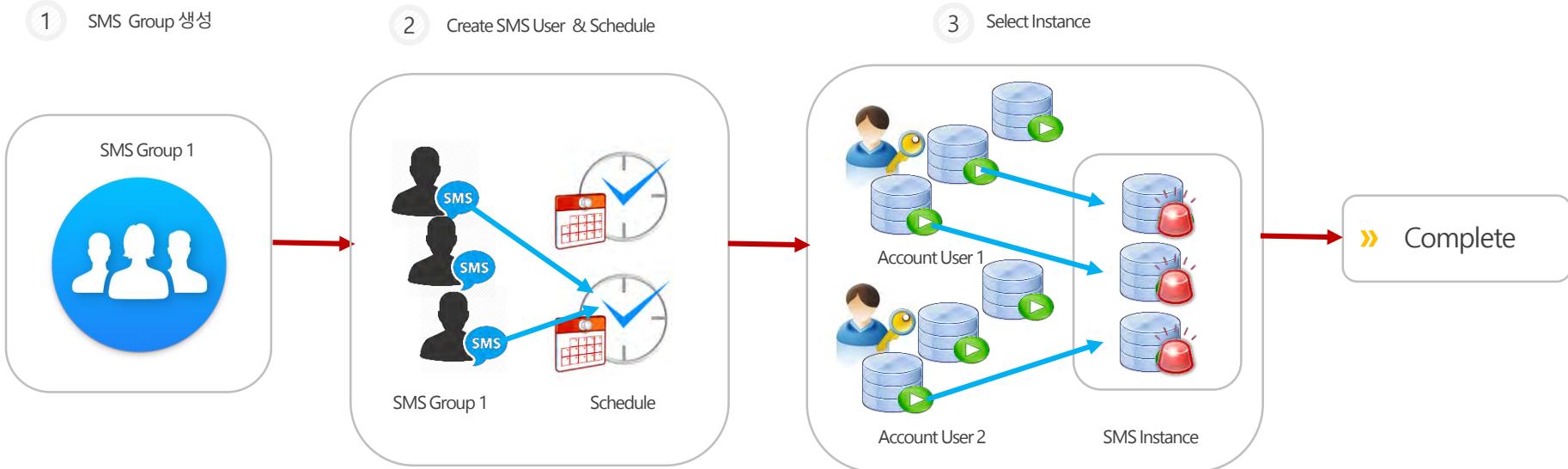


SMS / Mail 연동

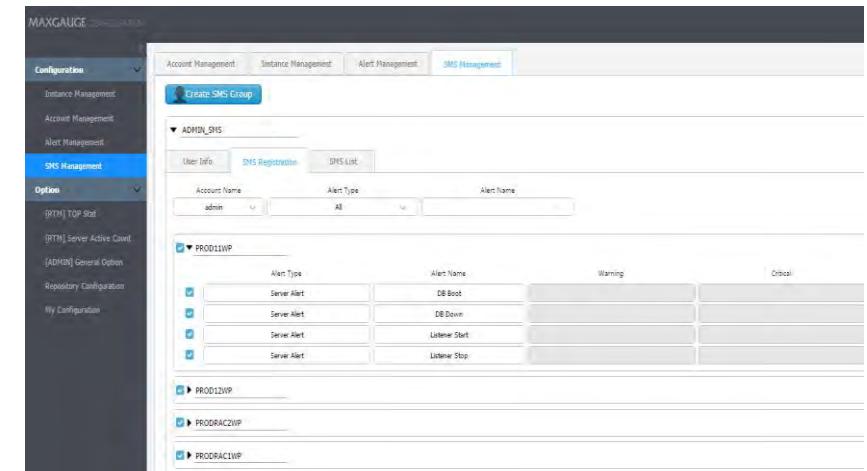
Alert Configuration

- Alert 설정 가이드
 - Alert 등록 순서
 - Alert Type 구분
 - Instance Alert 설정
 - Repeat / Interval 에 따른 발송 규칙
 - Alert 발생 관련 파라미터
 - Alert 발생 규칙
- SMS / Mail 연동

» SMS / Mail등록 순서



- 1 SMS Group 을 생성 합니다. (SMS Management 화면)
- 2 SMS User 와 SMS Schedule 생성하여, 설정해 줍니다.
- 3 SMS 로 전송받을 Instance 및 Alert 을 선택하여 설정 완료 해 줍니다.



MAXGAUGE



Practical Guide

Access Statistics [Performance Analyzer]

Contents

Access Statistics?

Access Statistics의 활용

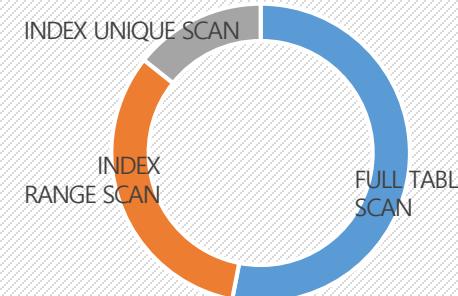
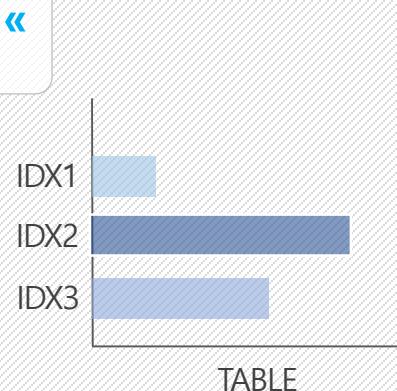
Case 1. 테이블 내 미사용 인덱스를 확인하고 싶어요

Case 2. 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을
뽑아내고 싶어요

Access Statistics?

Access Statistics는 언제 쓰나요?

특정 테이블의
인덱스 사용 빈도가 궁금할 때.

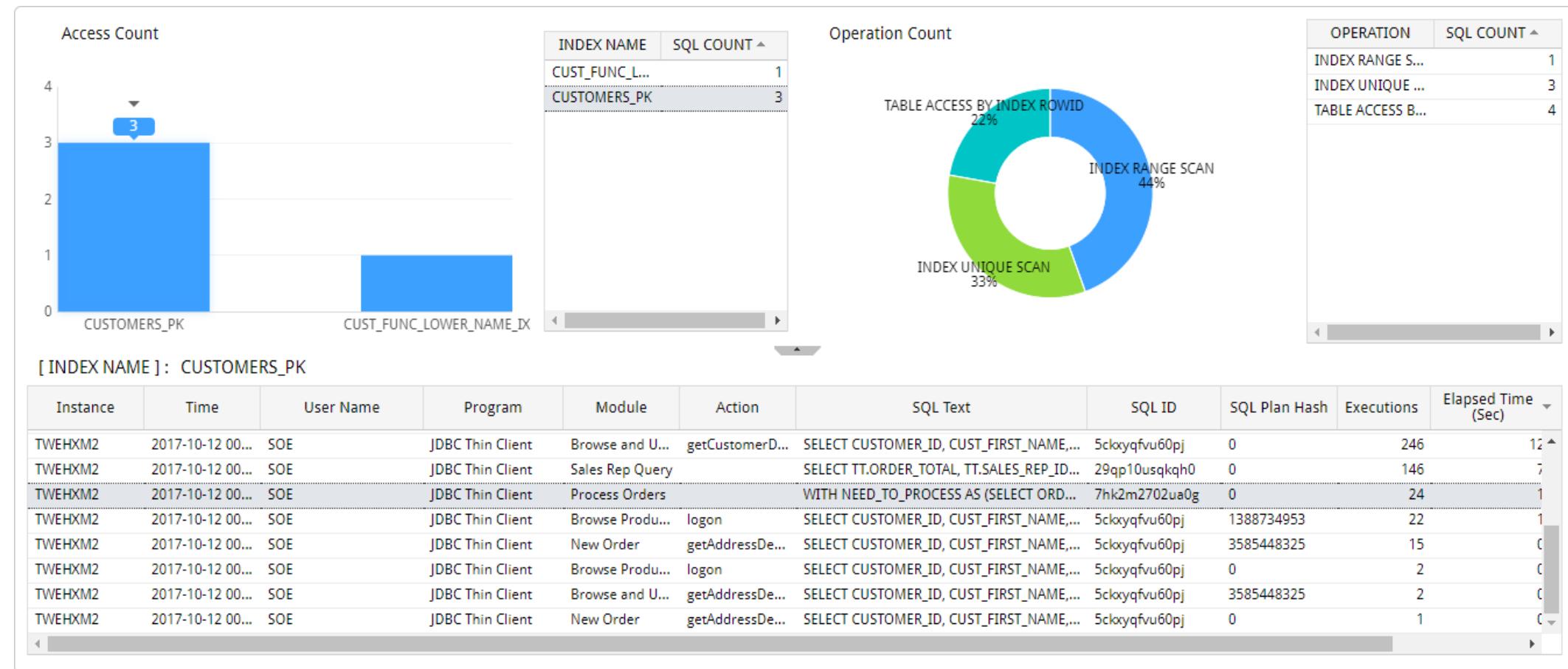


» 특정 테이블의 오퍼레이션 별 수행 횟수를
바탕으로 튜닝 대상 선별.

Access Statistics?

Access Statistics

- 선택한 기간 동안 특정 오브젝트의 액세스 건 수를 확인할 수 있는 화면입니다.
- 인덱스 / 오퍼레이션 별 액세스 건 수를 그래프와 그리드로 표현해줍니다.
- 그래프(또는 그리드)를 클릭하면, 지정된 인덱스(오퍼레이션)를 사용하는 SQL 들이 하단 그리드에 나열되어 각 SQL의 세부정보를 확인할 수 있습니다.
- 해당 기능은 Performance Analyzer ▶ SQL Analysis ▶ Object Analysis 경로를 통해 사용할 수 있습니다.



Access Statistics의 활용

Access Statistics의 사용 방법

Access Statistics

- 검색 조건 설정
- 데이터 분석

» Access Statistics 검색 화면

① 검색 대상 Instance를 선택합니다.

② 검색할 오브젝트 타입(Table/Index)를 정하고, Owner 와 Object 를 선택합니다.

③ 검색 기간을 지정해줍니다, 하루/한주/한달 또는 달력을 클릭해 직접 기간을 지정할 수 있습니다.

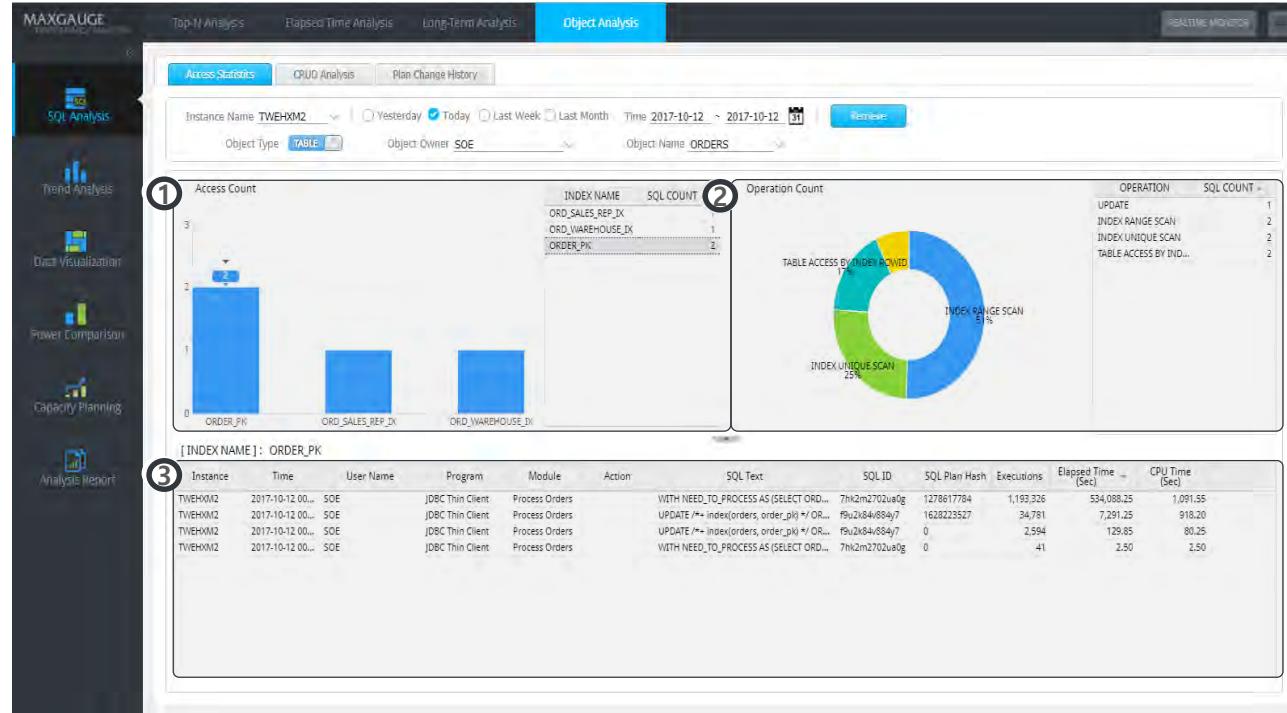
④ 위의 설정 조건으로 Access Statistics를 실행합니다.

Access Statistics의 사용 방법

Access Statistics

- 검색 조건 설정
- 데이터 분석

» Access Statistics 검색 화면



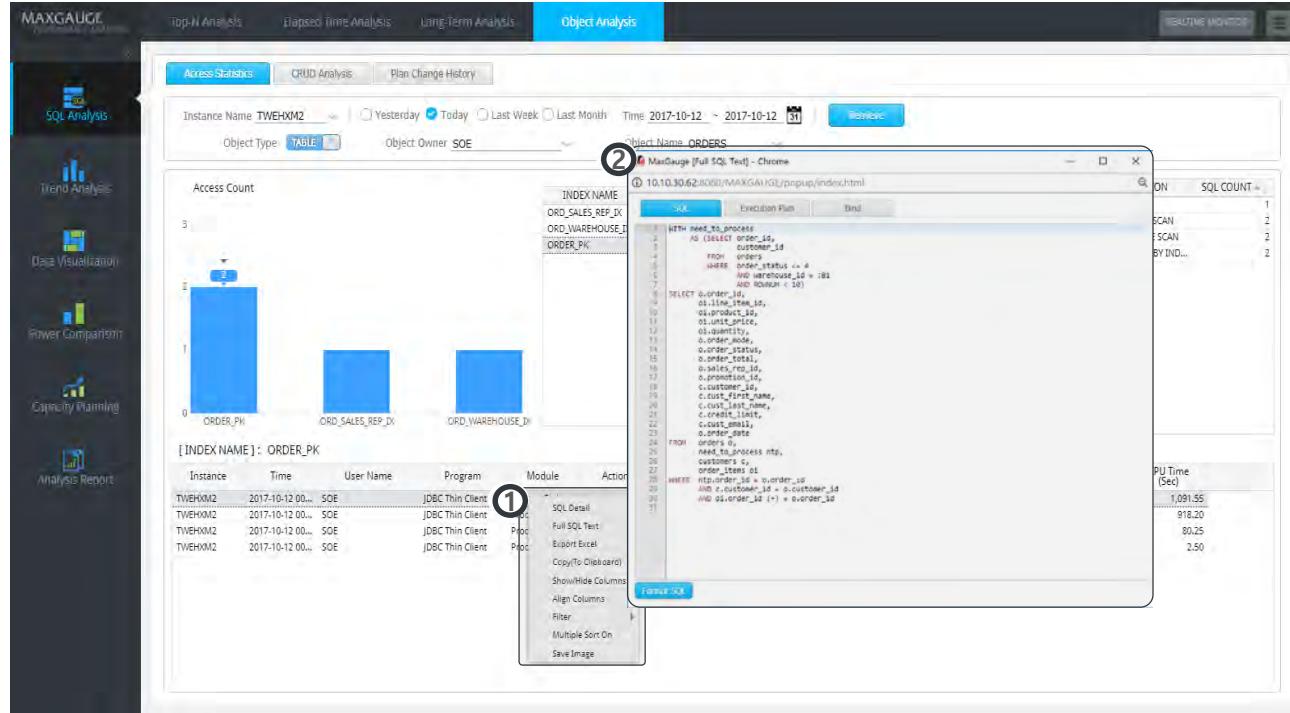
- 1 테이블에 포함된(Type 이 인덱스라면 선택된) 인덱스의 Access Count 를 그래프와 그리드로 보여줍니다.
- 2 오브젝트를 수행하는 Operation 별 건 수를 그래프와 그리드로 보여줍니다.
- 3 1,2 번 항목에서 클릭한 그래프(그리드)의 SQL 리스트들을 응답시간이 큰 순서로 정렬해서 보여줍니다.

Access Statistics의 사용 방법

Access Statistics

- 검색 조건 설정
 - 데이터 분석

» Access Statistics 검색 화면



- 1 하단의 그리드에서 분석 대상 SQL 을 우 클릭하여 팝업 메뉴를 호출합니다.
 - 2 팝업 메뉴의 Full SQL Text 항목을 선택하여, SQL Full Text 와 실행계획, 바인드 변수 정보를 확인할 수 있습니다.

실전 분석 사례 Case 1.
테이블 내 미사용 인덱스를
확인하고 싶어요

“ 테이블 내 미사용 인덱스를 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 검색 결과 확인
- 메타정보와 비교하여 미사용 인덱스 검출

» Access Statistics 검색 화면

- 1 비교 대상 Instance (**TWEHXM2**)를 선택합니다.
- 2 Object Type 을 TABLE로 놓고, Object Owner 와 Object Name 을 선택합니다.
- 3 검색 기간을 최근 한 달(**Last Month**)로 선택합니다.
- 4 위의 설정 조건으로 **Access Statistics**를 실행합니다.

✓ 시나리오

INSTANCE : TWEHXM2
SCHEMA : SOE
TABLE : ORDERS
검색 기간 : 최근 한 달

✓ 목표

최근 한 달 지정된 테이블이 포함된 SQL에서 어떤 인덱스를 사용했는지 확인하고, 미사용 인덱스를 제거하여 여유 공간을 확보한다.

“ 테이블 내 미사용 인덱스를 확인하고 싶어요 ”

STEP

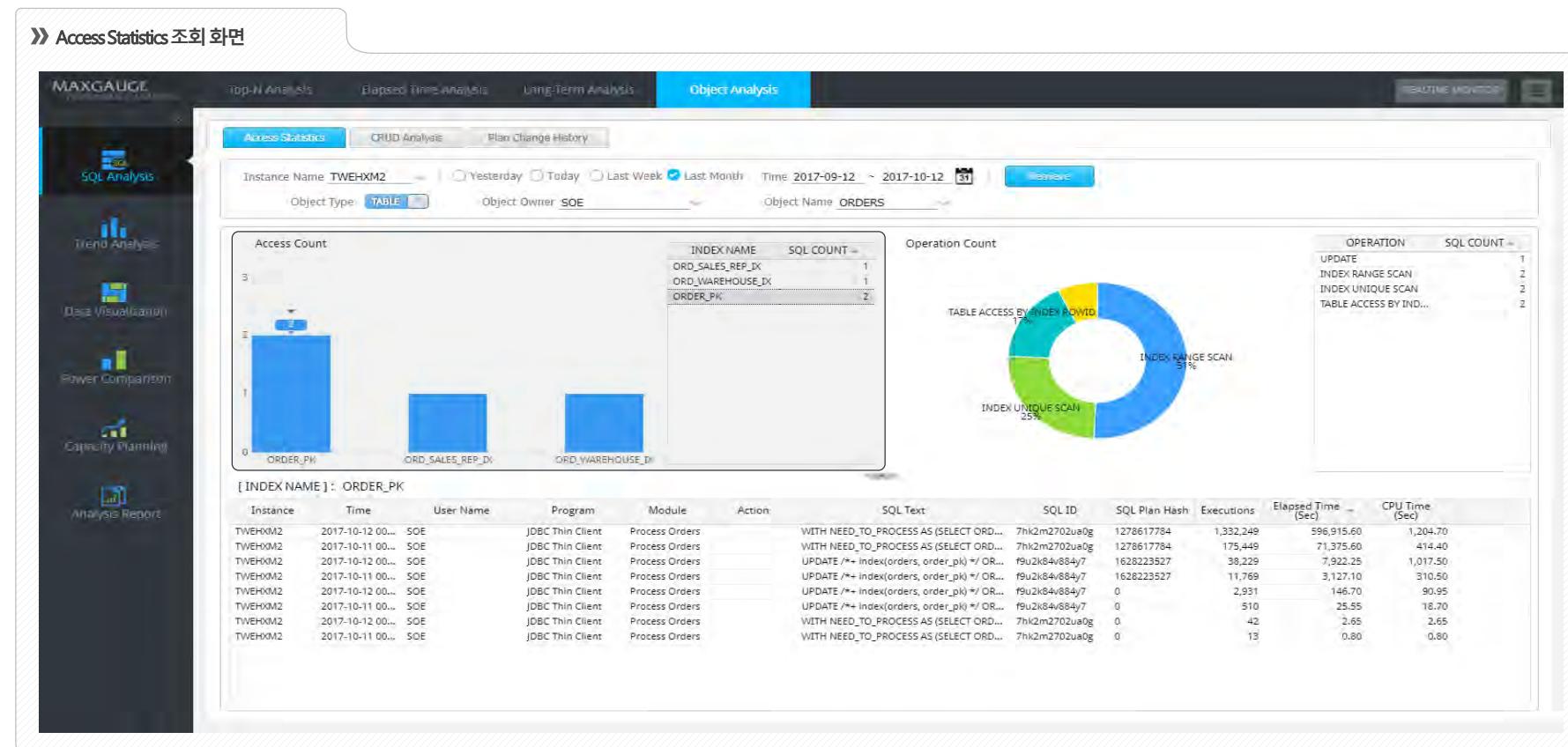
1. 검색 조건 설정

2. 데이터 분석

▪ 검색 결과 확인

▪ 메타정보와 비교하여 미사용

인덱스 검출



✓ 조회된 화면 중 **Access Count** 창에서 테이블에 속한 인덱스들의 사용 빈도를 확인할 수 있습니다.

✓ 실제 테이블에 속한 인덱스와 비교하기 위해, 단축키(**Ctrl+L**)을 눌러 제품에 내장된 Query Tool 을 호출합니다. (계속)

“ 테이블 내 미사용 인덱스를 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

▪ 검색 결과 확인

- 메타정보와 비교하여 미사용 인덱스 검출

» Access Statistics 조회 화면

Access Count

INDEX	COLUMN_NAME
SOE.ORDER_PK	ORDER_ID
SOE.ORD_SALES REP_IK	SALES REP_ID
SOE.ORD_CUSTOMER_IK	CUSTOMER_ID
SOE.ORD_ORDER_DATE_IK	ORDER_DATE
SOE.ORD_WAREHOUSE_IK	WAREHOUSE_ID
SOE.ORD_WAREHOUSE_DK	ORDER_STATUS

retrieved : 7.335 ms 6 ROWS COL : 2B ROW : 4

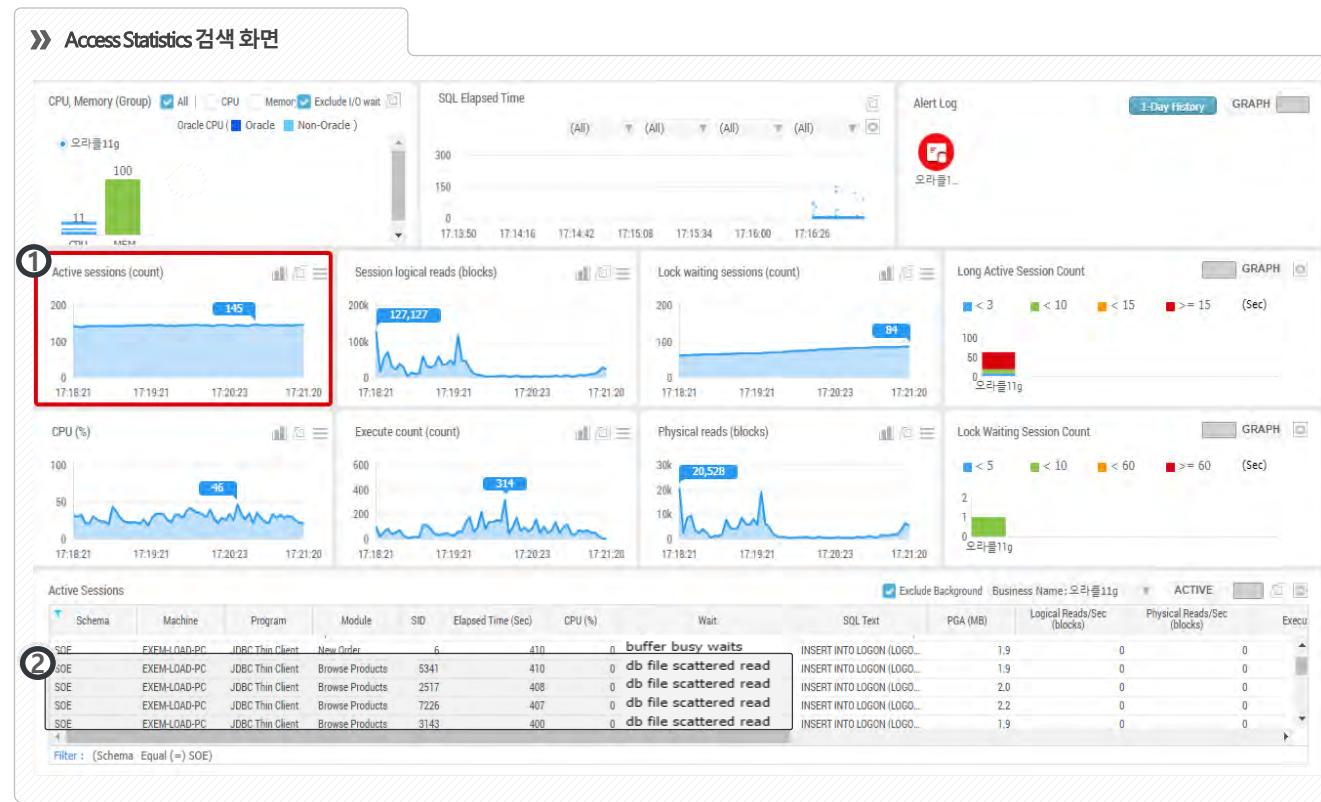
- ✓ 팝업 형태로 호출된 Query Tool에서 SQL을 이용해 해당 테이블의 전체 인덱스 정보를 조회합니다.
- ✓ Query Tool에서 조회된 결과(5건)와 제품화면의 인덱스(3건)를 비교하여 미사용 인덱스를 간추립니다.
- ✓ 업무적인 확인 절차를 거쳐 사용되지 않는 인덱스임이 확인되면, 간추려진 인덱스를 Drop 합니다.

실전 분석 사례 Case 2.
**지정된 테이블을 FULL TABLE SCAN
하는 SQL 목록을 뽑아내고 싶어요**

“ 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 ”

STEP

- 장애감지
- 검색 조건 설정
- 데이터 분석
 - Operation/SQL 확인
- Excel 파일로 내려받기



- 1 메인 차트에서 Active Sessions 지표가 임계치를 초과하여 알람이 감지되었습니다.
- 2 하단의 Active Sessions 그리드에서 확인한 결과 대부분의 Long Session이 Multi Block I/O event를 대기하고 있는 것으로 확인됩니다.

✓ 시나리오

INSTANCE : TPROD11WO
SCHEMA : SOE
TABLE : ORDERS

✓ 목표

특정 테이블을 FULL TABLE SCAN 으로 수행하는 SQL 의 리스트를 확보하여 튜닝 여부를 판단

“ 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 ”

STEP

1. 장애감지
2. 검색 조건 설정
3. 데이터 분석
 - Operation/SQL 확인
4. Excel 파일로 내려받기

» Access Statistics 검색 화면

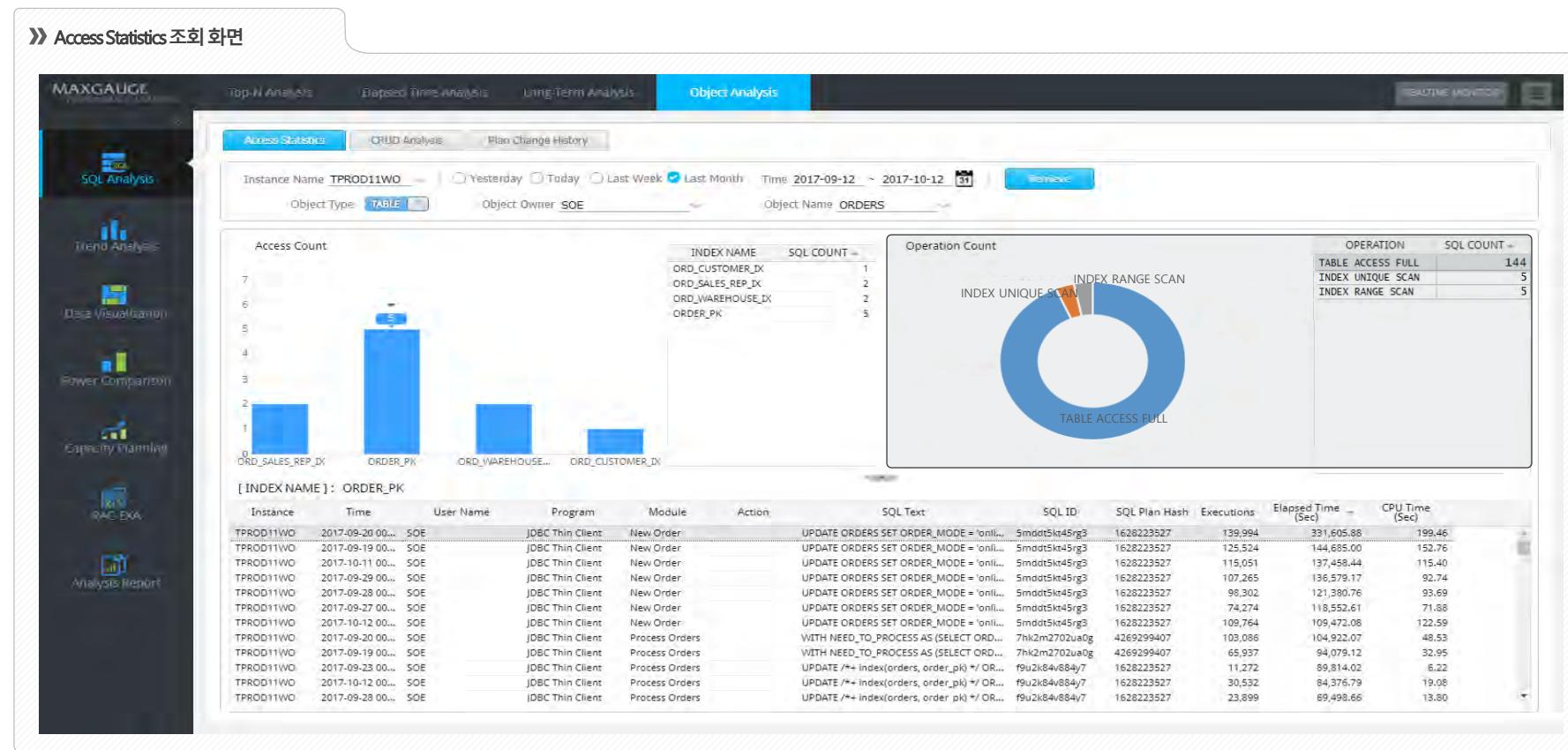
MAXGAUGE
Top-N Analysis Elapsed Time Analysis Long Term Analysis Object Analysis
Access Statistics Q/RUD Analysis Plan Change History
① Instance Name TPROD11WO ③ Yesterday Today Last Week Last Month Time 2017-09-12 ~ 2017-10-12
② Object Type TABLE ④ Object Owner SOE Object Name ORDERS
Access Count INDEX NAME SQL COUNT Operation Count OPERATION SQL COUNT
Instance Time User Name Program Module Action SQL Text SQL ID SQL Plan Hash Executions Elapsed Time (Sec) CPU Time (Sec)
Analysis Report

- 1 대상 Instance (**TPROD11WO**)를 선택합니다.
- 2 Object Type 을 TABLE 로 놓고, 대상 Object Owner 와 Object Name 을 선택합니다.
- 3 검색 기간을 최근 한 달로 선택합니다.
- 4 위의 설정 조건으로 **Access Statistics**를 실행합니다.

“ 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 ”

STEP

1. 장애감지
2. 검색 조건 설정
- 3. 데이터 분석**
 - Operation/SQL 확인
4. Excel 파일로 내려받기



- ✓ 결과 화면의 중앙 우측 Operation Count 창에서 선택한 테이블이 어떤 Operation으로 수행하였는지 표시됩니다.
- ✓ 최근 한 달 간 해당 테이블을 액세스할 때 FULL TABLE SCAN으로 대부분(93%) 수행된 것으로 확인되며, 해당 Operation으로 수행한 SQL 리스트를 하단의 프레임에서 연계하여 볼 수 있습니다.

“ 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 ”

STEP

1. 장애감지
2. 검색 조건 설정
3. 데이터 분석
4. Operation/SQL 확인

» Access Statistics 조회 화면

SQL Text	SQL ID	SQL Plan Hash	Executions	Elapsed Time (Sec)	CPU Time (Sec)
UPDATE ORDERS SET ORDER_MODE = 'onlin...	5mdot5kt45rg3	1628223527	139,994	331,605.88	193.46
DE = 'onlin...	5mdot5kt45rg3	1628223527	125,524	144,685.00	152.76
UPDATE Full SQL Text	5mdot5kt45rg3	1628223527	115,051	137,458.44	115.40
DE = 'onlin...	5mdot5kt45rg3	1628223527	107,265	136,579.17	92.74
UPDATE Export Excel	5mdot5kt45rg3	1628223527	98,302	121,380.76	93.69
DE = 'onlin...	5mdot5kt45rg3	1628223527	74,274	118,552.61	71.88
UPDATE CopyTo Clipboard	5mdot5kt45rg3	1628223527	109,764	109,472.08	122.59
Show/Hide Columns	5mdot5kt45rg3	1628223527	109,086	104,922.07	48.53
DE = 'onlin...	5mdot5kt45rg3	1628223527	65,937	94,079.12	32.95
UPDATE Align Columns	5mdot5kt45rg3	1628223527	11,272	89,814.02	6.22
DE = 'onlin...	5mdot5kt45rg3	1628223527	30,532	84,376.79	19.08
WITH NE ECT ORD...	7hk2m2702ua0g	4269299407	23,899	89,498.66	13.80
WITH NE Multiple Sort On	79u2k84v884y7	4269299407	11,272	89,814.02	6.22
DE = 'onlin...	79u2k84v884y7	1628223527	30,532	84,376.79	19.08
DE = 'onlin...	79u2k84v884y7	1628223527	23,899	89,498.66	13.80

- ✓ 그리드를 우 클릭하여 팝업 메뉴를 호출합니다.
- ✓ 팝업 메뉴 내 Export Excel 버튼을 통해 출력된 데이터를 Excel 파일로 내려받을 수 있습니다.

MAXGAUGE



Practical Guide

Application Call Tree[Performance Analyzer]

Contents

Application Call Tree?

Application Call Tree의 활용

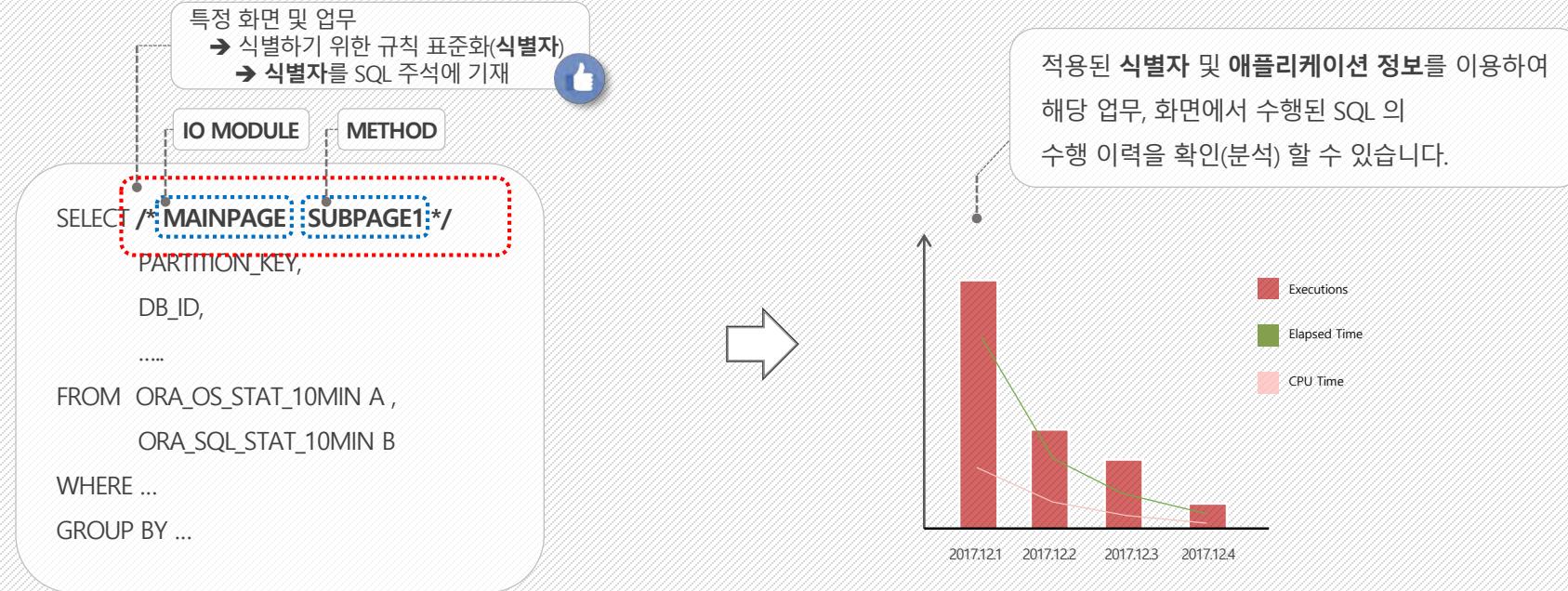
Case 1. 식별자를 사용하는 업무에서 SQL의 수정이 빈번하게 발생합니다.

해당 업무 성능에 문제가 생겼다고 의심이 될 때,

업무 단위 성능추이, SQL 이력등 관련 정보를 확인하고 싶습니다.

Application Call Tree?

Application Call Tree는 언제 쓰나요?



Application Call Tree?

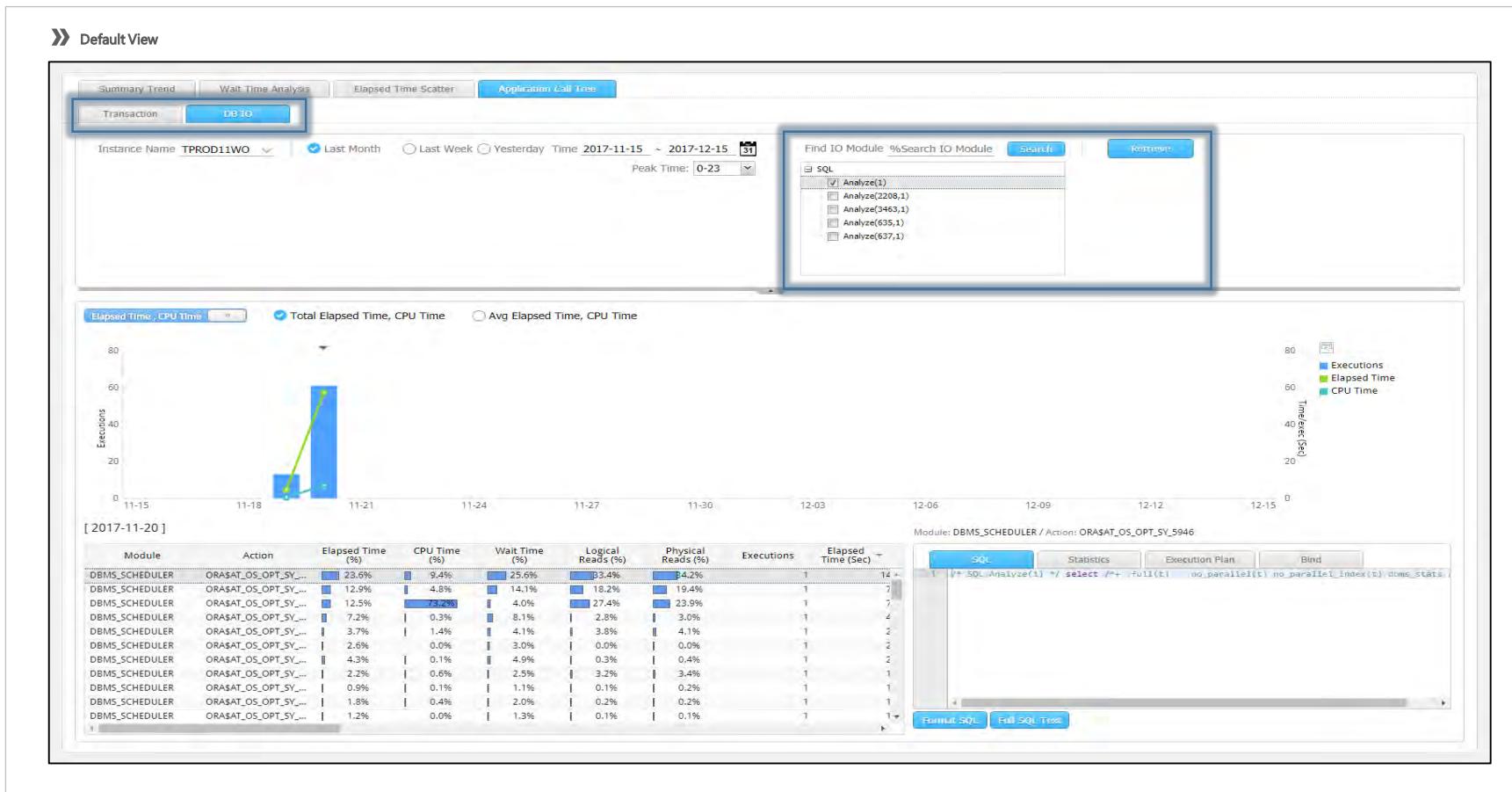
Application Call Tree

- 식별자(IO Module, Method)별 혹은 Module, Action별 **Trend & SQL** 을 확인할 수 있습니다.

TRANSACTION 탭 → **Parsing한 식별자**(IO Module, Method), 기간 별 관련 SQL & Trend 확인

DB IO 탭 → **Module, Action**, 기간 별 관련 SQL & Trend 확인

- 해당 기능은 **Performance Analyzer** ▶ **SQL Analysis** ▶ **Elapsed Time Analysis** ▶ **Application Call Tree** 경로에서 사용 가능합니다.



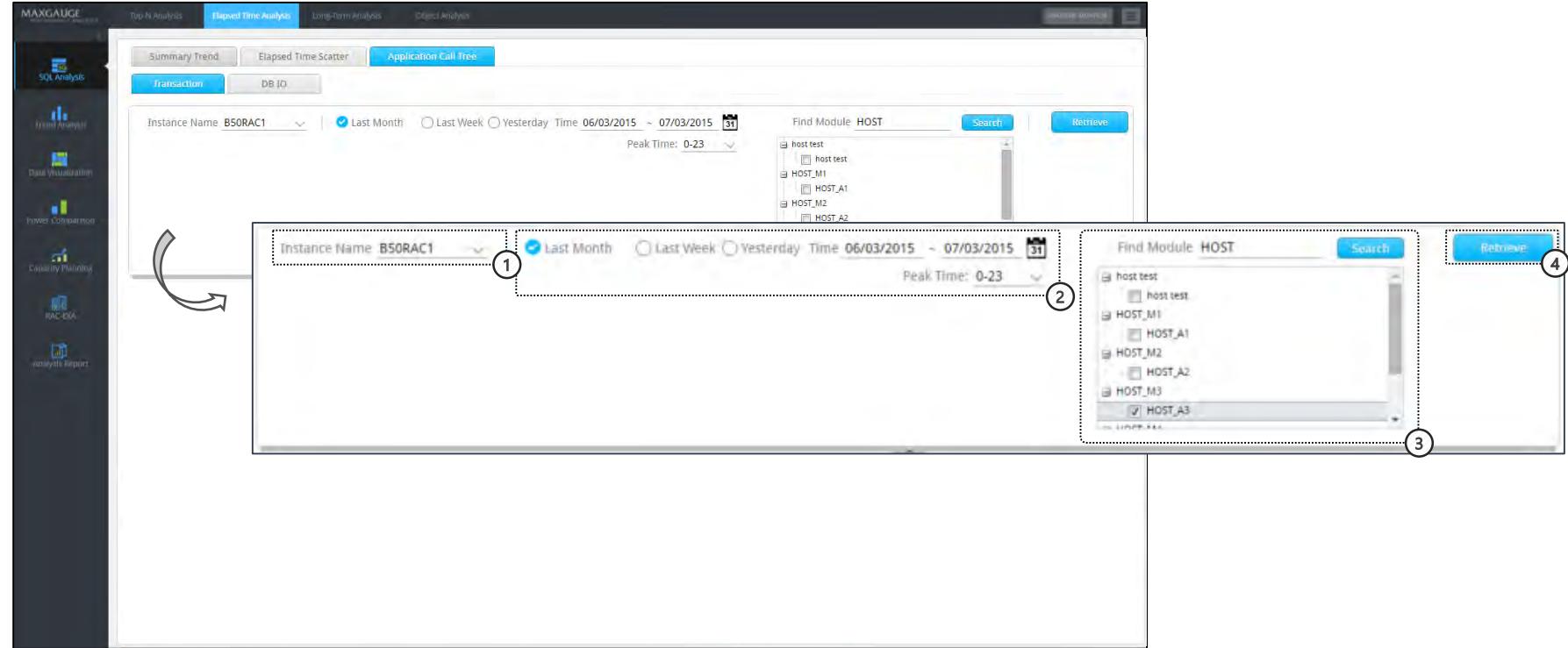
Application Call Tree의 활용

Application Call Tree의 사용 방법

Application Call Tree

- 검색 조건 설정
- 데이터 분석

» Application Call Tree 조회 화면

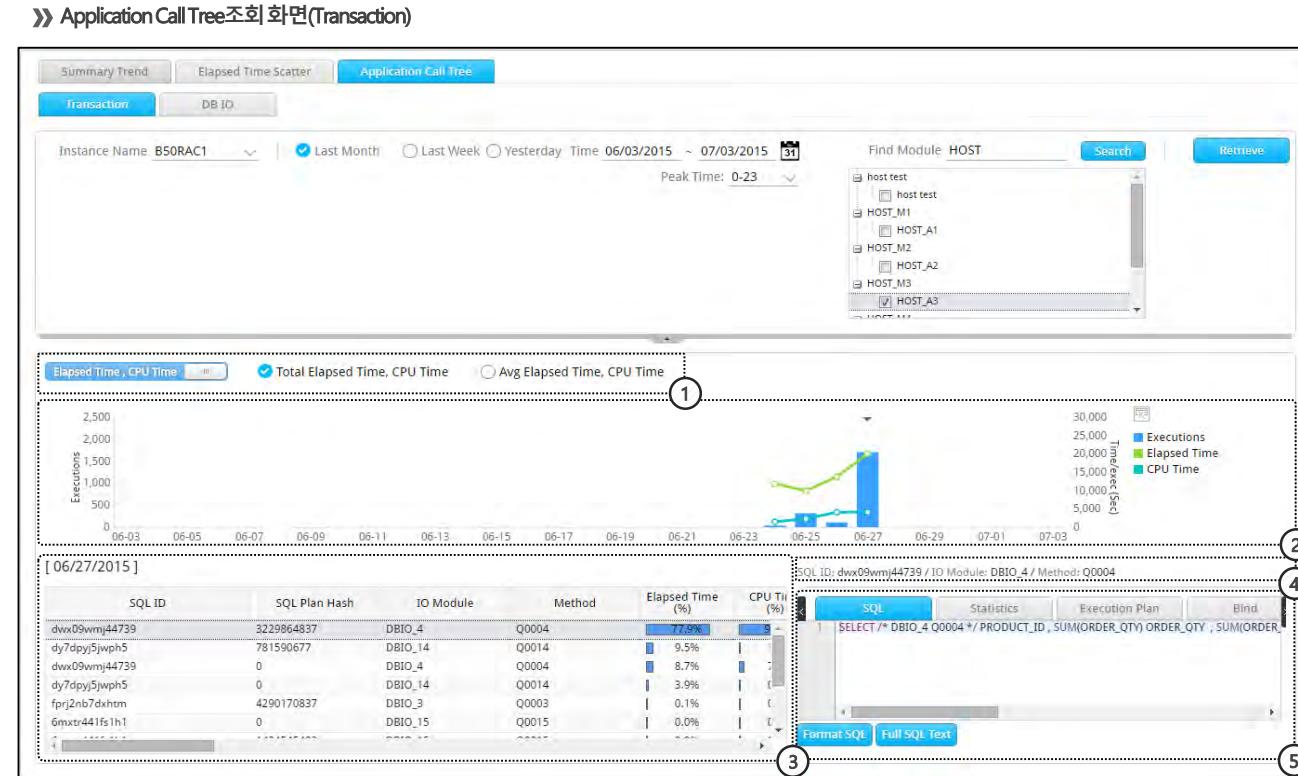


- ① 대상 Instance를 선택합니다.
- ② 분석하고자 하는 기간과 시간을 설정합니다.
- ③ Repository에 저장된 식별자(IO Module, Method) 및 Module, Action에 대해서 Tree 형식으로 검색 후 선택합니다.
- ④ 설정한 기간, Peak Time의 식별자(IO Module, Method), 또는 Module, Action 조건으로 Application Call Tree를 실행합니다.

Application Call Tree의 사용 방법

Application Call Tree

- 검색 조건 설정
- 데이터 분석



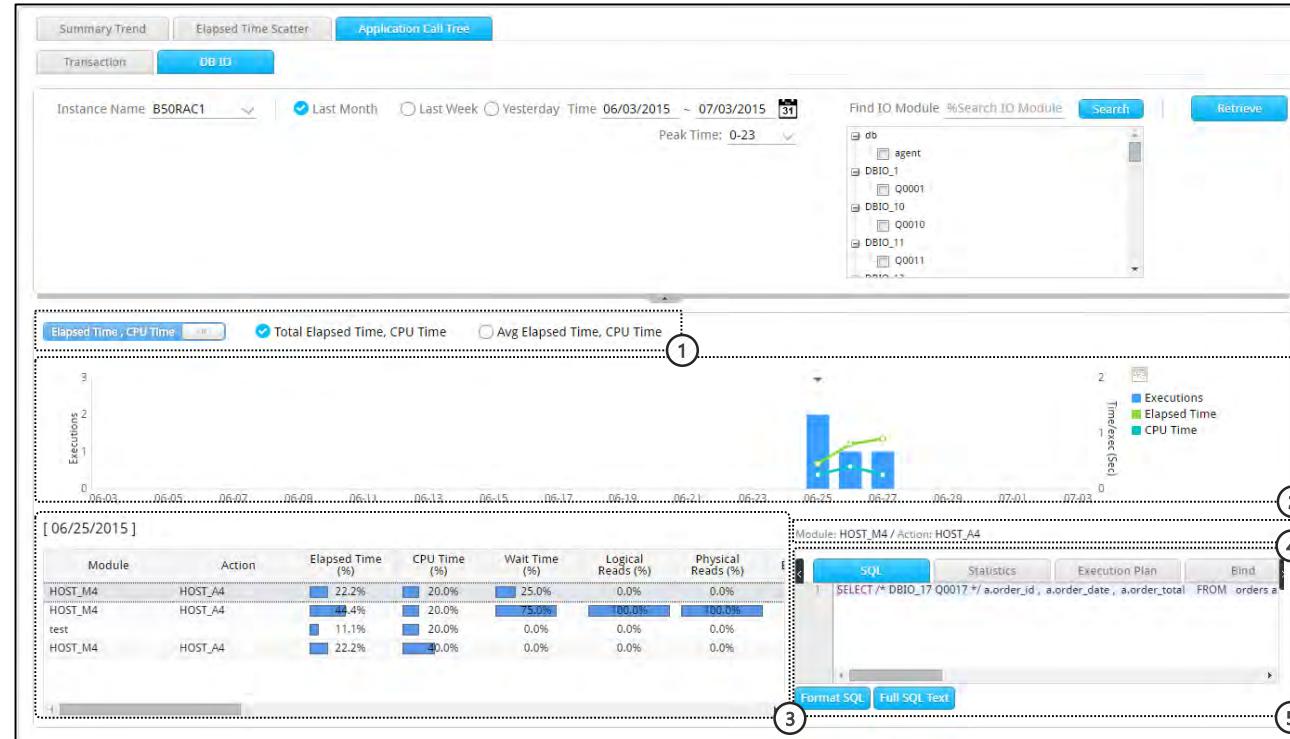
- ① Elapsed Time, CPU Time / Session Logical Reads, Physical Reads를 선택 가능하며, 각각의 Total 혹은 Avg값을 선택합니다.
- ② x축은 날짜(일), y축은 좌측 Executions, 우측 Time/exec(Block/exec)로 막대그래프와 선형그래프로 출력합니다.
- ③ 선택된 날짜의 SQL 개별 일량을 확인합니다.
- ④ 선택된 IO Module, Method, SQL ID 정보를 확인합니다.
- ⑤ 선택된 SQL에 대한 Text 및 Bind, Execution Plan 정보를 확인할 수 있습니다.

Application Call Tree의 사용 방법

Application Call Tree

- 검색 조건 설정
- 데이터 분석

» Application Call Tree 조회 화면(DBIO)



- ① Elapsed Time, CPU Time / Session Logical Reads, Physical Reads를 선택 가능하며, 각각의 Total 혹은 Avg값을 선택합니다.
- ② x축은 날짜(일), y축은 좌측 Executions, 우측 Time/exec(Block/exec)로 막대그래프와 선형그래프로 출력합니다.
- ③ 선택된 날짜의 Module 및 Action의 일량을 확인합니다.
- ④ 선택된 IO Module, Method, SQL ID 정보를 확인합니다.
- ⑤ 선택된 SQL에 대한 Text 및 Bind, Execution Plan 정보를 확인할 수 있습니다.

실전 분석 사례 Case 1.
식별자를 사용하는 업무에서 SQL 수정이 빈번하게 발생합니다.
해당 업무 성능에 문제가 생겼다고 의심이 될 때,
업무단위 성능추이, SQL 이력등 관련 정보를 확인하고 싶습니다.

STEP

1. PA – CPU 증가 원인 확인

2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인

“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”



① 특정 업무가 느려진 시점에 CPU의 수치가 증가한 것을 확인할 수 있습니다.

② 동일 시점에 IO Class Wait이 증가한 것을 확인할 수 있습니다.

✓ 업무 시스템 부하 원인 확인

PA – 특정 업무가 느려진 날짜의 CPU 증가

시점에 IO Class Wait이 증가한 것을 확인.

STEP

1. PA – CPU 증가 원인 확인

2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인

“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”



- ① 대상 Instance (TPROD11WO)를 선택합니다.
- ② 분석하고자 하는 날짜(장애 발생 시점) 및 Peak Time에 대해서 선택합니다.
- ③ 문제가 되는 업무 시스템의 식별자 중 IO Module을 검색하면 하단 박스에 트리형식으로 Method가 표시됩니다.
- ④ 트리형식으로 표시된 IO Module, Method에서 분석할 업무의 식별자를 선택합니다.
- ⑤ 위의 설정한 조건으로 Application Call Tree를 실행합니다.

✓ 시나리오

INSTANCE	: TPROD11WO
Peak Time	: 0 ~ 23
IO Module	: (null)
분석 상세 일자	: 11/09 ~ 11/20

✓ 분석 방향 및 의도

특정 업무 시스템에 부하가 발생한 시점에 해당 업무 식별자를 통하여 상세이력을 확인하고 싶습니다.

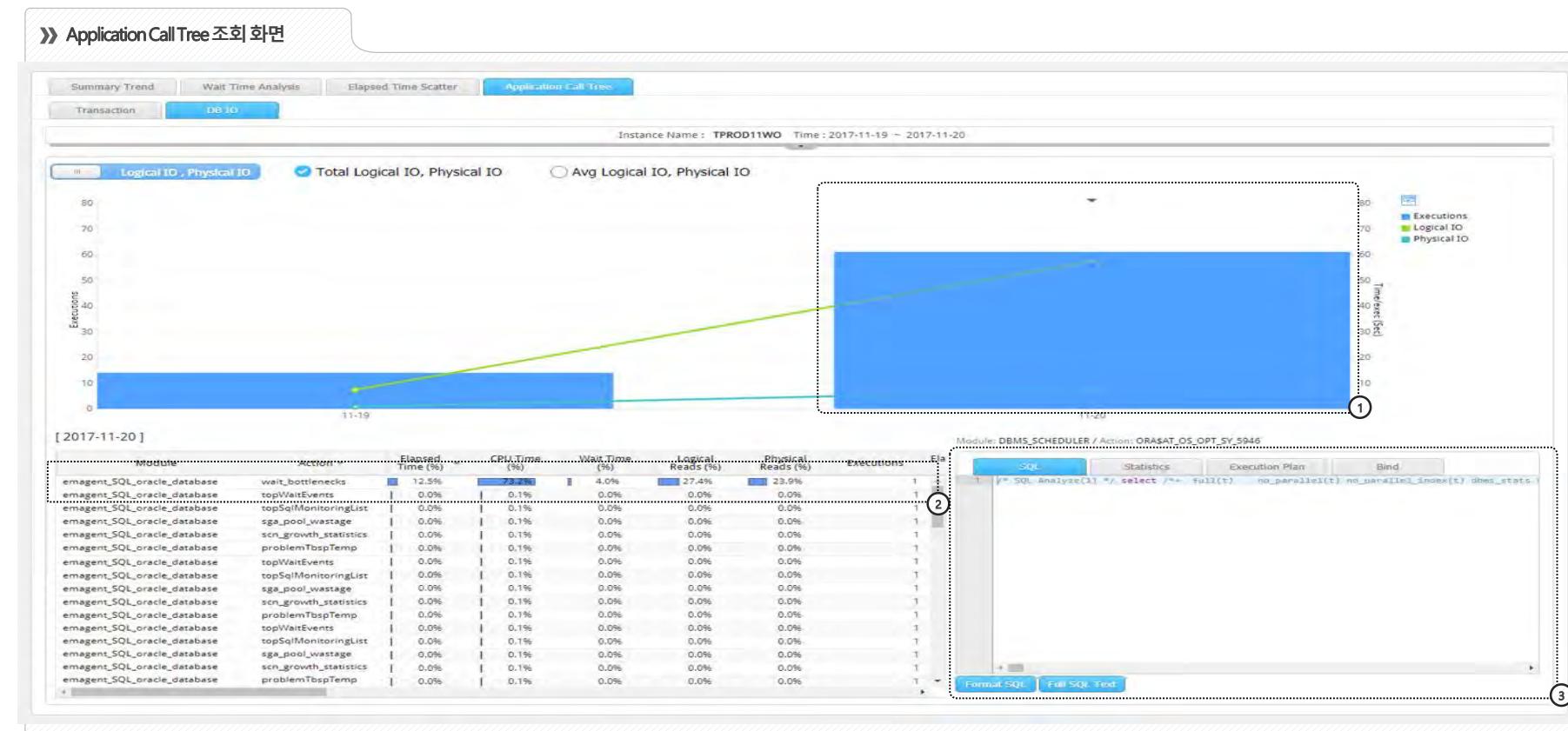
“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

STEP

1. PA – CPU 증가 원인 확인
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인



- ① 문제가 없던 날짜에 비하여 전체적으로 Executions, Logical IO, Physical IO가 증가한 것을 확인할 수 있습니다.
- ② 선택한 식별자(특정 업무)에 해당하는 SQL 목록 중에서 유독 부하가 많은 SQL을 확인할 수 있습니다.
- ③ 추출된 SQL에 대한 SQL TEXT, Statistics, Execution Plan, Bind 변수를 확인할 수 있습니다.

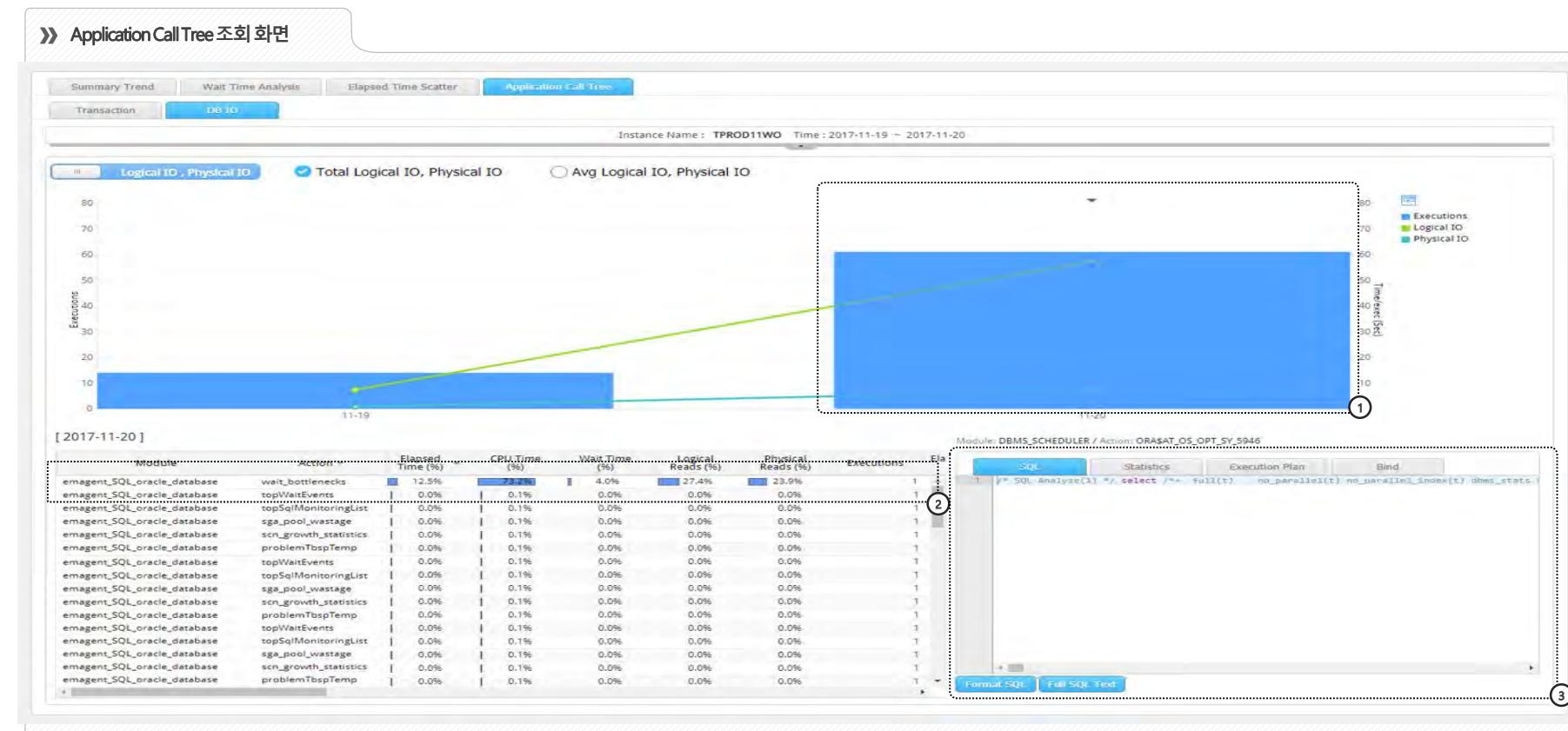
“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

STEP

- PA – CPU 증가 원인 확인
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인



- 문제가 없던 날짜에 비하여 전체적으로 Executions, Logical IO, Physical IO가 증가한 것을 확인할 수 있습니다.
- 선택한 식별자(특정 업무)에 해당하는 SQL 목록 중에서 유독 부하가 많은 SQL을 확인할 수 있습니다.
- 추출된 SQL에 대한 SQL TEXT, Statistics, Execution Plan, Bind 변수를 확인할 수 있습니다.

Application Call Tree의 활용

“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

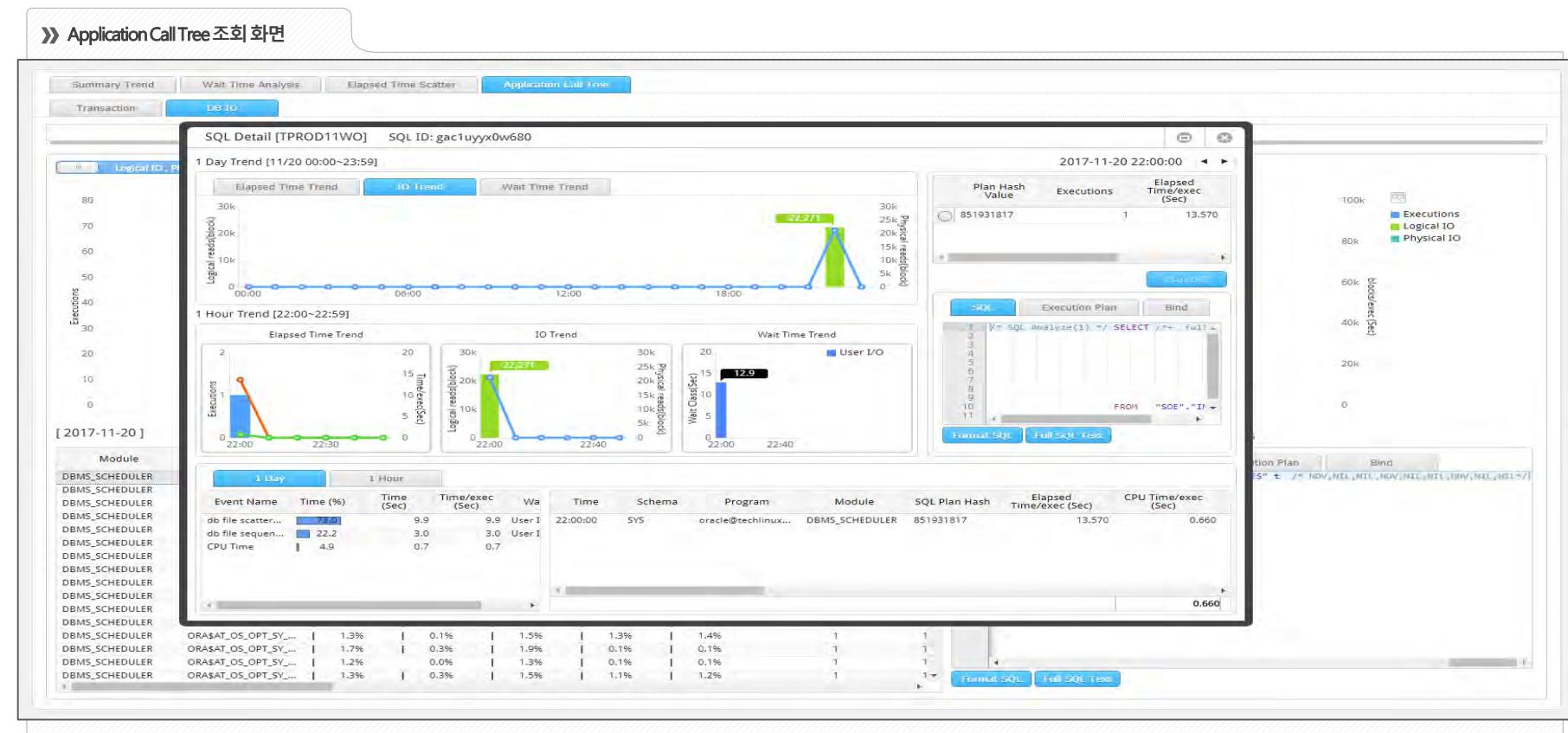
STEP

- PA – CPU 증가 원인 확인

- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인**
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인



✓ SQL Detail을 통해 선택한 날짜에 SQL 상세 정보를 확인할 수 있습니다.

✓ 'IO Trend Tab'을 선택하면 시간대별 IO 발생 Trend를 확인할 수 있습니다.

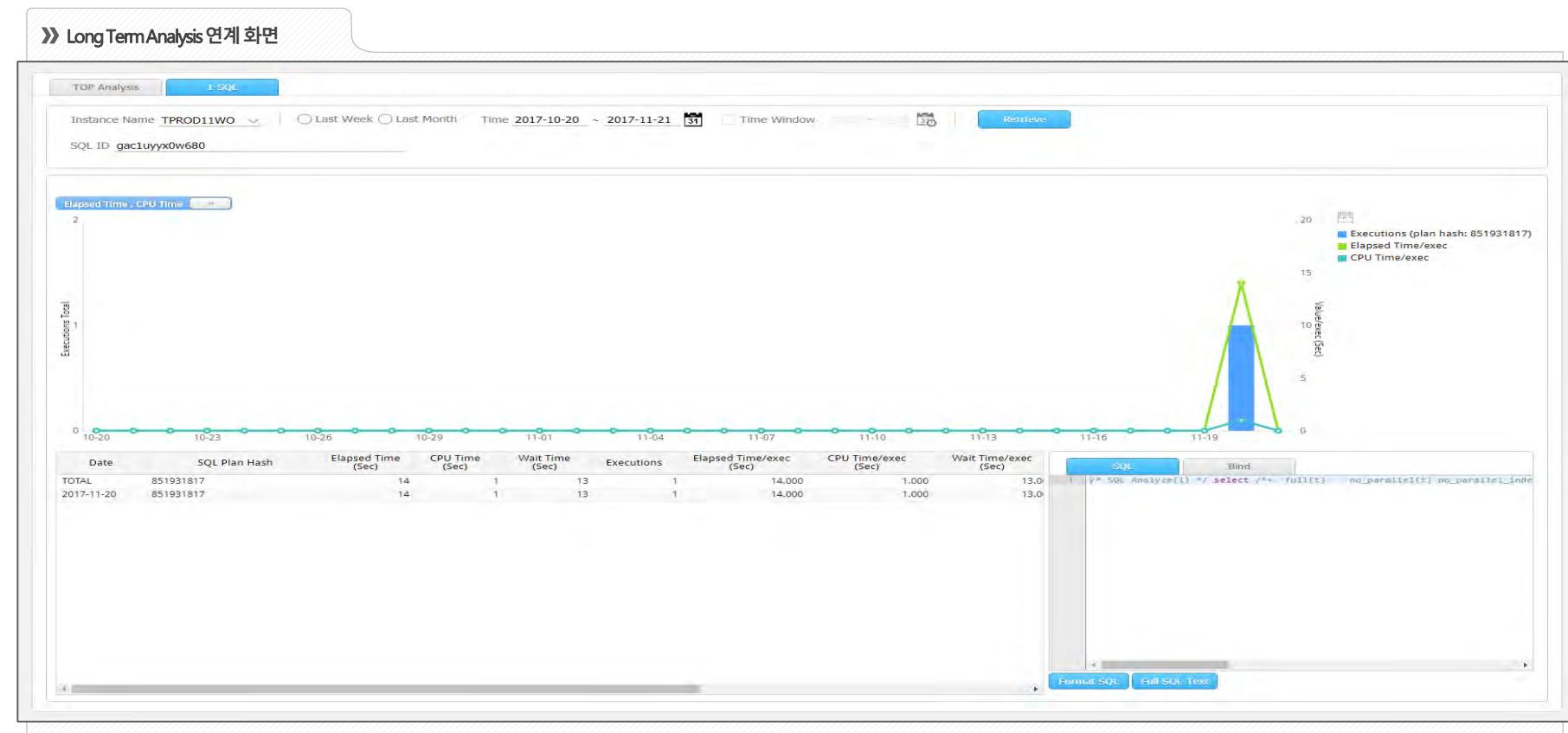
“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

STEP

1. PA – CPU 증가 원인 확인
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인



- ✓ 문제가 되는 SQL ID를 복사하여, Long-Term Analysis 탭으로 이동합니다.
- ✓ 해당 SQL은 기존에 수행되지 않은 새로운 SQL로 최적화가 되지 않았기에 SQL TUNING이 필요합니다.

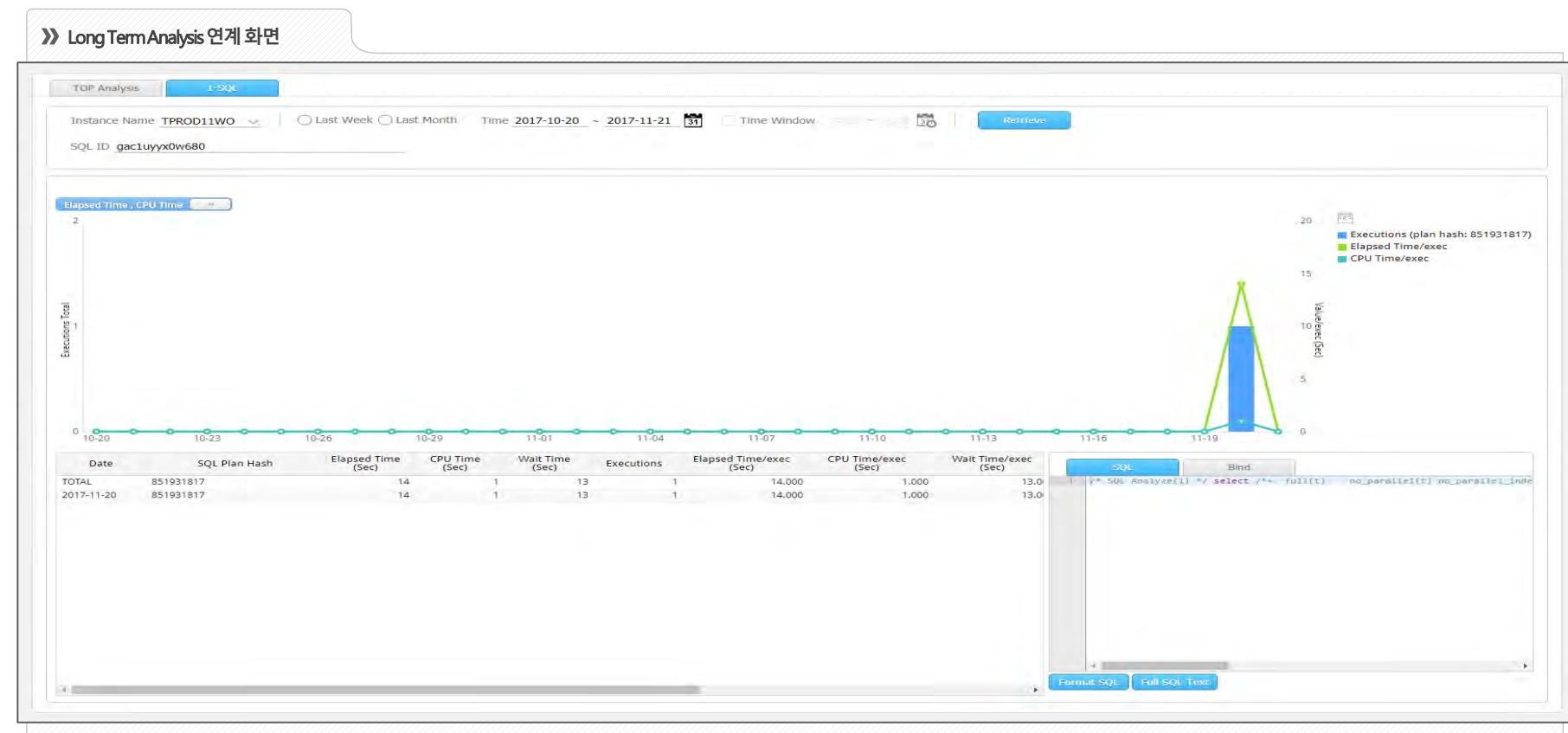
“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

STEP

1. PA – CPU 증가 원인 확인
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인



- ✓ 문제가 되는 SQL ID를 복사하여, Long-Term Analysis 탭으로 이동합니다.
- ✓ 해당 SQL은 기존에 수행되지 않은 새로운 SQL로 최적화가 되지 않았기에 SQL TUNING이 필요합니다.

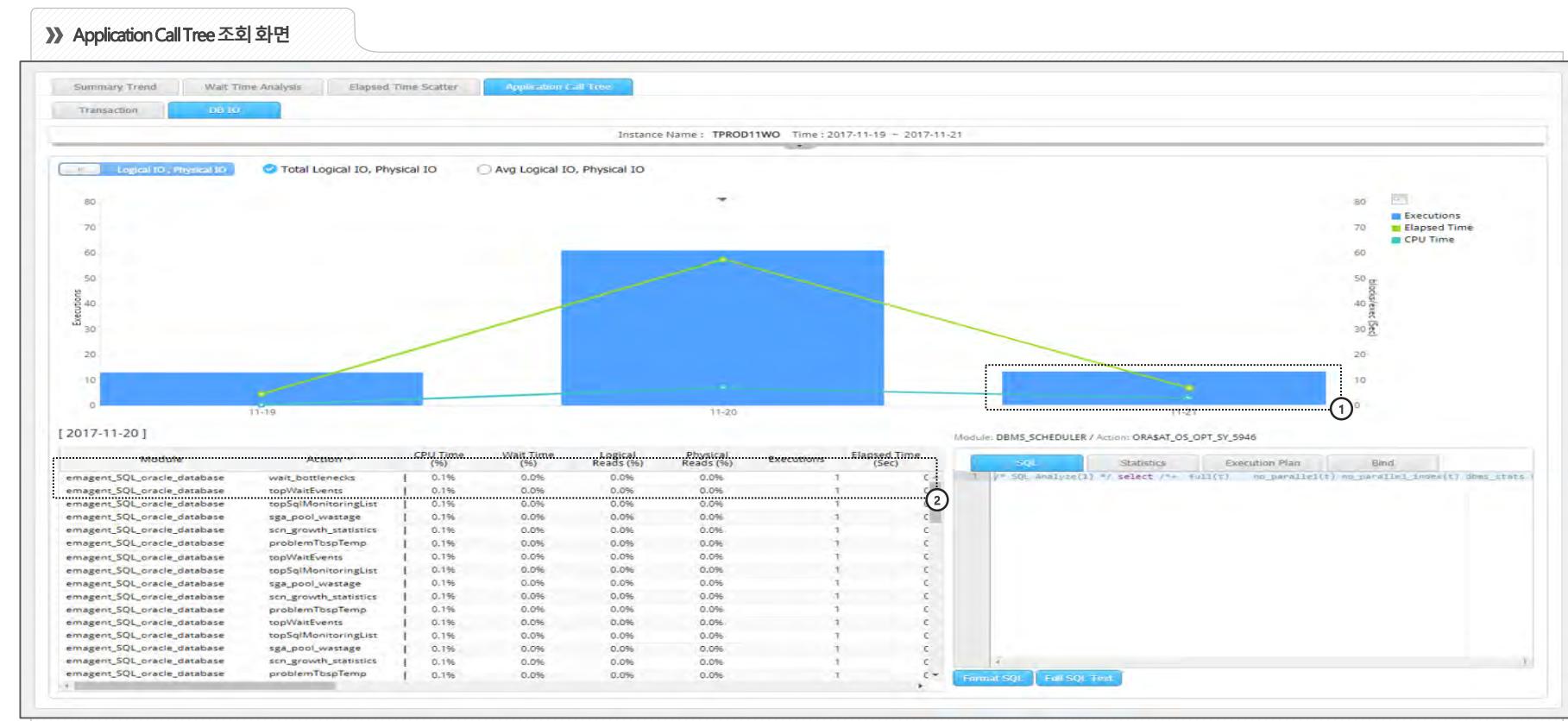
“ 특정업무의 SQL 수행내역을 확인하고 싶습니다.”

STEP

- PA – CPU 증가 원인 확인
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- SQL 시간대별 추이 확인
- 1-SQL Detail 정보 확인
- SQL TUNING 후 확인**



- ① 문제가 된 SQL에 대해서 TUNING을 통하여 최적화한 후 해당 업무의 전체적인 일량이 크게 감소한 것을 확인할 수 있습니다.
- ② 해당 SQL에 대한 성능정보 및 이력 역시 문제 시점과 비교하여 크게 감소한 것을 확인할 수 있습니다.

MAXGAUGE

Practical Guide

Elapsed Time Analysis [Performance Analyzer]

Contents

Elapsed Time Scatter?

Elapsed Time Scatter 의 활용

Case 1. 업무시간에 5초 이상 수행되는 SQL을 추출하고 싶어요.

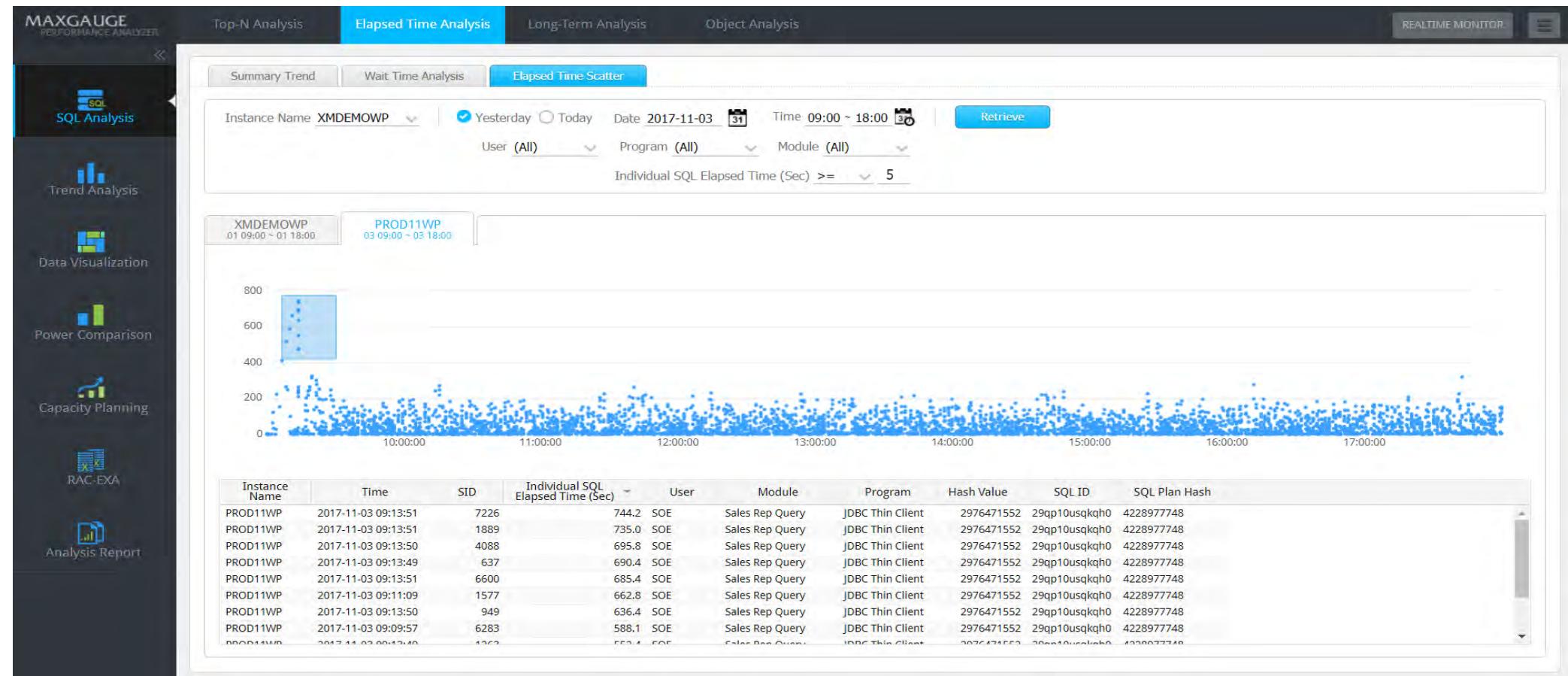
그리고 SQL이 느린 이유를 알고 싶어요

Elapsed Time Scatter?

Elapse Time Scatter?

Elapse Time Scatter

- 특정 날짜와 시간에 대한 SQL 수행 분포를 확인할 수 있다.
- 특정 Instance 를 보거나 Schema, Program, Module 을 필터링해서 SQL 수행 시간에 대한 분포도를 볼 수 있다.
- 개별 SQL수행시간에 따른 SQL 분포도를 확인할 수 있다.
- 해당 기능은 Performance Analyzer ▶ SQL Analysis ▶ Elapsed Time Analysis ▶ Elapsed Time Scatter 경로를 통해 사용할 수 있다.



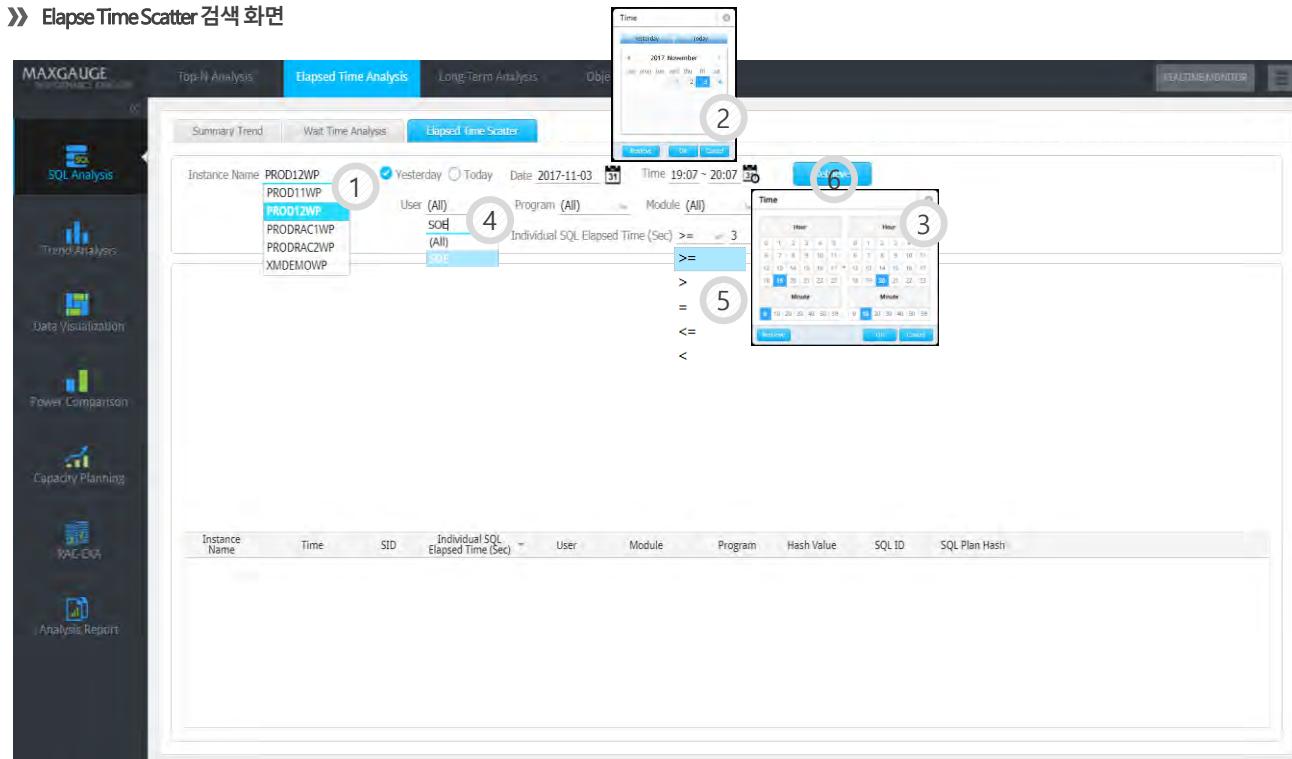
Elapse Time Scatter의 활용

Elapse Time Scatter의 사용 방법

Elapse Time Scatter

- 검색 조건 설정
- 데이터 분석

» Elapse Time Scatter 검색 화면



- 분석대상 Instance를 선택합니다.
- 분석 일자를 어제, 오늘, 혹은 분석을 원하는 일자를 직접 입력하거나 달력 UI를 이용하여 선택합니다.
- 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- Combo box를 이용하여 해당 Instance 의 User, Program ,Module 을 선택할 수 있고 2개 이상의 schema를 선택할 수도 있습니다.
- 개별 SQL의 응답시간 검색 기준을 설정합니다.
- 위의 설정한 조건으로 Elapse scatter를 실행합니다.

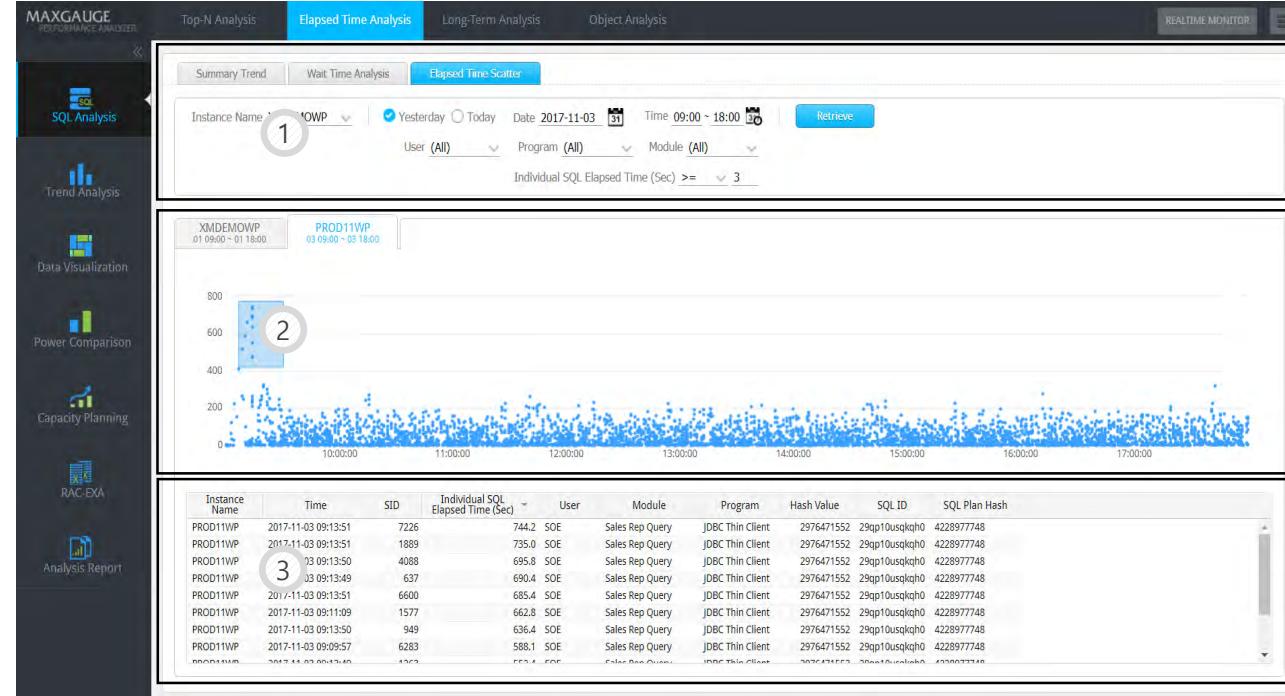
Elapse Time Scatter의 활용

Elapse Time Scatter의 사용 방법

Elapse Time Scatter

- 검색 조건 설정
- 데이터 분석

» Elapse Time Scatter 조회화면



- Search Condition 창으로 검색조건을 입력하기 위한 부분입니다.
- Elapsed Time에 대한 Scatter Chart입니다. 특정 범위만을 선택하고 싶을 때에는 drag 하여 선택 선택합니다.
- Drag 하여 현재 선택한 범위에 해당하는 SQL 정보를 표시하여 줍니다.

실전 분석 사례 Case 1.
5초이상 수행된 SQL중 수행시간이
긴 SQL을 분석하고 싶어요

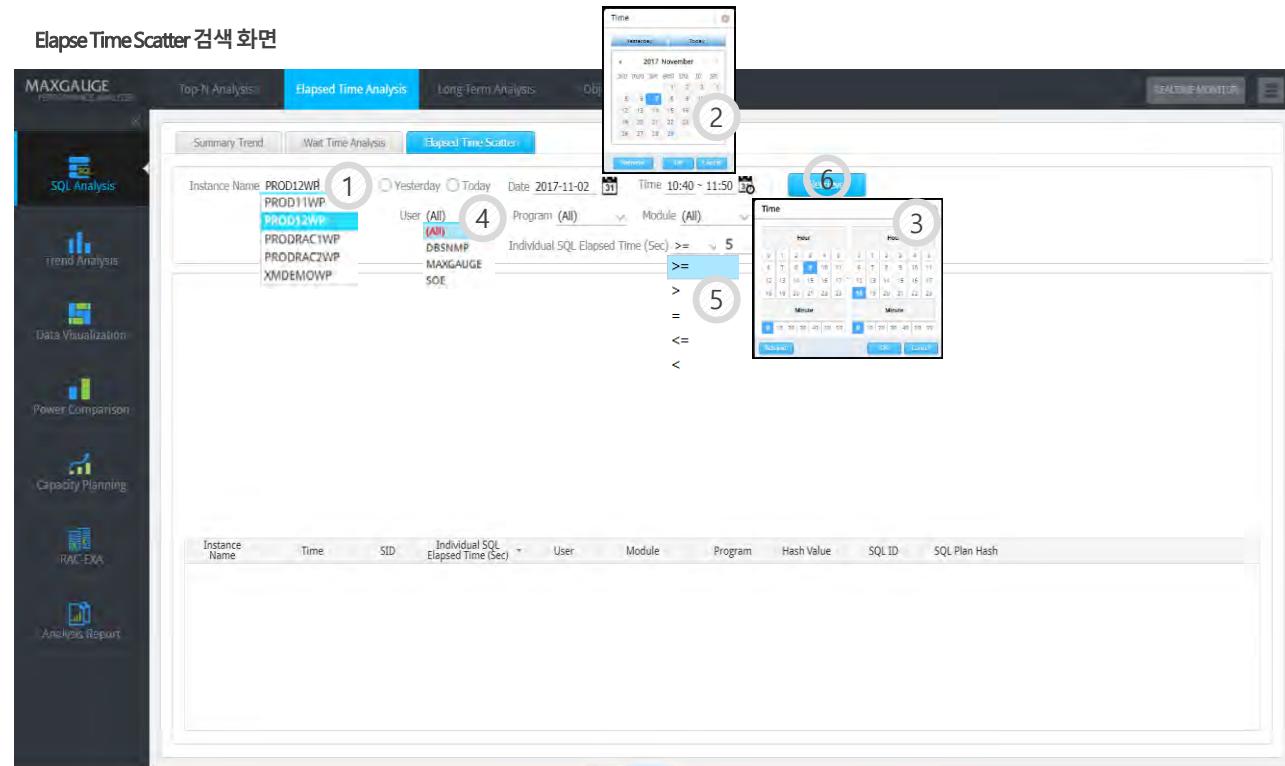
“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - └ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - └ SQL Detail연계
- 해당 SQL Plan 정보확인
 - └ SQL Plan Diff연계



1. 분석대상 Instance (PROD12WP)를 선택합니다.
2. 분석 일자를 원하는 일자인 2017년 11월 7일을 선택합니다.
3. 상세 시간 설정에는 업무시간인 09:00 ~ 18:00 시를 선택합니다.
4. Combo box를 이용하여 해당 Instance 의 User, Program, Module 을 선택할 수 있고 2개 이상의 schema를 선택할 수도 있습니다. 현재는 모든 SQL이 대상이므로 (ALL , ALL , ALL)로 선택하도록 합니다.
5. 5초 이상 걸리는 SQL을 찾는 것이 목적이므로 SQL의 응답시간 기준을 (≥ 5)로 설정합니다.
6. 위의 설정한 조건으로 Elapse scatter를 실행합니다.

✓ 시나리오

INSTANCE	: PROD12WP
SCHEMA	: ALL
PROGRAM	: ALL
MODULE	: ALL
일자	: 2017년 11월 7일
시간	: 09:00~ 18:00
개별 SQL 응답시간	: ≥ 5

✓ 목표

특정날짜에 수행된 SQL수행 분포도를 통해
특정 SQL성능과 원인 파악

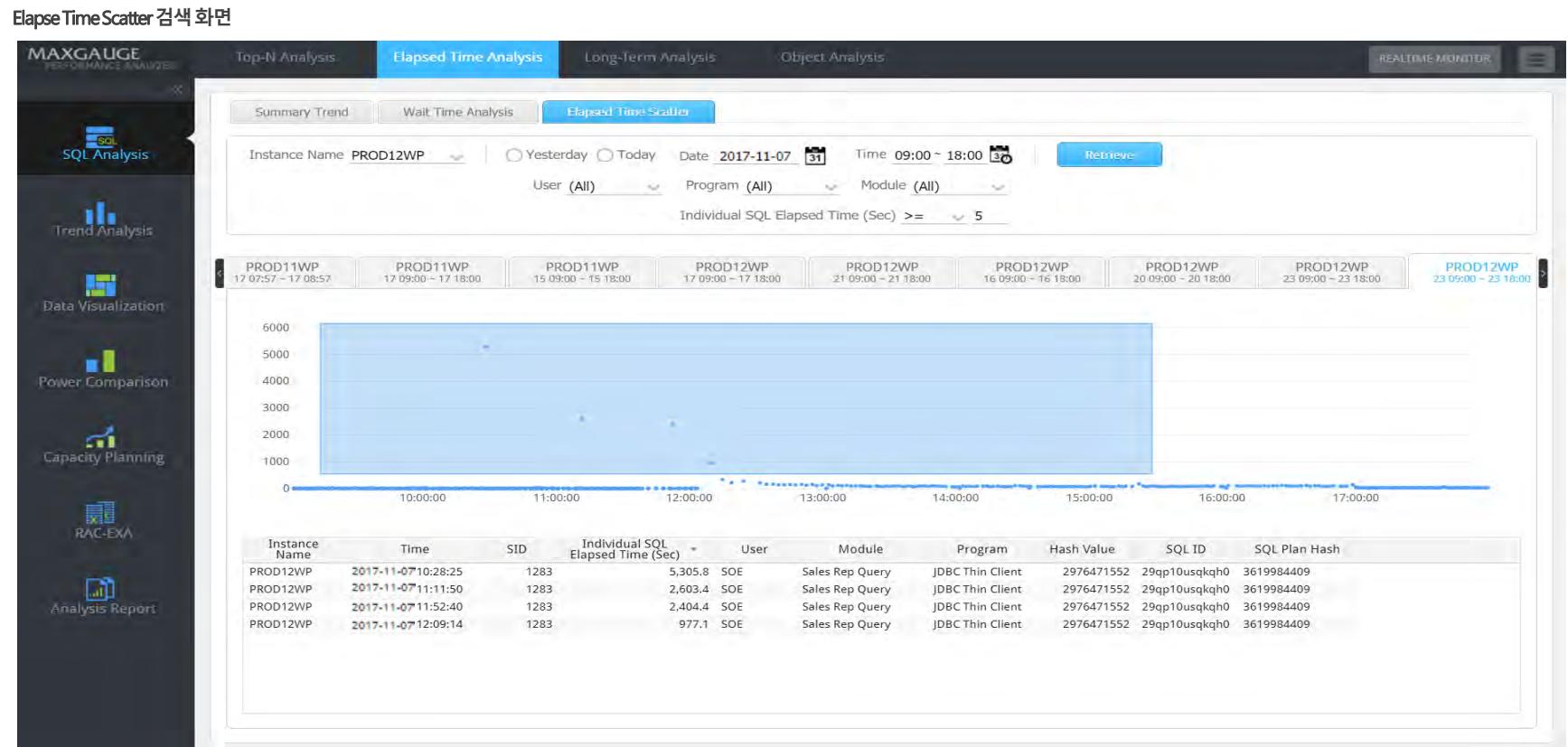
“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - └ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - └ SQL Detail연계
- 해당 SQL Plan 정보확인
 - └ SQL Plan Diff연계



- ✓ 대부분의 SQL의 Elapsed time 이 5초 미만으로 수행되었지만, 특정 SQL은 Elapsed time 이 3000초 이상에서 점이 찍힌 걸로 보아 매우 장시간 수행되는 것을 확인할 수 있습니다. (점의 빈도로 느린 SQL의 분포를 한눈에 파악 가능합니다.)
- ✓ 느린 SQL의 리스트를 확인하기 위해서, 점들이 포함되도록 드래그 하여 선택합니다.
- ✓ 하단의 그리드는 선택된 박스 안에 해당 하는 SQL의 정보를 가지고 있습니다 (Time, SID, Elapsed Time, SQL ID, User, Module, Program 등)

“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

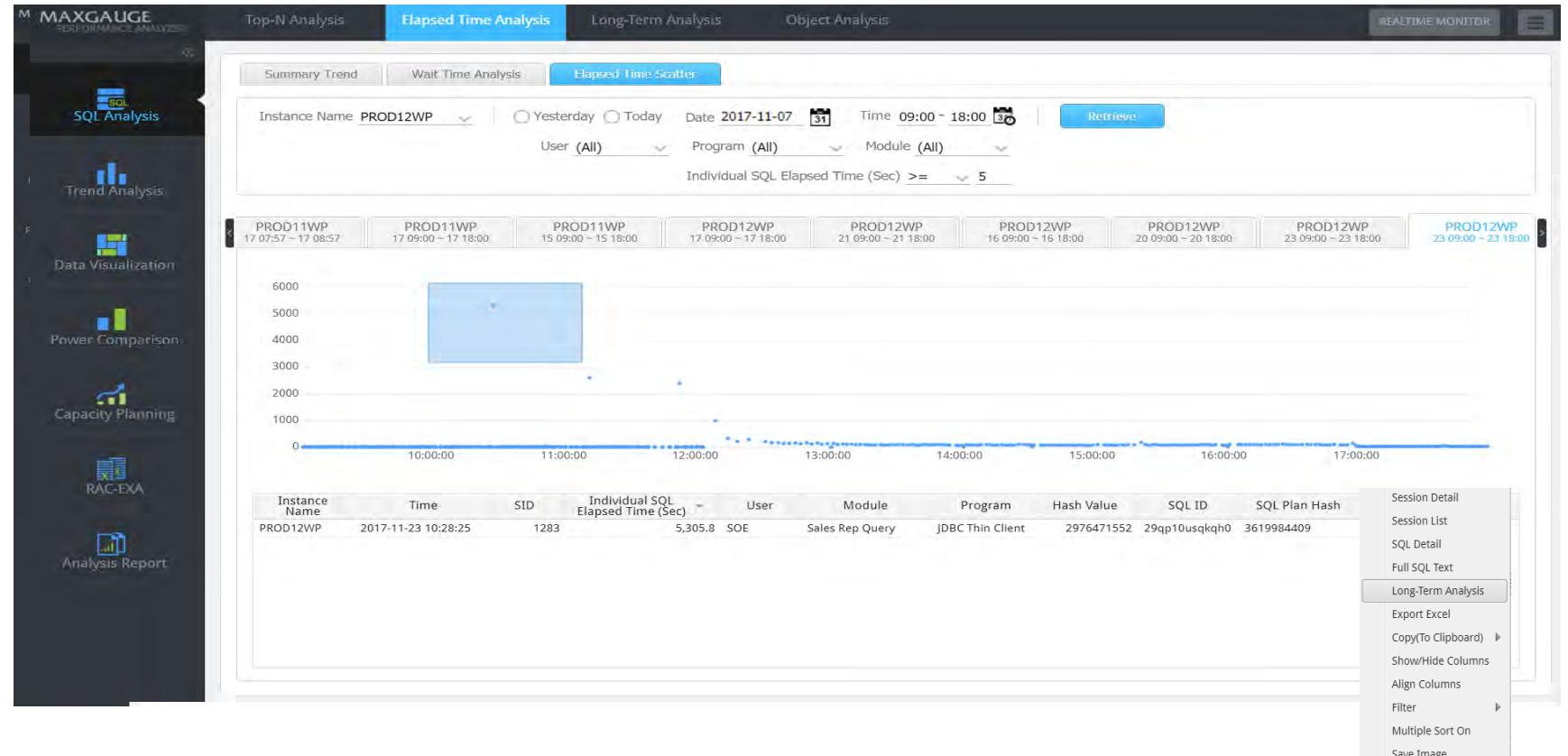
STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - └ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - └ SQL Detail연계
- 해당 SQL Plan 정보확인
 - └ SQL Plan Diff연계

Elapsed Time Scatter 검색 화면



✓ Elapsed Time 시간이 5,305.8 초인 SQL의 수행이력을 확인해보기 위해 마우스 우클릭 후 Long Term Analysis 을 클릭합니다.

“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

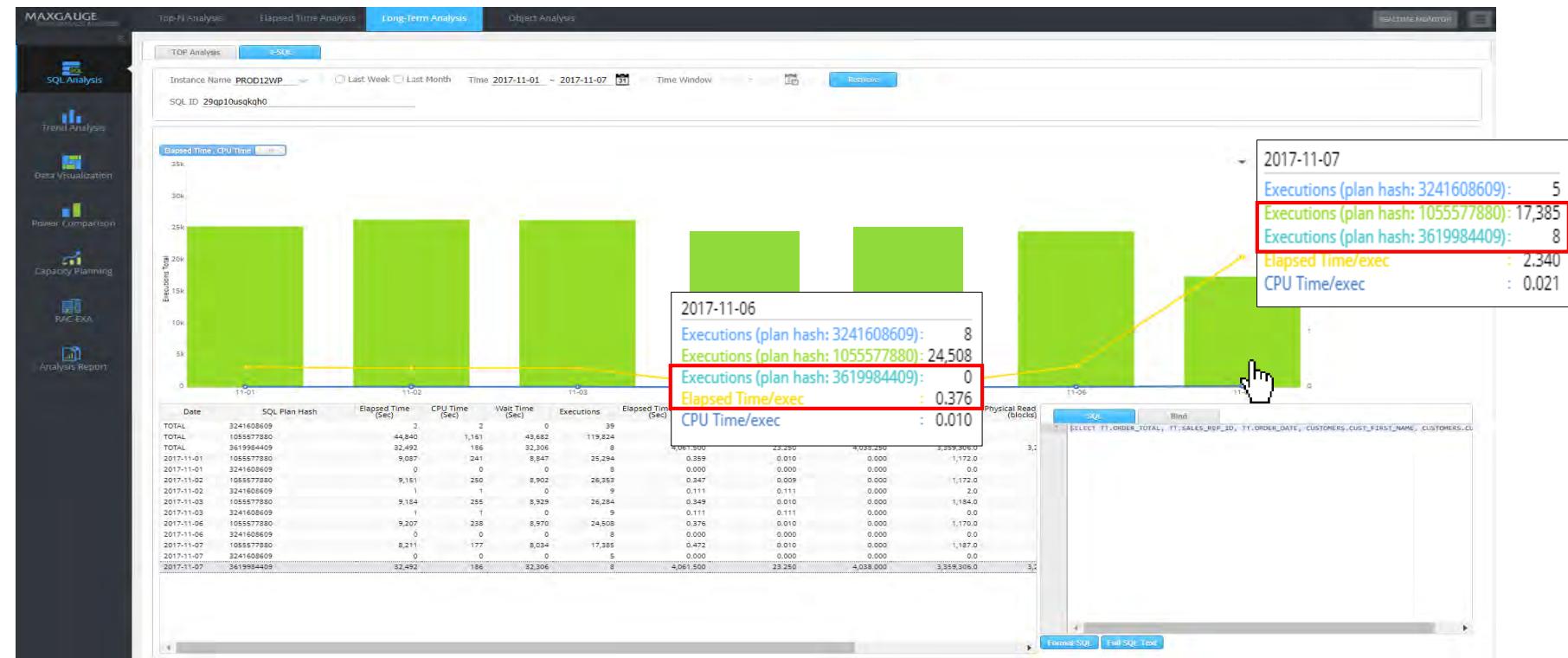
STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - └ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - └ SQL Detail연계
- 해당 SQL Plan 정보확인
 - └ SQL Plan Diff연계

Long Term Analysis 연계화면



- ✓ Long Term Analysis를 통해 확인해 보면, 1회 수행 시간은 0.37초입니다. 즉 앞에서의 5,300초란 결과는 Plan이 변경으로 인해 성능이 나빠졌을 확률이 높습니다.
- ✓ SQL의 일 평균 수행횟수는 약 25만번으로 빈번하게 수행되는 SQL입니다. 하지만 11월 6일과 11월 7일의 1회당 수행시간을 비교해 보면, 0.37초에서 2.34초로 증가하였습니다.
- ✓ 둘 차이를 비교해 보면, 11월 6일에는 Plan Hash Value가 3619984409로 수행되는 경우가 없었는데, 7일에는 8회 수행 된 점이 확인 됩니다. 즉 Plan Hash Value 3619984409의 느린 수행시간 때문에 평균 수행 시간까지 높아 진 것이 아닌지 의심이 됩니다.
- ✓ 그럼 SQL Plan Hash 별 일량 및 플랜을 확인하기 위해 막대를 클릭하고 SQL Detail로 이동합니다.

“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

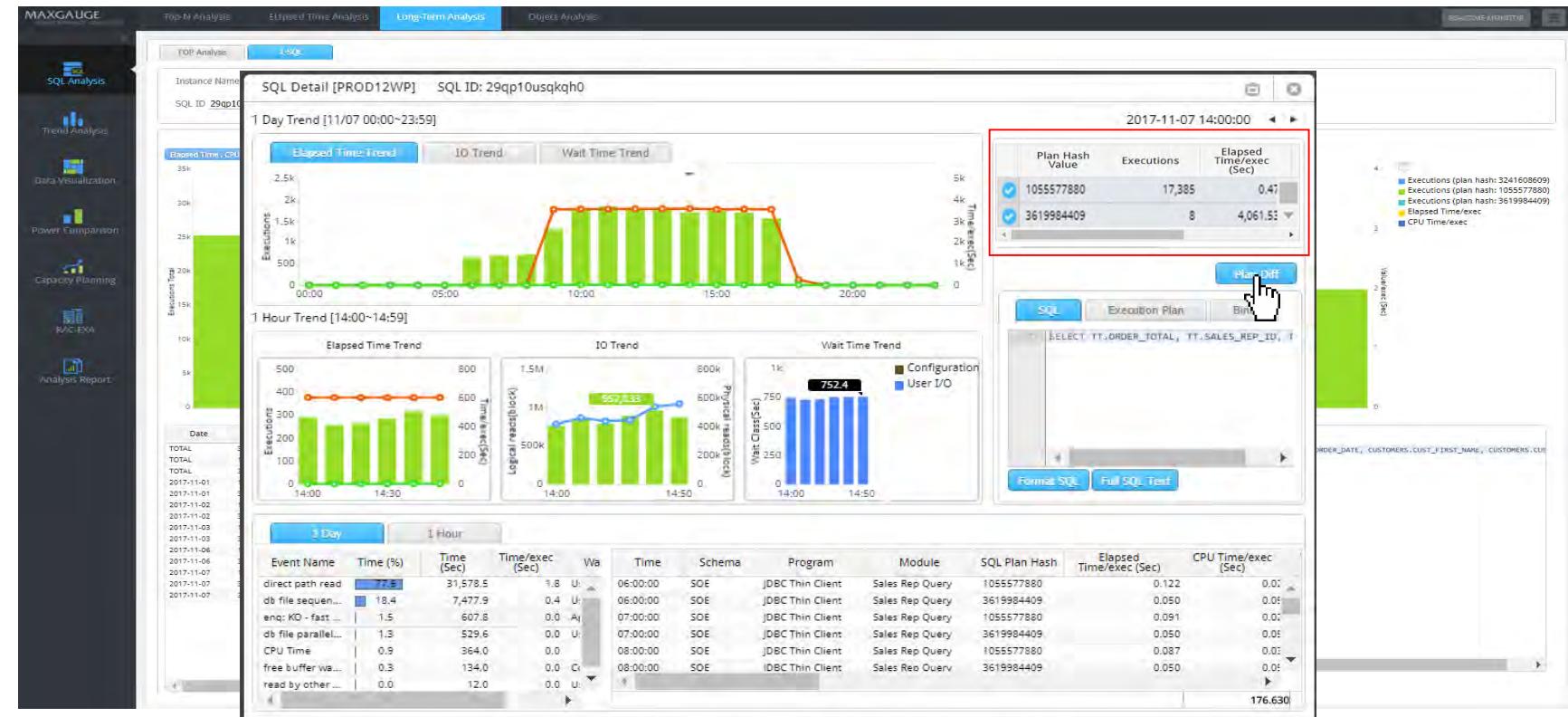
STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - ↳ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - ↳ SQL Detail연계
- 해당 SQL Plan 정보확인
 - ↳ SQL Plan Diff연계

SQL Detail연계화면



- ✓ SQL Detail은 선택한 날짜에 대한 SQL의 일량 및 수행횟수등을 추이 그래프로 보여주며, 각 Plan Hash Value별로 일량 및 수행시간을 확인 할 수 있습니다.
- ✓ 우측상단 박스를 보면 Plan hash Value별 수행시간과 I/O를 확인할 수 있습니다.
- ✓ Plan Hash Value 1055577880은 1회 수행당 약 0.47초지만 3619984409은 4061초가 걸립니다. 즉 3619984409가 악성 Plan이라 예측할 수 있습니다.
- ✓ 두 플랜의 차이와 어떤 부분이 문제를 일으켰는지 파악하기 위해서, 두 Plan Hash Value를 선택 후 Plan Diff 버튼을 눌러 플랜을 비교해 보겠습니다.

“ 5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요 ”

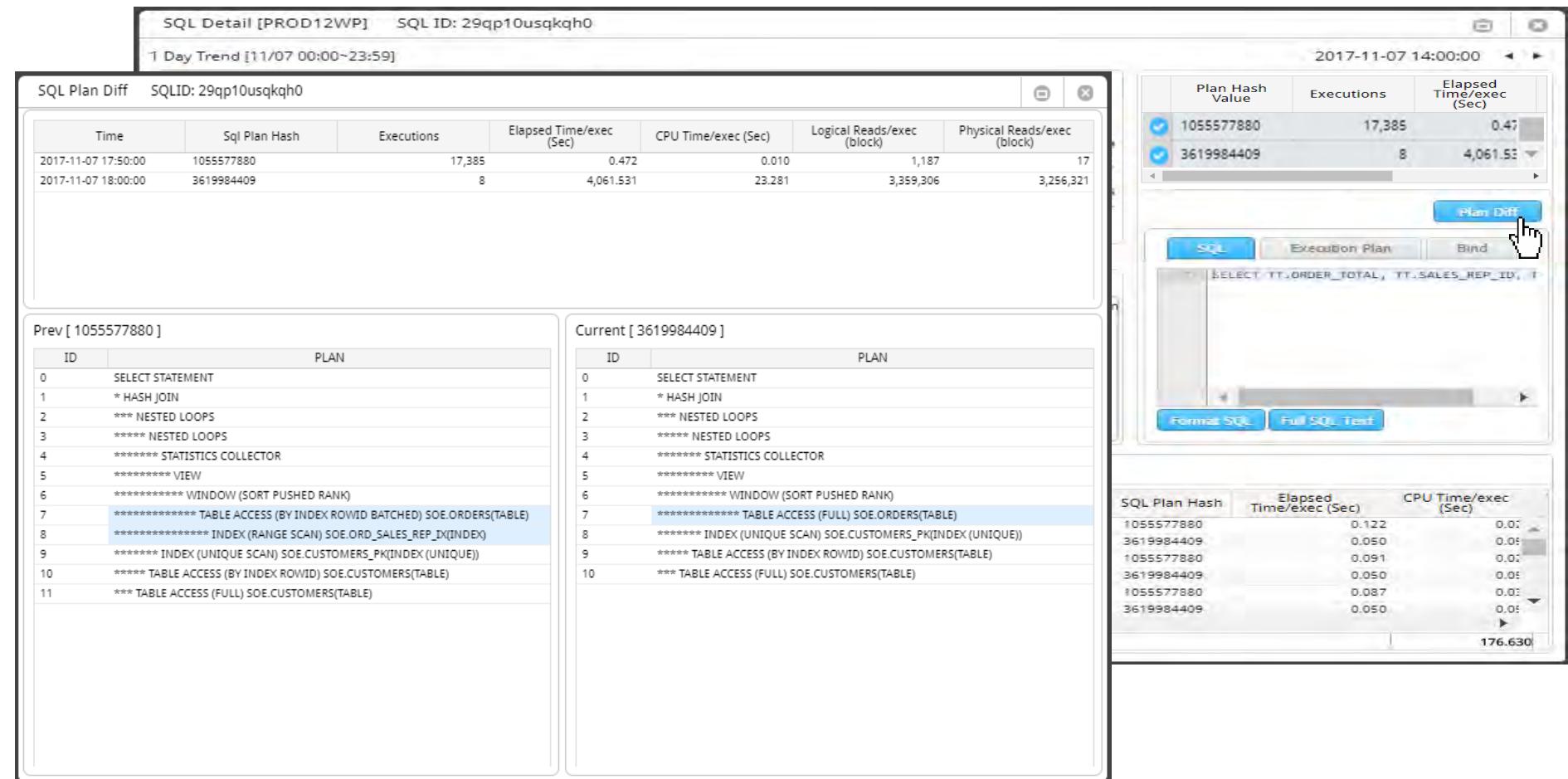
STEP

1. 검색 조건 설정

2. 데이터 분석

- 분석을 원하는 SQL 선택
- 해당 SQL 수행 이력 확인
 - └ Long Term Analysis
- 해당 SQL 상세 정보 확인
 - └ SQL Detail연계
- 해당 SQL Plan 정보확인
 - └ SQL Plan Diff연계

SQL Detail연계화면



✓ Plan Diff는 두 플랜간의 차이를 하이라이트 처리하여, 어느 부분이 변경 되었는지 확인이 매우 용이합니다. 두 플랜은 SOE_ORDERS 테이블을 읽을 때

Index Scan 하는가 Full Table Scan을 하는 가의 차가 있었으며, 기존 Index Scan이 Full Scan으로 변경 되면서 속도가 느려졌음을 확인할 수 있었습니다.

MAXGAUGE



Practical Guide

Long-Term Analysis [Performance Analyzer]

Contents

Long-Term Analysis?

Long-Term Analysis의 활용

Case 1. 특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요!

Case 2. SQL 개선 후, 상태를 체크하고 싶어요!

Long-Term Analysis ?

Long-Term Analysis?

Long-Term Analysis 은 언제 쓰나요?



MaxGauge

case 1.

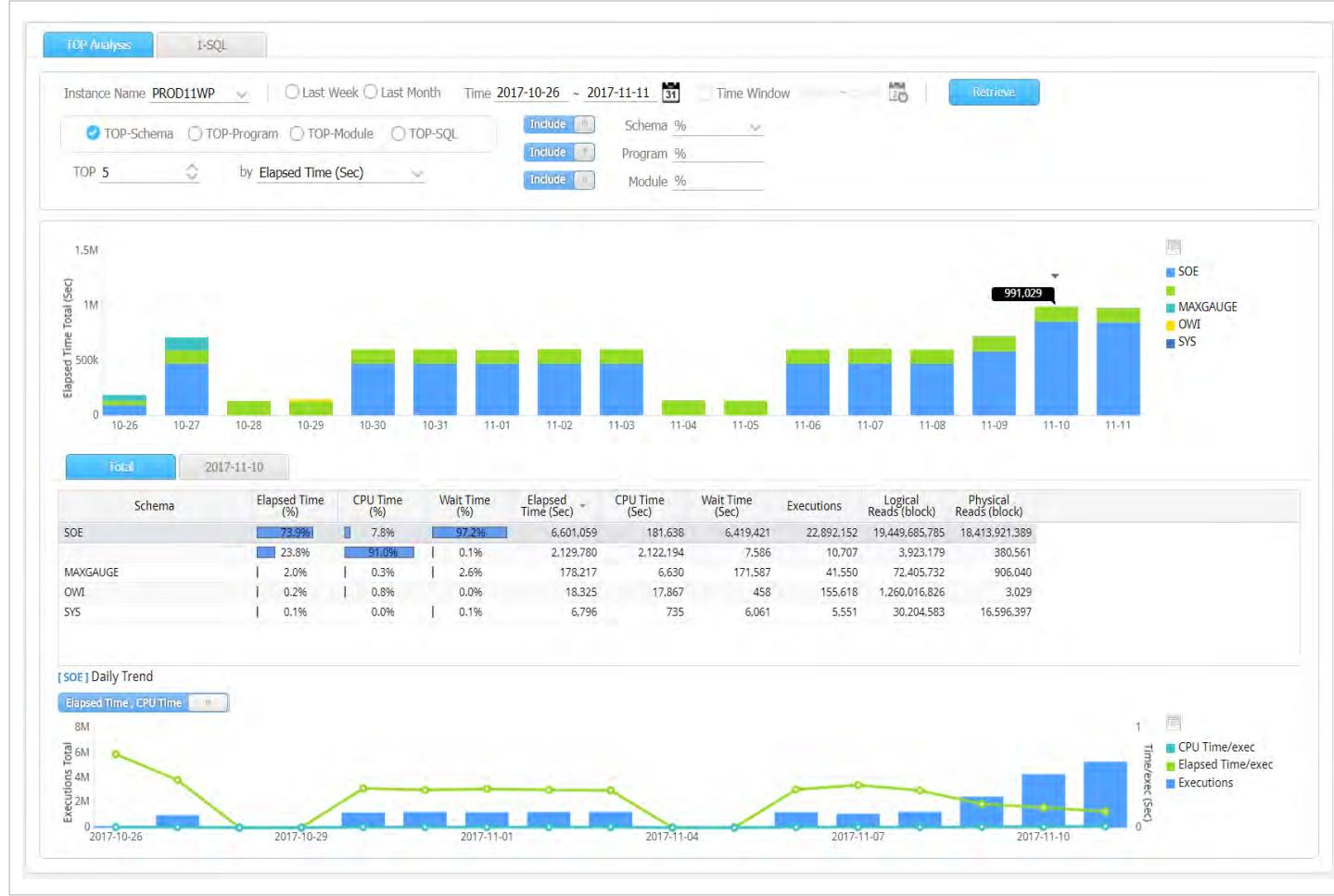
특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요.
TOP SQL을 분석해 일별 시간대별 부하 쿼리를 찾고 싶습니다.

case 2.

SQL 개선 후, 상태를 체크하고 싶어요!
특정 SQL 개선 후 성능을 확인하고 싶습니다.

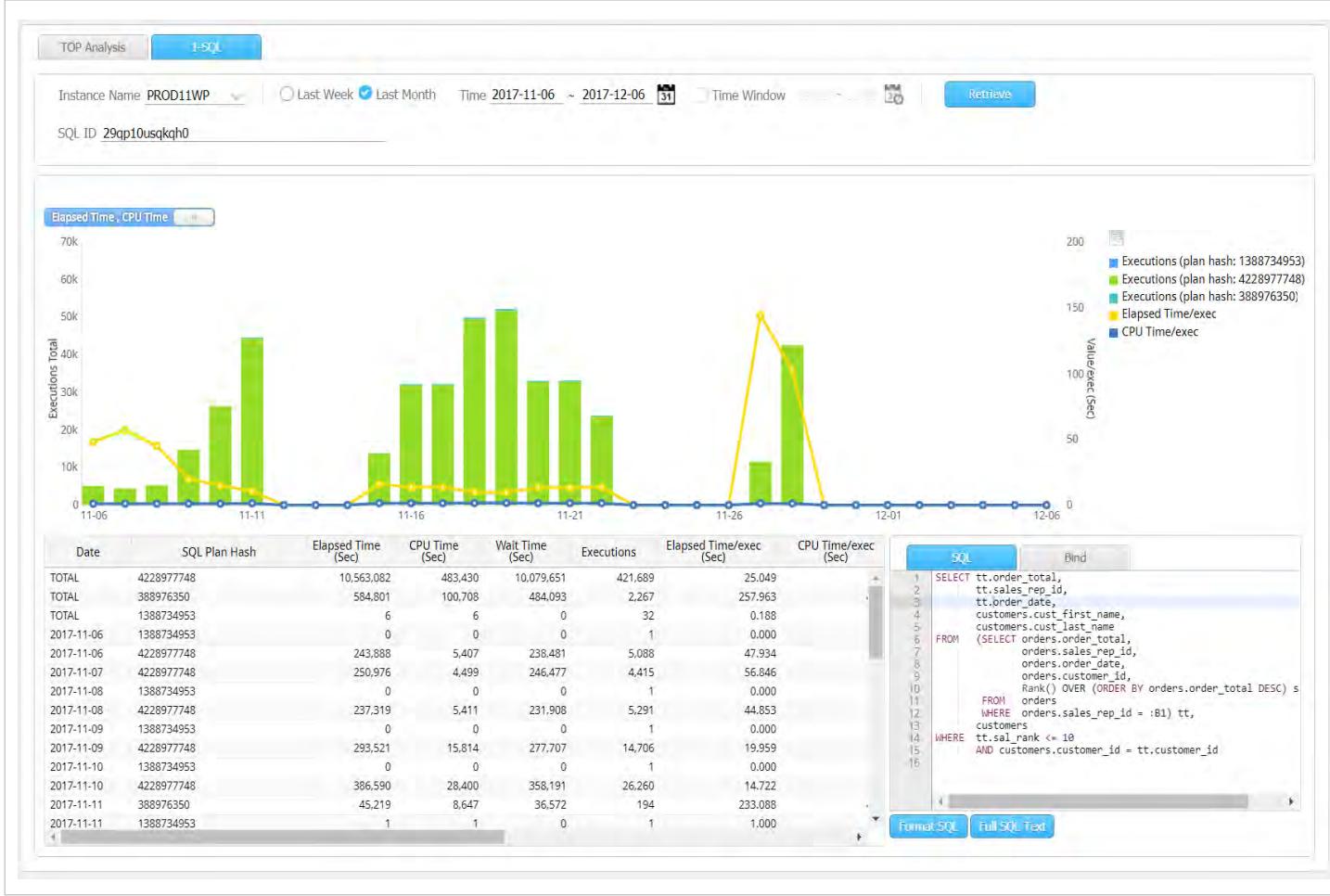
Long-Term Analysis?

Long-Term Analysis – TOP Analysis



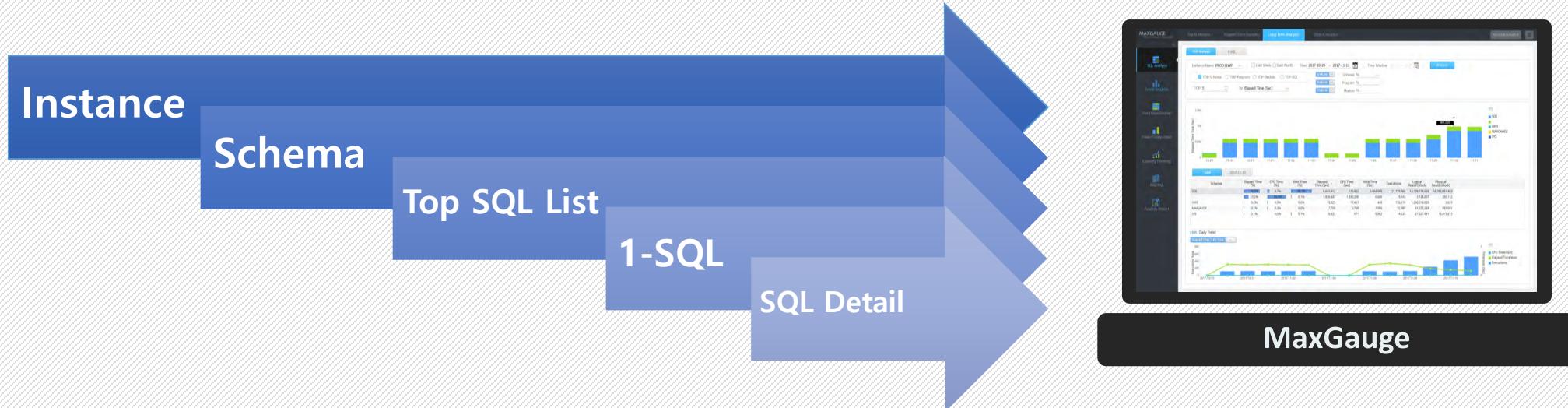
Long-Term Analysis?

Long-Term Analysis – 1-SQL



- Performance Analyzer > SQL Analysis 탭에 위치한다.
- Long-Term Analysis 2개의 기능 중 1개이다.
- 1-SQL기능은 특정 SQL에 대하여 장기간 추이 분석 기능을 제공한다.
- 일별 수행 횟수, 수행 시간, I/O량을 확인할 수 있고 SQL Text, Bind 정보 확인이 가능하다.
- 기간은 Last Week / Last Month/ 날짜 선택으로 조회 가능하며, Time Window 기능으로 특정 시간대만 분석 가능하다.
- 조회한 특정 SQL ID에 대한 Executions 그래프와 기본 성능 정보가 하단 리스트로 출력되며 SQL TEXT, BIND, PLAN 정보를 출력한다.

SQL 중점으로 TOP –DOWN 분석 가능합니다!



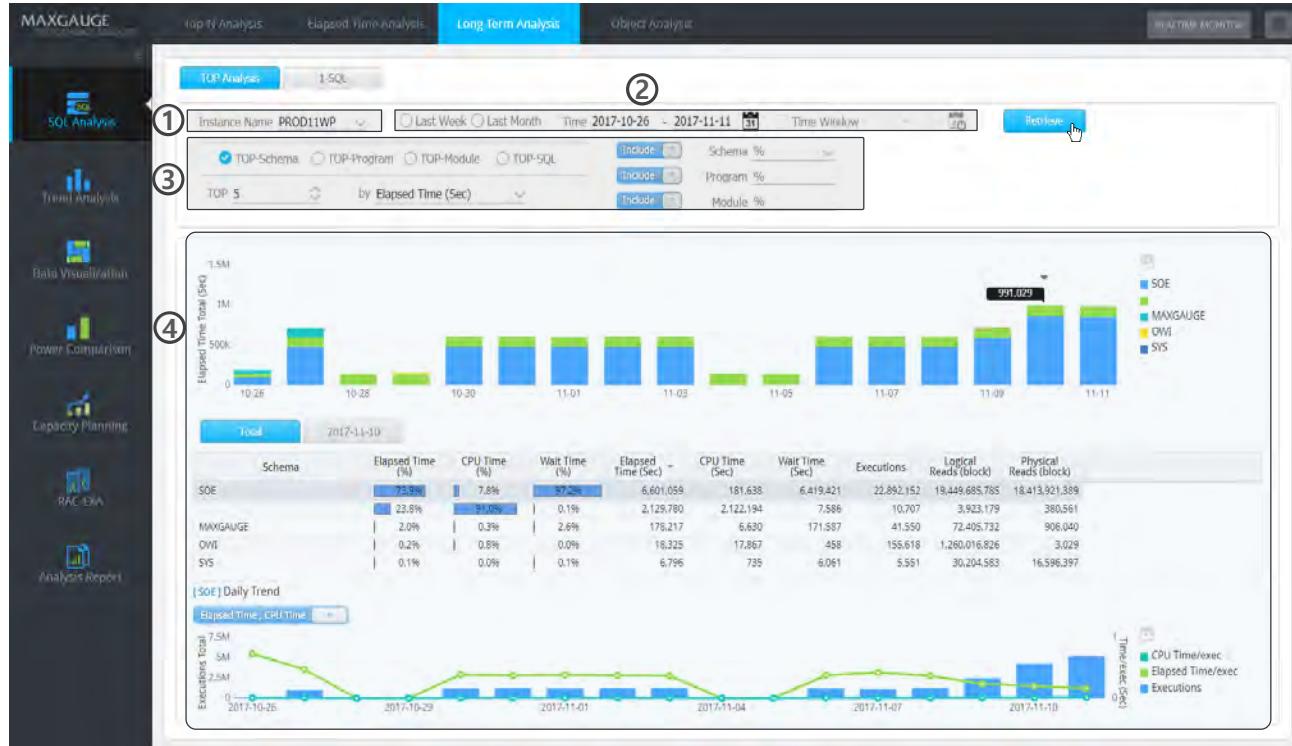
Long-Term Analysis의 활용

Long-Term Analysis의 사용 방법

Long-Term Analysis

- TOP Analysis
- 1 SQL Analysis

» TOPAnalysis 사용

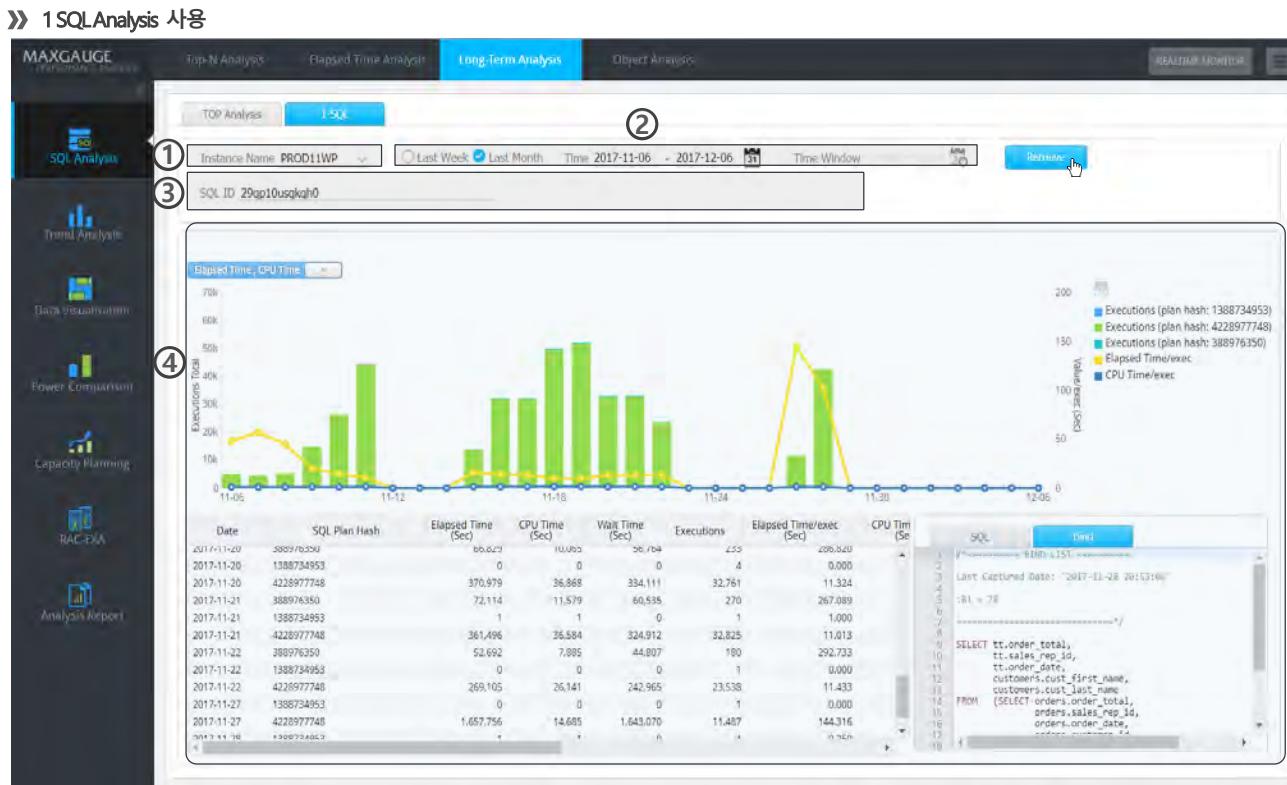


- 1 Instance를 선택합니다.
- 2 확인할 기간과 시간대를 선택합니다.
- 3 TOP-N 조건과 TOP 추출 개수, 검색 조건 포함 여부를 선택합니다.
- 4 Retrieve 버튼을 클릭하면, 조회한 정보에 대한 TOP-N장기 추이 분석 내용이 출력됩니다.

Long-Term Analysis의 사용 방법

■ Long-Term Analysis

- TOP Analysis
 - 1 SQL Analysis



- 1 Instance를 선택합니다.
 - 2 확인할 기간과 시간대를 선택합니다.
 - 3 특정 SQL ID를 입력합니다.
 - 4 Retrieve 버튼을 클릭하면, 조회한 SQL ID에 대한 장기 추이 분석 내용이 출력됩니다.

실전 분석 사례 Case 1.
**특정 Schema에서 매일 빈번하게
동작하는 SQL을 찾고 싶어요!**

STEP

1. Top Schema 추출

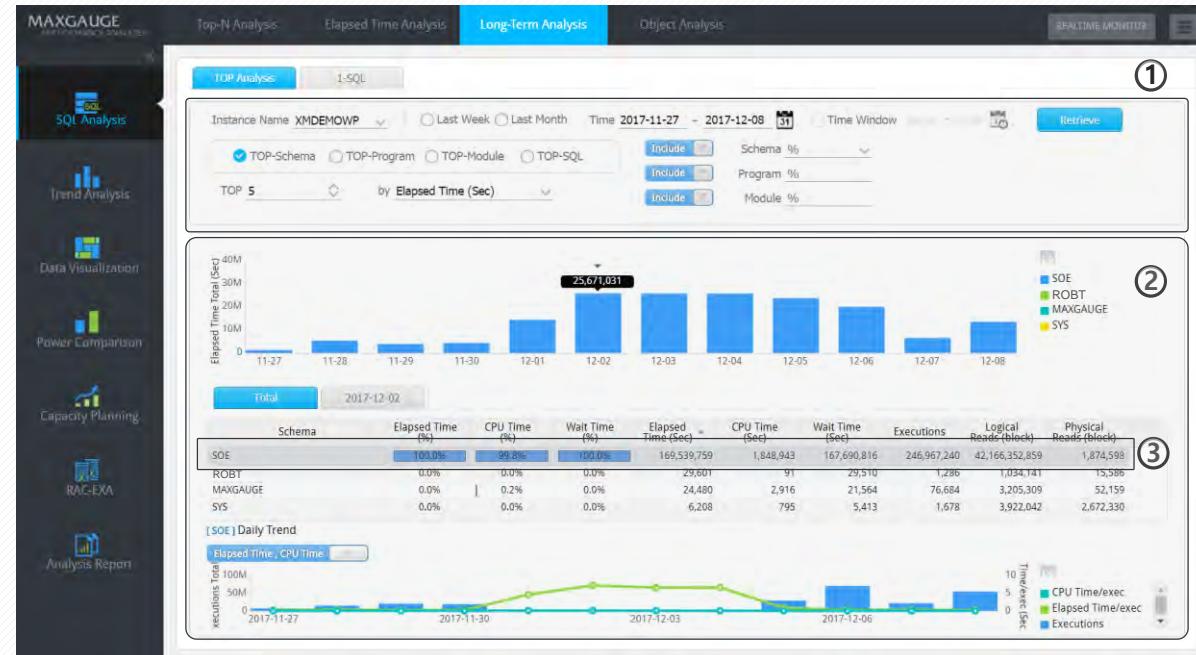
- Top Schema 조건으로 검색
 - Top Analysis > TOP-Schema 조회

2. 특정 Schema의 Top SQL 추출

- Top SQL 조건으로 검색
 - Top Analysis > TOP-SQL 조회
- 부하의 원인이 되는 SQL조회
- 특정 SQL의 상세 정보 확인
 - SQL Detail 연계

“특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요! ”

» Top Analysis > TOP-Schema 조회



✓ 시나리오

INSTANCE : XMDEMOWP
수행 확인 일자 : 2017년 11월 27일
~ 2017년 12월 08일

✓ 목표

특정 Schema에서 동작하는 TOP SQL을 추출
해 일별 시간대별 빈번하게 동작하는 SQL을
조회

- SQL Analysis ▶ Long-Term Analysis ▶ TOP Analysis에서 인스턴스(XMDEMOWP) 명, 날짜(2017.11.27~2017.12.08), TOP-Schema 조건으로 조회합니다.
- 인스턴스에 대한 TOP Schema 정보가 하단에 출력됩니다. 해당 내용을 확인해, 인스턴스 리스트 중 활동량이 많은 Schema를 확인합니다.
- 특정 Schema를 클릭 시, 하단 Daily Trend 탭으로 시간대 별 이력을 확인할 수 있습니다.

XDEMOWP 인스턴스 확인 결과, Elapsed Time, CPU, Wait 값이 높은 SOE Schema에 대해서 분석하도록 하겠습니다.

“특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요!”

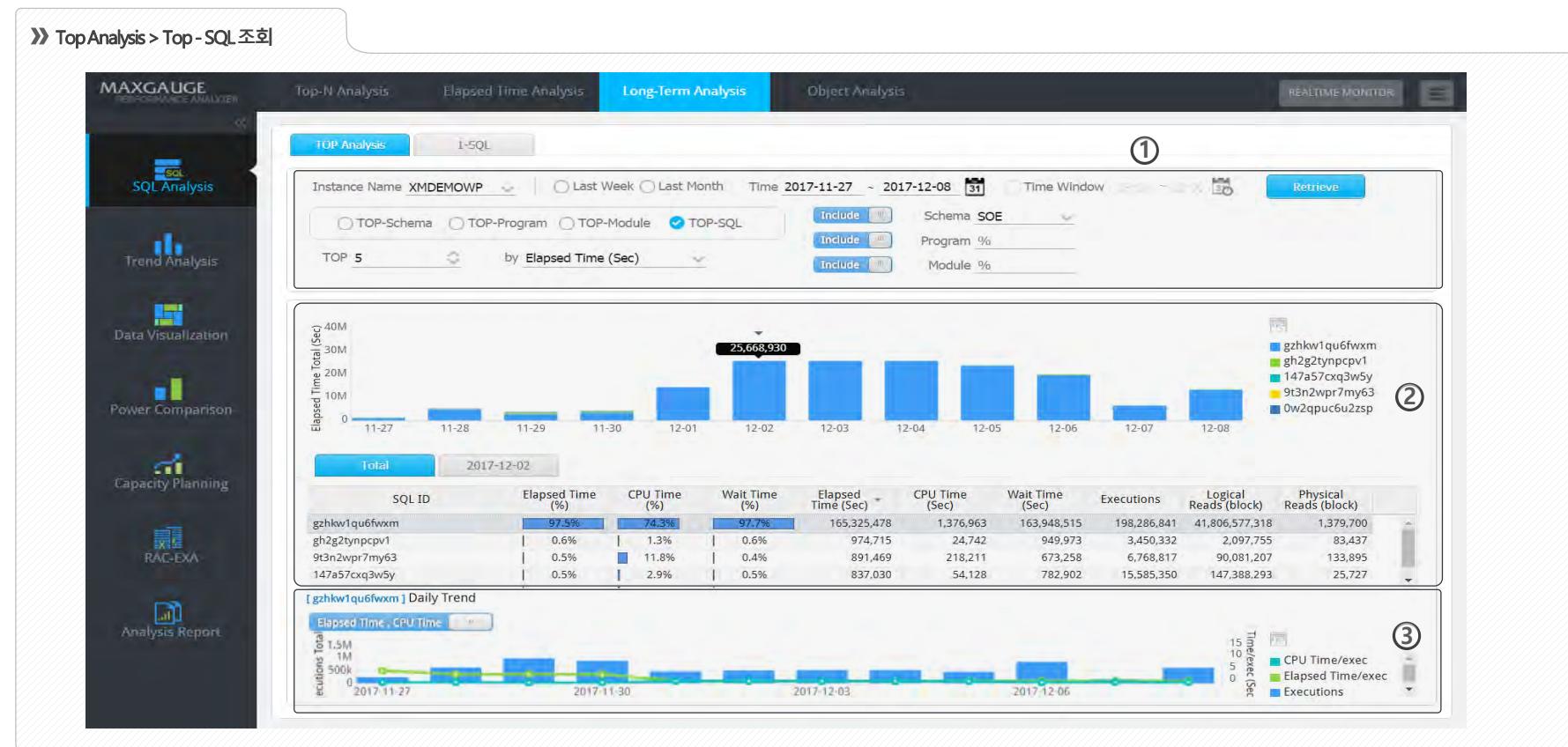
STEP

1. Top Schema 추출

- Top Schema 조건으로 검색
 - Top Analysis > TOP-Schema 조회

2. 특정 Schema의 Top SQL 추출

- Top SQL 조건으로 검색
 - Top Analysis > TOP-SQL 조회
- 부하의 원인이 되는 SQL조회
- 특정 SQL의 상세 정보 확인
 - SQL Detail 연계



1 SQL Analysis ▶ Long-Term Analysis ▶ TOP Analysis에서 인스턴스(XMDEMOWP) 명, 날짜(2017.11.27~2017.12.08), Schema(SOE), TOP-SQL 조건으로 조회합니다.

2 Schema에 동작하는 쿼리 중, Elapsed Time 기준의 TOP SQL 정보를 하단에 출력합니다. 기준은 변경 가능합니다.

3 특정 SQL을 클릭할 경우, Daily Trend 탭으로 특정 SQL의 일별 동작 이력을 확인할 수 있습니다.

“특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요!”

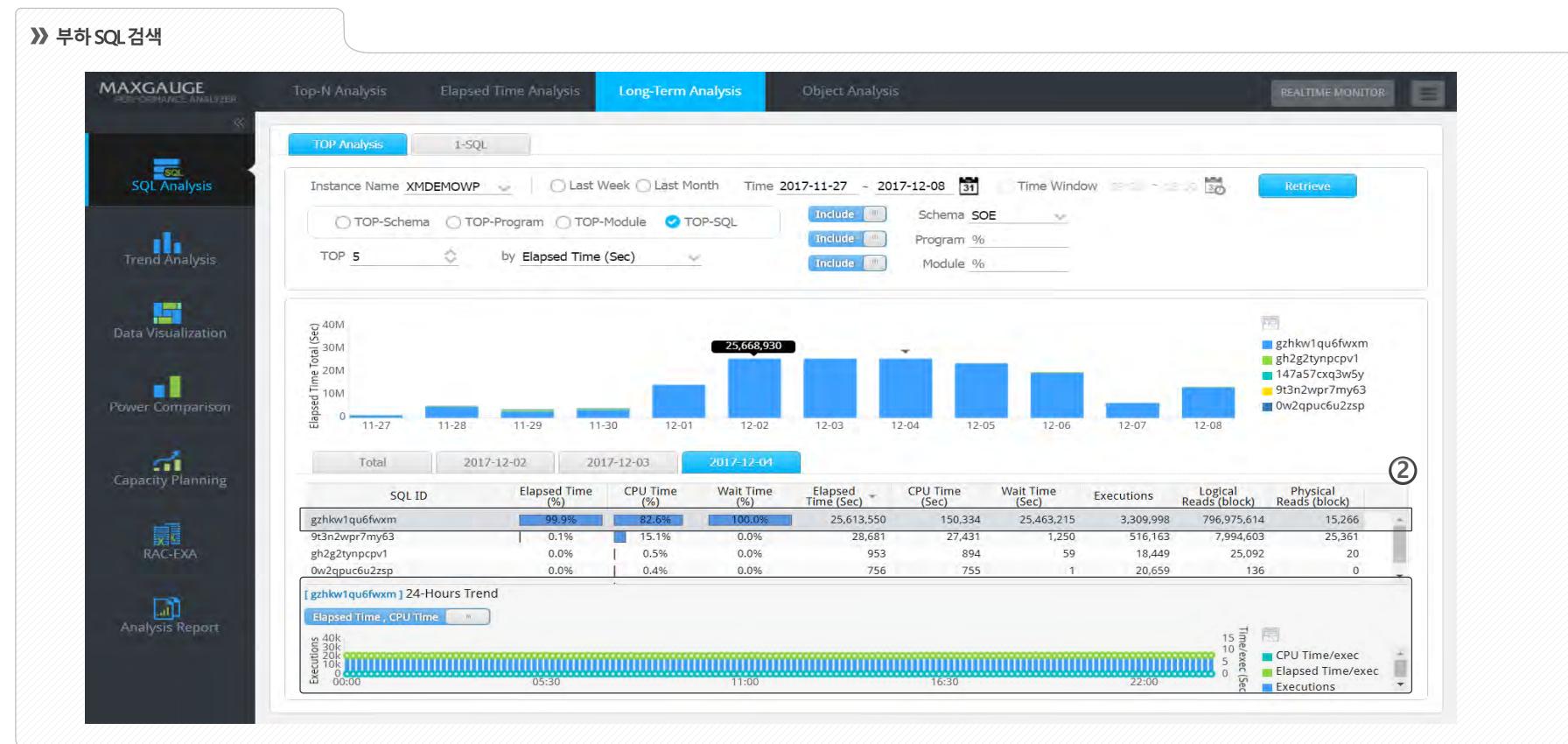
STEP

1. Top Schema 추출

- Top Schema 조건으로 검색
 - Top Analysis > TOP-Schema 조회

2. 특정 Schema의 Top SQL 추출

- Top SQL 조건으로 검색
 - Top Analysis > TOP-SQL 조회
- 부하의 원인이 되는 SQL 조회
- 특정 SQL의 상세 정보 확인
 - SQL Detail 연계



✓ 부하의 원인이 되는 SQL ID를 확인합니다.

1 매일 TOP SQL에 포함되는 SQL

2 특정 날짜 클릭해 하단 24-Hours Trend 확인 결과, 빈도수가 높은 SQL

✓ XMDEMOWP의 SOE Schema 확인 결과, 특정 SQL ID(gzhkw1qu6fwxm)가 매일 매 시간대 부하를 발생시키는 것으로 확인됩니다.

※ 해당 탭으로 일시적으로 발생한 TOP SQL과 주기적으로 발생하는 TOP SQL을 구분할 수 있습니다.

“특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요! ”

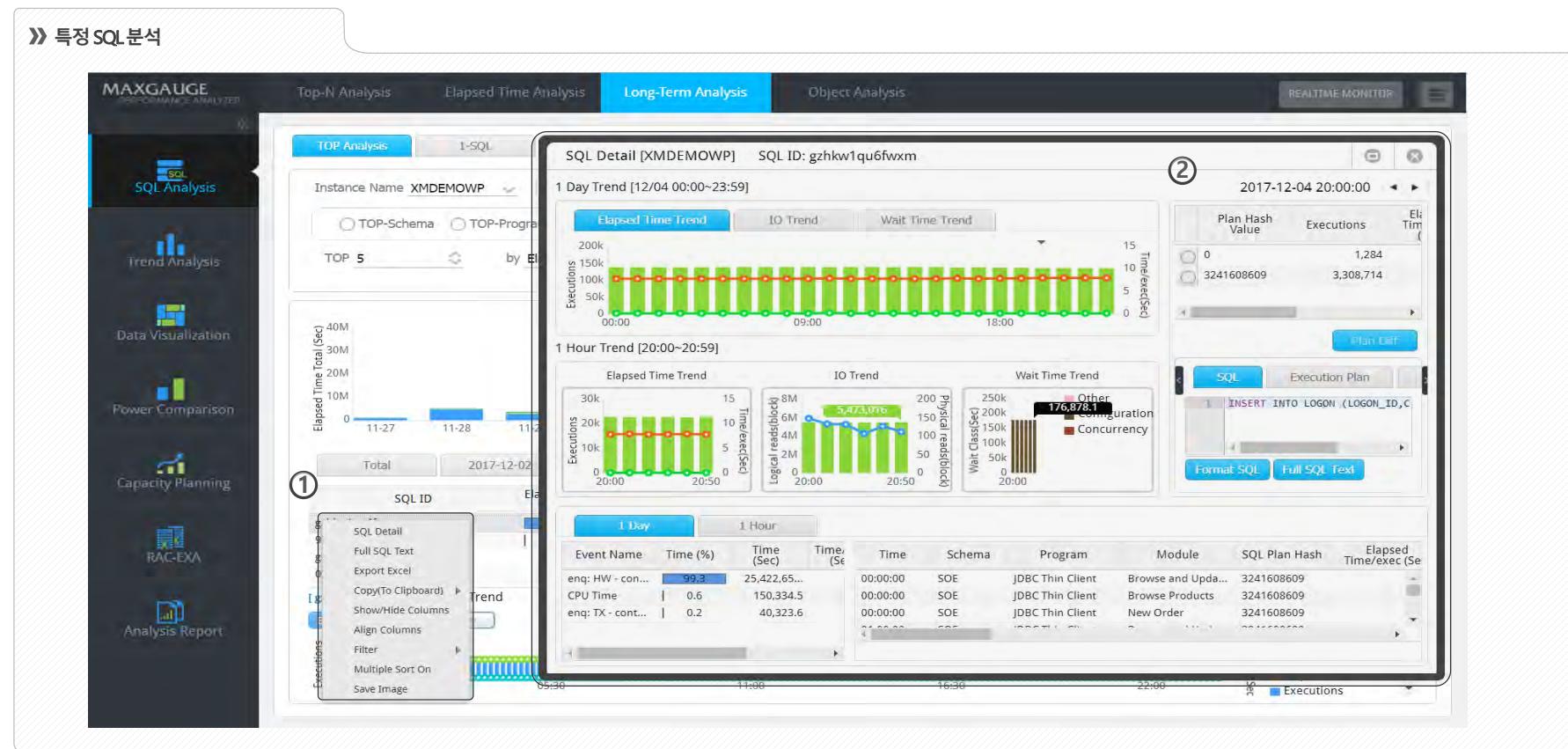
STEP

1. Top Schema 추출

- Top Schema 조건으로 검색
 - Top Analysis > TOP-Schema 조회

2. 특정 Schema의 Top SQL 추출

- Top SQL 조건으로 검색
 - Top Analysis > TOP-SQL 조회
- 부하의 원인이 되는 SQL 조회
- 특정 SQL의 상세 정보 확인
 - SQL Detail 얻기



① 부하 쿼리로 판단되는 SQL의 상세 수행 정보를 SQL Details를 통해 확인합니다.

SQL ID(gzhkw1qu6fwxm) 선택한 후, 오른쪽 마우스 클릭해 SQL Detail 정보 출력합니다.

② SQL Detail에서 SQLTEXT, PLAN, BIND, IO, 1 Day Summary, 1 Hour Summary 정보를 확인할 수 있습니다.

해당 정보를 확인 후, SQL 개선 방향에 대해 생각해 조치합니다.

실전 분석 사례 Case 2.

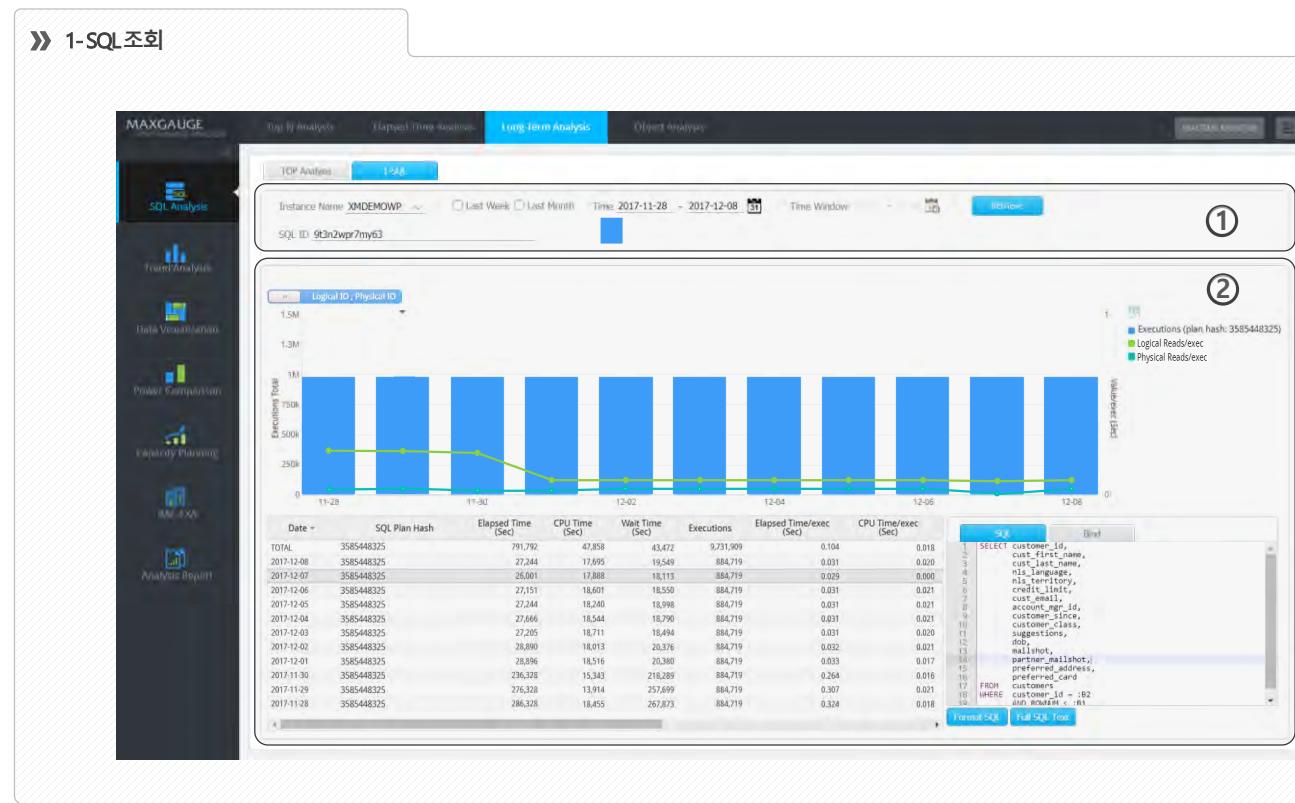
SQL 개선 후,
상태를 확인하고 싶어요!

“SQL 개선 후, 상태를 확인하고 싶어요!”

STEP

1. 특정 SQL 검색

- SQL ID 입력 후 검색
 - 1-SQL 조회
- SQL 분석 진행



✓ 시나리오

INSTANCE : XMDEMOWP
SQL ID : 9t3n2wpr7my63
수행 확인 일자 : 2017년 11월 28일
~ 2017년 12월 08일

✓ 목표

특정 SQL 개선 후, 일별 SQL 성능 분석.

- ✓ 12월 1일에 개선한 특정 SQL ID에 대한 사후 분석을 진행합니다.

① SQL Analysis ▶ Long-Term Analysis ▶ 1-SQL에서 인스턴스(XMDEMOWP) 명, 날짜(2017.11.30~2017.12.11), SQL ID (9t3n2wpr7my63)

조건으로 조회합니다.

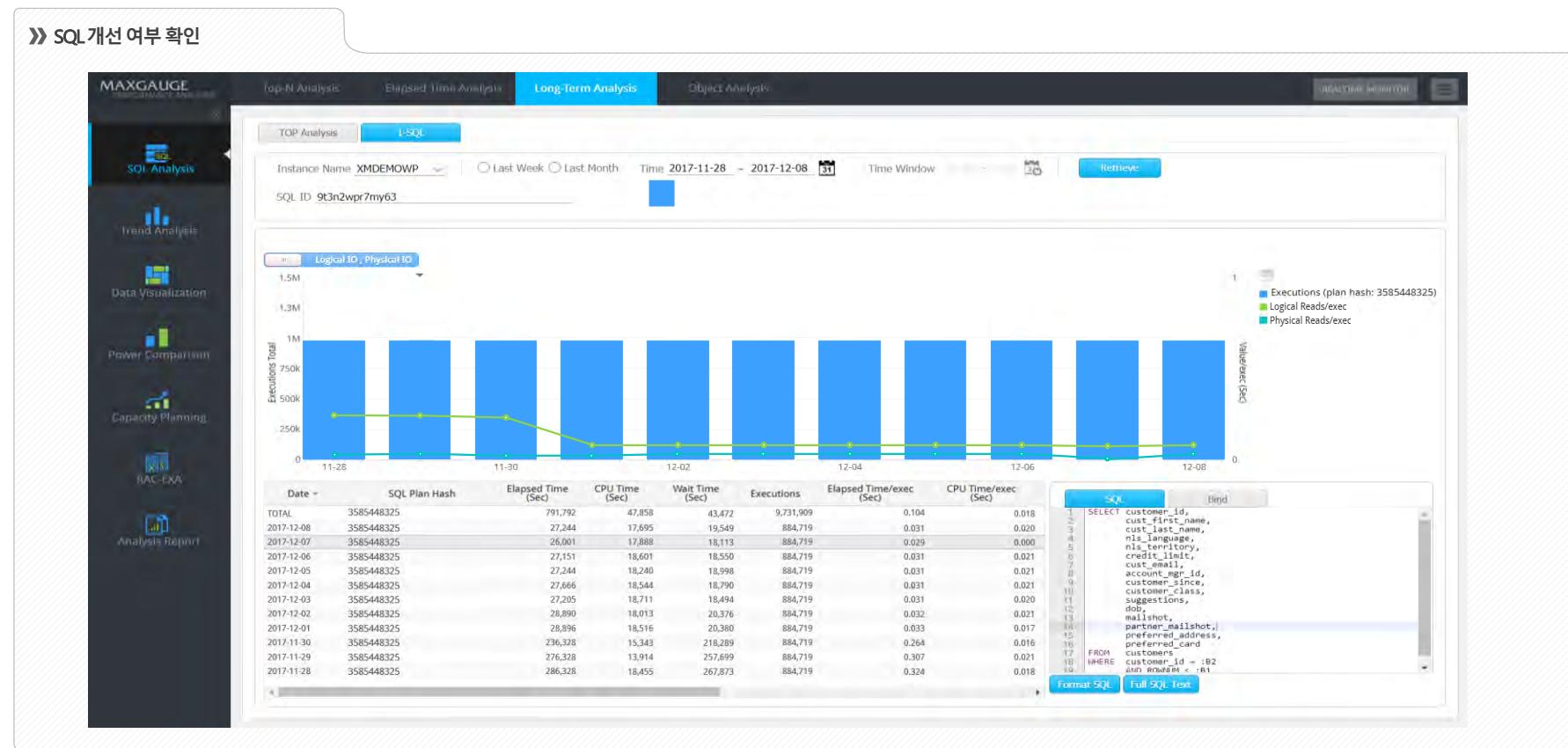
② SQL ID(9t3n2wpr7my63)에 대한 일자 별 추이, SQL TEXT, BIND 정보가 출력됩니다.

“SQL 개선 후, 상태를 확인하고 싶어요!”

STEP

1. 특정 SQL 검색

- SQL ID 입력 후 검색
 - ↳ SQL 조회
- SQL 분석 진행



- ✓ 해당 SQL ID의 12월 1일 전 후 성능 정보를 확인 가능합니다.
- ✓ 하단 리스트에는 일별 SQL ID 성능 정보를 출력하며, SQL PLAN 변화가 있을 경우 SQL Plan 마다의 일별 성능 추이를 출력합니다.

1 XMDEMOWWP의 SQL ID(9t3n2wpr7my63) 확인 결과 매일 일정량 Execution을 유지하나, 12월 1일 이후로 wait Time, CPU Time값이 감소한 것을 확인할 수 있습니다. 추가로 그래프 상으로 IO값이 감소한 것을 확인 가능합니다.

MAXGAUGE



Practical Guide

Plan Diff [Performance Analyzer]

Contents

Plan Diff?

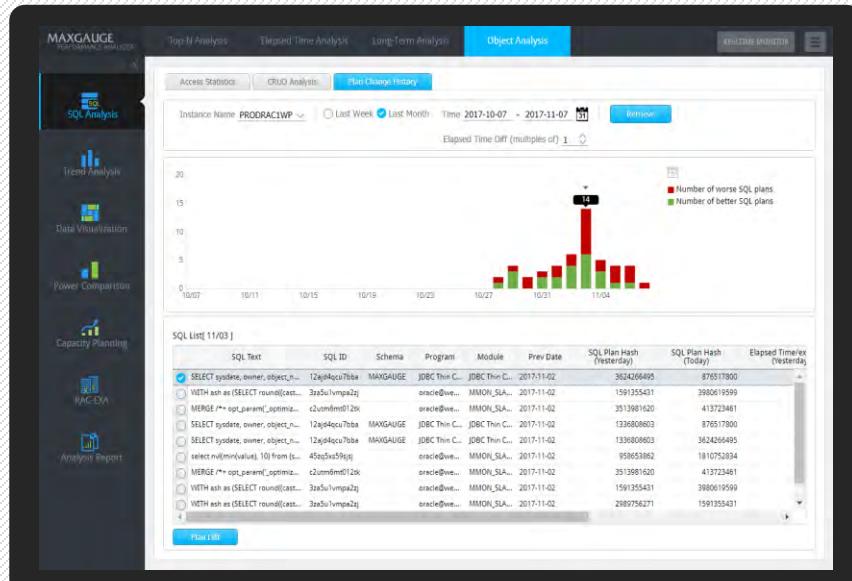
Plan Diff 의 활용

Case 1. SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요.

Case 2. 통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요!

Plan Diff?

Plan Diff는 언제 쓰나요?



MaxGauge

case 1.

SQL이 갑자기 느려 졌습니다.
Plan의 변경 여부를 확인하고 싶습니다.

case 2.

통계정보 갱신 작업을 하였습니다.
DB 전반의 SQL Plan을 확인하고 싶습니다.

Plan Diff?

Plan Diff

SQL Plan Diff SQLID: 12ajd4qcu7bba

Time	Sql Plan Hash	Executions	Elapsed Time/exec (Sec)	CPU Time/exec (Sec)	Logical Reads/exec (block)	Physical Reads/exec (block)
2017-11-02 16:10:00	1336808603	109	0.214	0.214	4,591	0
2017-11-03 23:50:00	876517800	139	0.246	0.244	4,325	0

Prev [1336808603]

ID	PLAN
0	* SELECT STATEMENT *** COUNT (STOPKEY) ***** TABLE ACCESS (BY INDEX ROWID)SOE.ADDRESSES' (TABLE) ***** INDEX (RANGE SCAN)SOE.ADDRESS_CUST_IDX' (INDEX)
id	Plan Hash Value : 900611645
*0	SELECT STATEMENT
*1	COUNT (STOPKEY) <i>filter(ROWNUM<:B1)</i>
*2	TABLE ACCESS (BY INDEX ROWID)SOE.CUSTOMERS' (TABLE) INDEX (UNIQUE SCAN)SOE.CUSTOMERS_PK' (INDEX (UNIQUE)) <i>access("CUSTOMER_ID":>B2)</i>
*3	

Predicate Information (identified by operation id)

1* - filter(ROWNUM<:B1)
3* - access("CUSTOMER_ID":>B2)

Current [876517800]

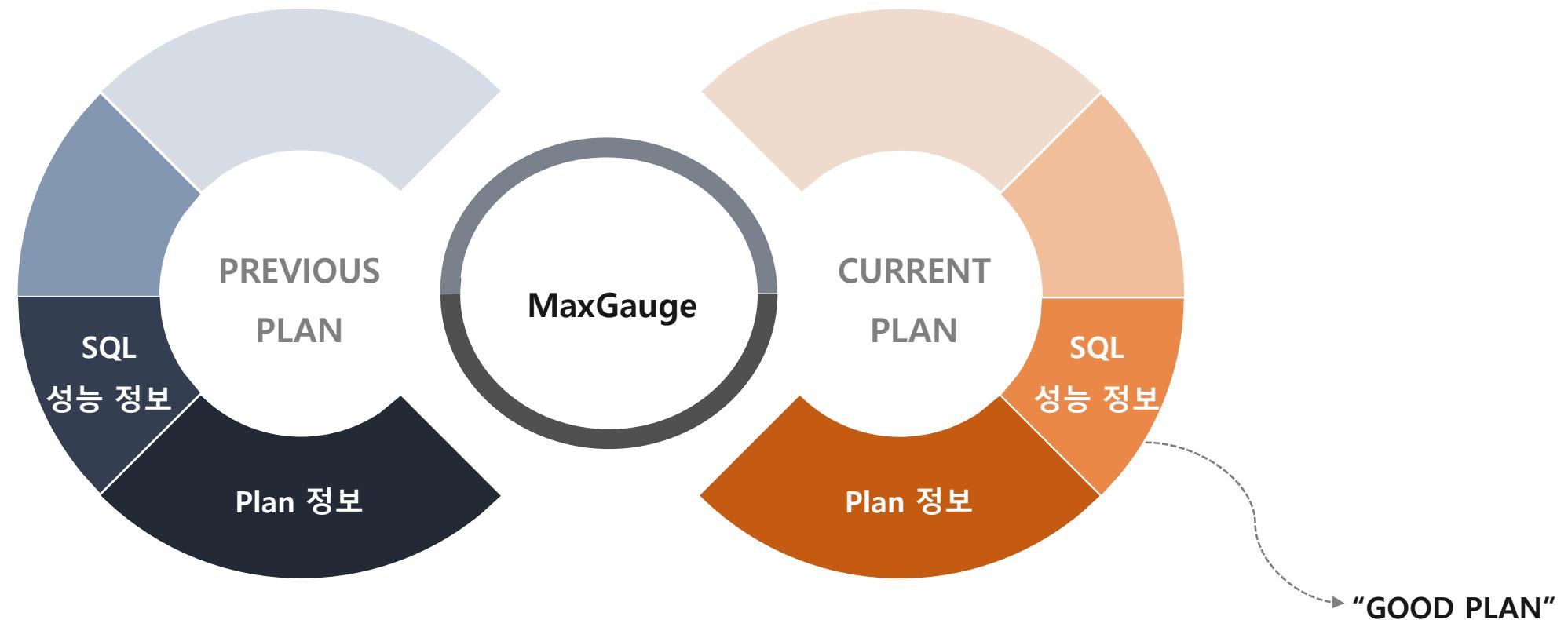
ID	PLAN
0	* SELECT STATEMENT *** COUNT (STOPKEY) ***** TABLE ACCESS (BY INDEX ROWID)SOE.CUSTOMERS' (TABLE) ***** INDEX (UNIQUE SCAN)SOE.CUSTOMERS_PK' (INDEX (UNIQUE))
id	Plan Hash Value : 3585448325
*0	SELECT STATEMENT
*1	COUNT (STOPKEY) <i>filter(ROWNUM=<:B1)</i>
*2	TABLE ACCESS (BY INDEX ROWID)SOE.ADDRESSES' (TABLE) INDEX (RANGE SCAN)SOE.ADDRESS_CUST_IDX' (INDEX) <i>(access("CUSTOMER_ID">>:B2)</i>
*3	

Predicate Information (identified by operation id)

1* - filter(ROWNUM<:B1)
3* - access("CUSTOMER_ID":>B2)

- 다른 Plan 을 갖는 동일한 SQL의 두 Plan을 비교하는 기능이다.
- Plan hash value 별로 SQL 성능 정보를 제공한다.
- 해당 기능은 Performance Analyzer 의 SQL Detail Window 팝업 창에서 제공한다.
- 일반적으로 Performance Analyzer ▶ SQL Analysis ▶ Object Analysis ▶ Object Analysis ▶ Plan Change History 와 연계하여 사용된다.

Plan 비교와 Plan 변경 원인 파악이 쉽습니다.



Plan Diff의 활용

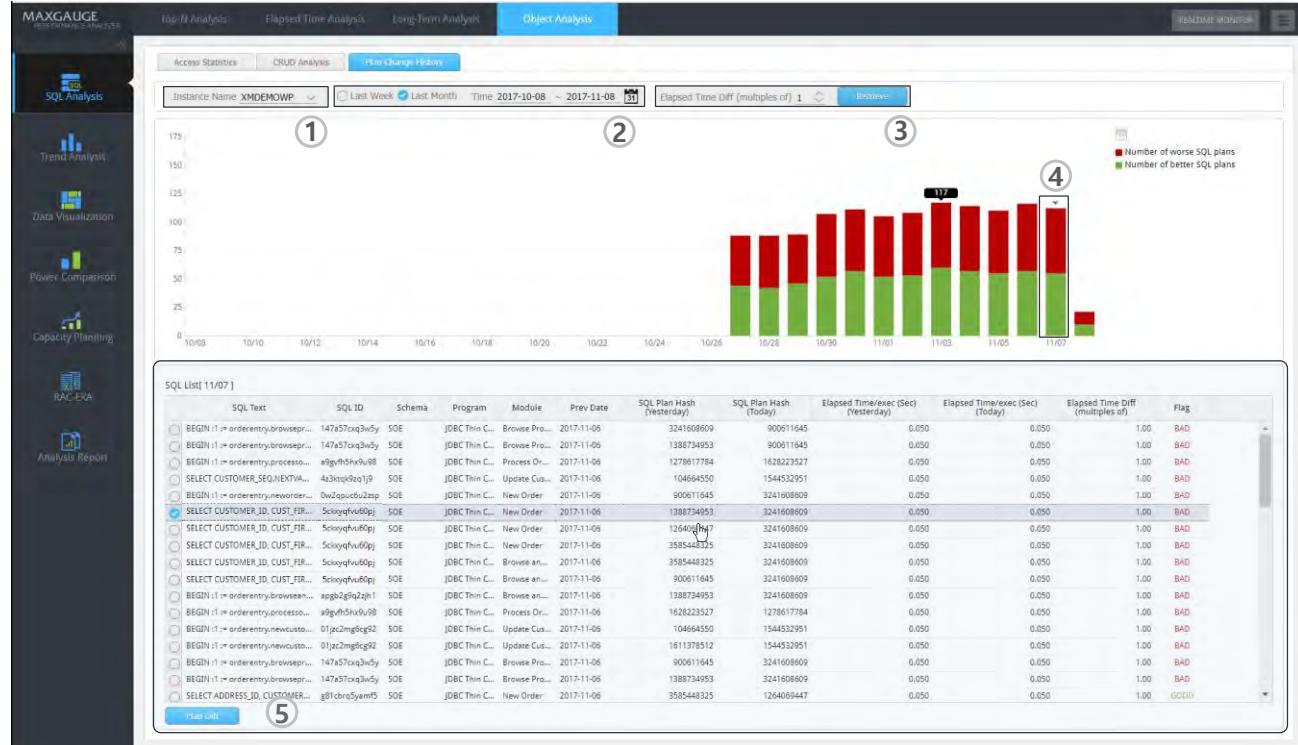
Plan Diff의 활용

Plan Diff의 사용 방법

Plan Diff

- 검색 조건 설정
- Plan 비교 및 분석

» Plandiff 사용



- 1 Instance를 선택합니다.
- 2 Plan 변경이 이루어졌는지 확인할 기간과 선택합니다.
- 3 Plan에 따라 수행된 SQL의 응답시간 (Elapsed Time diff) 차이 배수를 선택합니다.
- 4 날짜를 클릭하면, 해당 날짜에 수행된 SQL에 대한 SQL PLAN 및 수행정보가 하단 화면에 리스트화 되어 나타납니다.
- 5 Plan 변경을 확인하고자 하는 SQL을 체크 후, Plan diff를 클릭합니다.

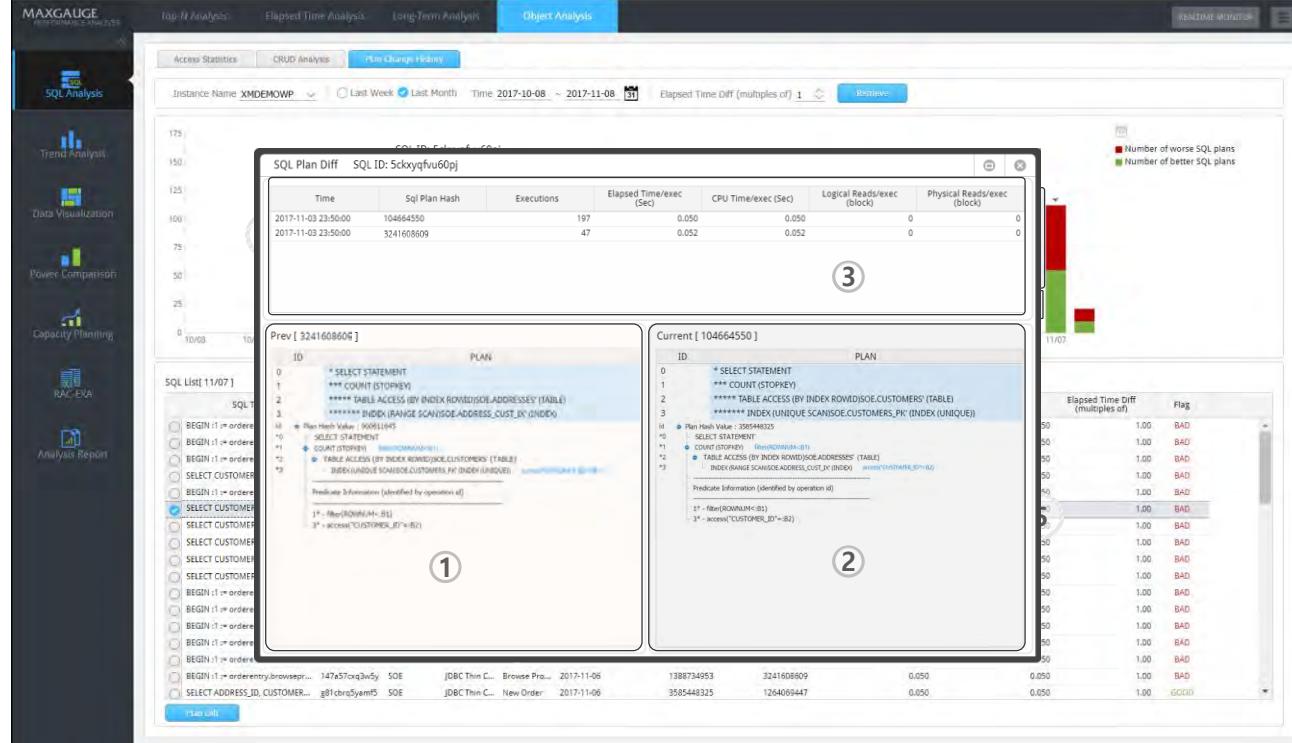
Plan Diff의 활용

Plan Diff의 사용 방법

Plan Diff

- 검색 조건 설정
- Plan 비교 및 분석

» Plan Diff 조회



- 변경 전 Execution Plan에 대한 정보를 제공합니다.
- 변경 후 Execution Plan에 대한 정보를 제공합니다.
- Plan Hash Value 별 SQL 성능정보(Executions, Elapsed Time, CPU Time, Wait Time, Logical Reads, Physical Reads) 를 제공합니다.

실전 분석 사례 Case 1.
**SQL이 느려졌어요, Plan의 변경
여부를 확인하고 싶어요.**

Plan Diff의 활용

"SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요!"

STEP

1. 데이터 분석

▪ 느려진 SQL 성능 확인

└ Trend Analysis ▷ SQL List 조회

▪ 느려진 SQL의 상세 정보 확인

└ SQL Detail 연계

▪ 느려진 SQL의 Plan 비교

└ Plan diff 연계

» SQL LIST >> SQL 조회

From Time	Schema	Program	Module	SQL Text	SQL ID	SQL Plan Hash	Executions	Elapsed Time (%)	CPU Time (%)	Elapsed Time (Sec)	CPU Time (Sec)	Elapsed Time/CPU (Sec)	Logical Reads	Logical Block Read (Blocks)	Physical Reads	Physical Block Read (Blocks)
11-18 00:00	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	62	1.0%	1.0%	3	0.050	3	0.046	0	0	
11-18 00:10	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	0	1	0.0%	0.0%	0	0.050	0	0.050	0	0	
11-18 00:19	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	62	0.8%	0.8%	3	0.050	2	0.047	0	0	
11-18 00:20	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	54	0.9%	0.8%	3	0.050	2	0.045	0	0	
11-18 00:40	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	56	0.8%	0.8%	3	0.050	2	0.044	0	0	
11-18 00:50	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	73	1.1%	1.1%	4	0.050	3	0.046	0	0	
11-18 00:55	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	40	0.6%	0.6%	2	0.050	2	0.044	0	0	
11-18 01:00	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	62	1.0%	1.0%	3	0.050	3	0.046	0	0	
11-18 01:10	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	60	1.1%	1.1%	3	0.050	3	0.046	0	0	
11-18 01:20	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	63	0.8%	0.8%	3	0.050	2	0.045	0	0	
11-18 01:30	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	0	3	0.0%	0.1%	0	0.050	0	0.050	0	0	
11-18 01:30	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	53	0.8%	0.8%	3	0.050	2	0.044	0	0	
11-18 01:40	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	58	0.9%	0.8%	3	0.051	3	0.046	2,340	28	
11-18 01:50	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	65	1.0%	1.0%	3	0.050	3	0.048	0	0	
11-18 02:00	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	46	0.7%	0.8%	2	0.051	2	0.051	1,375	34	
11-18 02:10	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	66	1.0%	1.1%	3	0.050	3	0.050	0	0	
11-18 02:20	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	54	0.8%	0.9%	3	0.050	3	0.050	0	0	
11-18 02:30	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	53	0.8%	0.8%	3	0.050	3	0.050	0	0	
11-18 02:45	SOE	JDBC Thin Client	Sales Rep ...	SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...	29qp10usqkqh0	4229977748	62	0.8%	0.8%	3	0.050	3	0.050	0	0	

SQL Text:

```
SELECT TT_ORDER_TOTAL, TT_SALES REP_ID, TT.ORD...
```

SQL ID: 29qp10usqkqh0

SQL Plan Hash: 4229977748

Elapsed Time: 0.050 Sec

Logical Reads: 0 Block

Physical Reads: 0 Block

- SQL Analysis ▷ TOP-N Analysis에서 인스턴스(XMDEMOWP) 명, Schema(SOE), 날짜(2017.11.18), SQL ID(29qp10usqkqh0)로 조회합니다.
- 해당 SQL에 대한 10분당 Summary 데이터를 통해, SQL이 느려진 시간의 SQL수행 정보를 확인합니다.
- 느려진 SQL의 상세 수행정보를 확인하기 위해 오른쪽 마우스 클릭 후, SQL Detail을 선택합니다.

✓ 시나리오

INSTANCE

: XMDEMOWP

SCHEMA

: SOE

PROGRAM

: JDBC Thin Client

MOUDLE

: Sales Rep Query

수행 확인 일자

: 2017년 11월 18일

✓ 목표

특정 SQL이 느려 짐을 확인함.

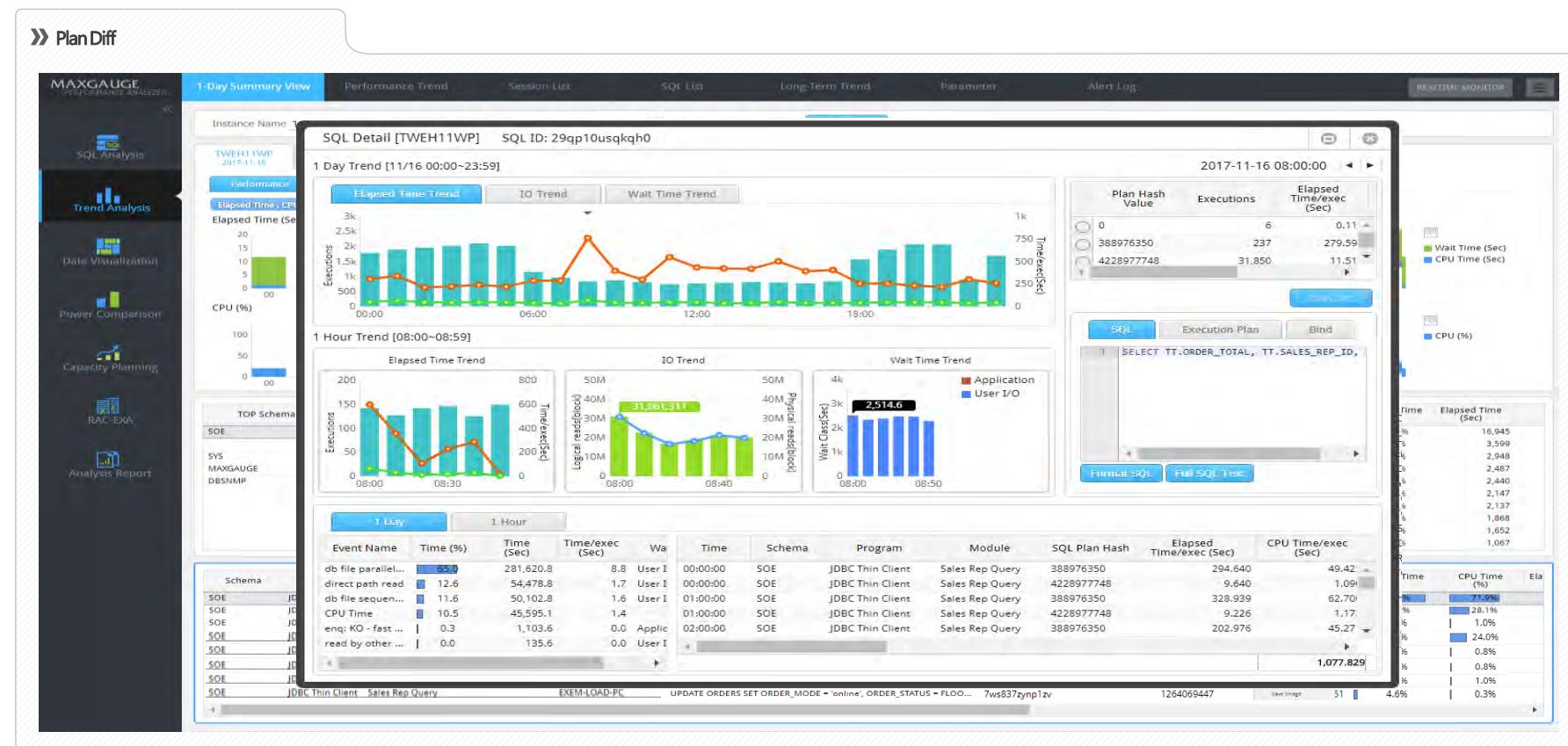
SQL 느려진 원인 파악을 위해 해당 SQL의 수행 정보 및 PLAN 확인.

“SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요!”

STEP

1. 데이터 분석

- 느려진 SQL 성능 확인
 - └ Trend Analysis ▷ SQL List 조회
- 느려진 SQL의 상세 정보 확인
 - └ SQL Detail 연계
- 느려진 SQL의 Plan 비교
 - └ Plan diff 연계



- ✓ SQL Details에서, SQL 상세 수행 정보를 확인합니다.
- ✓ 상단 우측 Plan Hash Value 뷰에서 Hash value가 여러 개임을 확인합니다.
- ✓ Plan Hash value가 두 개 이상일 경우, Plan 변경으로 인해 SQL 수행이 오래 걸릴 수 있습니다.
- ✓ Plan Hash value 별 Elapsed Time, Executions 수치를 포함한 SQL 수행정보를 확인 후, 비교하고자 하는 Plan Hash value를 체크하고 plan diff를 클릭합니다.

“SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요!”

STEP

1. 데이터 분석

▪ 느려진 SQL 확인

└ Trend Analysis ▷ 1-Day Summary View 연계

▪ 느려진 SQL의 상세 정보 확인

└ SQL Detail 연계

▪ 느려진 SQL의 Plan 비교

└ Plan diff 연계

» Plan Diff

MAXGAUGE PERFORMANCE ANALYZER

1-Day Summary View Performance Trend Session List SQL List Long-Term Trend Parameter Alert Log REALTIME MONITOR

Instance Name: TWFH11WP 2019-11-16

Performance Elapsed Time, CPU Elapsed Time (Sec)

CPU (%)

TOP Schema: SOE

Schema: SOE, SOE, SOE, SOE, SOE, SOE, SOE, SOE

SQL Detail [TWFH11WP] SQL ID: 29qp10usqkqh0

SQL Plan Diff SQLID: 29qp10usqkqh0

Time	SQL Plan Hash	Executions	Elapsed Time/exec (Sec)	CPU Time/exec (Sec)	Logical Reads/exec (block)	Physical Reads/exec (block)
2017-11-16 08:50:00	388976350	31,850	11.516	1.108	138,917	137,606
2017-11-16 23:50:00	428977748	237	279.597	12.472	6,306,427	6,276,313

Prev [388976350]

ID	PLAN
0	SELECT STATEMENT
1	* NESTED LOOPS
2	*** NESTED LOOPS
3	***** VIEW
4	***** WINDOW (SORT PUSHED RANK)
5	***** TABLE ACCESS (BY INDEX ROWID) SOE.ORDERS(TABLE)
6	***** INDEX (RANGE SCAN) SOE.ORD_SALES REP_IDX(INDEX)
7	**** INDEX (UNIQUE SCAN) SOE.CUSTOMERS_PK(INDEX (UNIQUE))
8	*** TABLE ACCESS (BY INDEX ROWID) SOE.CUSTOMERS(TABLE)

Current [428977748]

ID	PLAN
0	SELECT STATEMENT
1	* NESTED LOOPS
2	*** NESTED LOOPS
3	***** VIEW
4	***** WINDOW (SORT PUSHED RANK)
5	***** TABLE ACCESS (BY INDEX ROWID) SOE.ORDERS(TABLE)
6	***** INDEX (SKIP SCAN) SOE.ORD_SALES REP_IDX(INDEX)
7	**** INDEX (UNIQUE SCAN) SOE.CUSTOMERS_PK(INDEX (UNIQUE))
8	*** TABLE ACCESS (BY INDEX ROWID) SOE.CUSTOMERS(TABLE)

- ✓ SQL Plan Diff에서 해당 SQL의 상세 수행 정보를 비교합니다.
- ✓ 하단 화면에서 SQL이 다를 경우 하이라이트 되어 보여집니다.
- ✓ Current SQL과 Prev SQL의 Plan을 비교합니다.
- ✓ Index Scan0| Index Skip Scan으로 바뀌어 수행되어, SQL 수행 속도에 영향을 주었음을 알 수 있습니다.

실전 분석 사례 Case 2.
통계정보를 갱신했습니다.
전반적인 Plan 정보를 알고 싶어요!

“통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요!”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Plan 변경 및 성능 확인
- "BAD" Plan 성능 분석
 - └ Plan Diff 연계
- "GOOD" Plan 성능 분석
 - └ Plan Diff 연계

» Plan Diff 화면

The screenshot shows the MAXGAUGE interface with the 'Object Analysis' tab active. Step 1 highlights the selection of the 'Instance Name' (TXMDEMOWP). Step 2 highlights the time range selection ('Last Week' to 'Last Month'). Step 3 highlights the 'Elapsed Time Diff (multiples of)' dropdown set to 1. Below the summary, there is a 'SQL List' table with columns like SQL Text, SQL ID, Schema, Program, Module, Prev Date, SQL Plan Hash (Yesterday), SQL Plan Hash (Today), Elapsed Time/exec (Sec) (Yesterday), Elapsed Time/exec (Sec) (Today), Elapsed Time Diff (multiples of), and Flag. A 'Plan Diff' button is located at the bottom of this section.

- 1 Plan 정보를 확인할 Instance명(TXMDEMOWP)를 선택합니다.
- 2 통계정보 갱신 이후 날짜를 포함하여 기간을 선택합니다.
- 3 Plan 별 수행시간(Elapsed Time) 차이를 옵션으로 사용할 수 있습니다. 단위는 배수이며, 1배수도 선택 가능합니다.

✓ 시나리오

INSTANCE : TXMDEMOWP

통계 정보 갱신 일자 : 2017년 11월 12일

Plan 비교 일자 : 2017년 11월 11일

✓ 목표

통계 정보 갱신 후, 기존에 수행하던 SQL의

Plan 변경 및 성능 저하가 발생여부 파악

“통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요!”

STEP

1. 검색 조건 설정

2. 데이터 분석

▪ Plan 변경 및 성능 확인

▪ "BAD" Plan 성능 분석

└ Plan Diff 연계

▪ "GOOD" Plan 성능 분석

└ Plan Diff 연계

» Plan Diff 화면

SQL Text	SQL ID	Schema	Program	Module	Prey Date	SQL Plan Hash (Yesterday)	SQL Plan Hash (Today)	Elapsed Time/exec (Sec) (Yesterday)	Elapsed Time/exec (Sec) (Today)	Elapsed Time Diff (multiples of)	Flag
SELECT NVL(0,GOODS_SALE_PR...	a3gj0fjd9u2	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	2275396676	3241608609	1.010	0.010	0.01	GOOD
SELECT NVL(GOODS_SALE_PR...	a3gj0fjd9u2	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	2275396676	7483297179	1.000	0.010	0.01	GOOD
UPDATE ORDERS SET ORDER_MODE_PROD...	0v2epu6i2zsp	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	1623525319	494735477	1.000	0.010	0.01	GOOD
UPDATE ORDERS SET ORDER_MODE_PROD...	0v2epu6i2zsp	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	1623525319	494735477	1.000	0.010	0.01	GOOD
SELECT * FROM (SELECT TRUNC(SALE_PR...	6j3n680nvb13	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	90061645	435473321	0.010	0.010	1.00	GOOD
SELECT empno,select option,dy_code FRO...	4u8m4fn1apxpt	SOE	JDBC Thin C...	New Order	2017-11-11	1388734953	1627389057	2.500	1.000	0.40	GOOD
SELECT ename,*,(select option,dy_code FRO...	4u8m4fn1apxpt	SOE	JDBC Thin C...	New Order	2017-11-11	1388734953	1627389057	2.500	1.000	0.40	GOOD
SELECT ename,*,(select option,dy_code FRO...	4u8m4fn1apxpt	SOE	JDBC Thin C...	New Order	2017-11-11	1388734953	8372681938	10.000	1.000	0.10	GOOD
SELECT COUNT(*) FROM (SELECT /*+ INDEX...	3za5u1vmpazj	SOE	JDBC Thin C...	New Order	2017-11-11	388976350	453782398	0.100	0.010	0.10	GOOD
SELECT COUNT(*) FROM (SELECT /*+ INDEX...	3za5u1vmpazj	SOE	JDBC Thin C...	New Order	2017-11-11	388976350	453782398	0.100	0.010	0.10	GOOD
UPDATE ORDERS SET ORDER_MODE_PROD...	0v2epu6i2zsp	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	2141863993	662741834	3.000	1.000	0.33	GOOD
UPDATE ORDERS SET ORDER_MODE_PROD...	0v2epu6i2zsp	SOE	JDBC Thin C...	Browse Pro...	2017-11-11	1623772625	128361273	0.010	0.009	0.00	BAD

- ✓ 막대 그래프 화면에서 통계정보 갱신 이후 날짜(2017.11.11)를 선택합니다.
- ✓ 하단의 SQL List 화면을 통해 통계정보 갱신 전 날(2017.11.12)과 비교된 SQL PLAN 정보를 확인할 수 있습니다.
- ✓ Flag 칼럼은 Previous Plan의 Elapsed Time을 Current Plan Elapsed Time으로 나눈 값으로 산정합니다.
- ✓ Flag 칼럼이 "GOOD"인 SQL은 Previous Plan보다 수행속도가 좋아진 SQL입니다.
- ✓ Flag 칼럼이 "BAD"인 SQL은 Plan 이전보다 성능이 저하 된 SQL로 수행성능 및 분석이 필요합니다.

“통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요!”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Plan 변경 및 성능 확인
- “BAD” PLAN 성능 분석
 - └ Plan Diff 연계
- “GOOD” PLAN 성능 분석
 - └ Plan Diff 연계

» Plan Diff 조회 화면

- ✓ Plan 변경 여부를 확인하고자 SQL(SQL ID: 6j3rn680twb13)을 선택하고, Plan Diff를 클릭합니다.
- ✓ SQL Plan Diff 화면에서 통계 변경 전 SQL PLAN(Hash value:388976350)과 통계 정보 갱신 된 후 SQL PLAN(Hash Value:238977748)을 확인할 수 있습니다.
- ✓ Current Plan(ID=4)에서 Prev Plan과 다른 인덱스(SOE.OP_ORD_DTL_INFO_IX010)로 Index Scan을 하는 것을 알 수 있습니다.
- ✓ 인덱스 정보 및 SQL 수행 이력을 고려하여, 해당 SQL 개선 여부를 고려해야 합니다.

“통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요!”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Plan 변경 및 성능 확인
- “BAD” PLAN 성능 분석
 - └ Plan Diff 연계
- “GOOD” PLAN 성능 분석
 - └ Plan Diff 연계

» PlanDiff 조회 화면

Time	Sql Plan Hash	Executions	Elapsed Time/exec (Sec)	CPU Time/exec (Sec)	Logical Reads/exec (block)	Physical Reads/exec (block)
2017-11-12 23:50:00	1632772635	5,147	8.00	3.58	123,935,000	23,935,000
2017-11-11 23:50:00	128361273	256	0.10	0.01	79,351	9,351

Prev [388976350]

ID	PLAN
0	SELECT STATEMENT
1	* SORT AGGREGATE
2	*** FILTER
3	***** TABLE ACCESS (BY INDEX ROWID) SOE.OP_ORD_DTL_INFO (TABLE)
4	***** INDEX (RANGE SCAN) SOE.OP_ORD_DTL_INFO_IDX04(INDEX)

Current [428977748]

ID	PLAN
0	SELECT STATEMENT
1	* SORT AGGREGATE
2	*** FILTER
3	***** TABLE ACCESS (BY INDEX ROWID) SOE.OP_ORD_DTL_INFO (TABLE)
4	***** INDEX (RANGE SCAN) SOE.OP_ORD_DTL_INFO_IDX04(INDEX)

Elapsed Time Diff (Multiples of)

Flag	
0.01	GOOD
1.00	GOOD
1.00	GOOD
0.40	GOOD
0.40	GOOD
0.10	GOOD
0.10	GOOD
0.33	GOOD
0.33	GOOD
0.00	BAD

- ✓ Plan 변경 여부를 확인하고자 SQL(SQL ID: 3za5u1vmpa2zj)을 선택하고, Plan Diff를 클릭합니다.
- ✓ SQL Plan Diff 화면에서 통계 변경 전 SQL PLAN(Hash value: 388976350)과 통계 정보 갱신 된 후 SQL PLAN(Hash Value: 453782398)을 확인할 수 있습니다.
- ✓ Current Plan(ID=4)에서 Prev Plan과 다른 인덱스(SOE.SYS_MAIL SND_QUE_INFO_IDX04)로 Index Scan을 하는 것을 알 수 있습니다.
- ✓ 상단 SQL 수행 이력에서 이전에 비해 I/O와 Elapsed Time이 개선되었음을 알 수 있습니다.

MAXGAUGE



Practical Guide

Wait Time Analysis [Performance Analyzer]

Contents

Wait Time Analysis?

Wait Time Analysis의 활용

Case 1. Real-Time Monitor에서 SQL Elapsed Time이 증가하였습니다.

Wait Time이 긴 SQL들을 확인하고 싶습니다.

Contents

Wait Time Analysis?

Wait Time Analysis의 활용

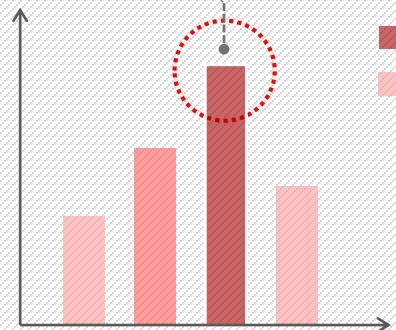
Case 1. Real-Time Monitor에서 SQL Elapsed Time이 증가하였습니다.

Elapsed Time = CPU Time + Wait Time 일때,
Wait Time 이 긴 SQL들을 확인하고 싶습니다.

Wait Time Analysis?

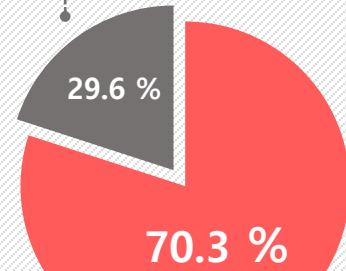
Wait Time Analysis은 언제 쓰나요?

Top SQL의 Elapsed Time(=wait)을
상세 분석하고,



User I/O
Concurrency

Safe Zone을 사용하여 SQL 투닝
대상을 선정할 때 사용합니다.

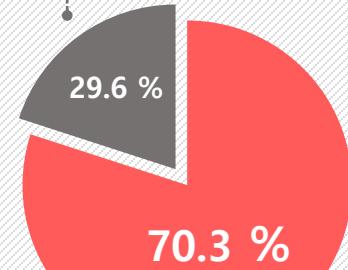


Wait Time Analysis은 언제 쓰나요?

Top SQL의 Elapsed Time(=cpu time + wait time)
을 상세 분석하고,



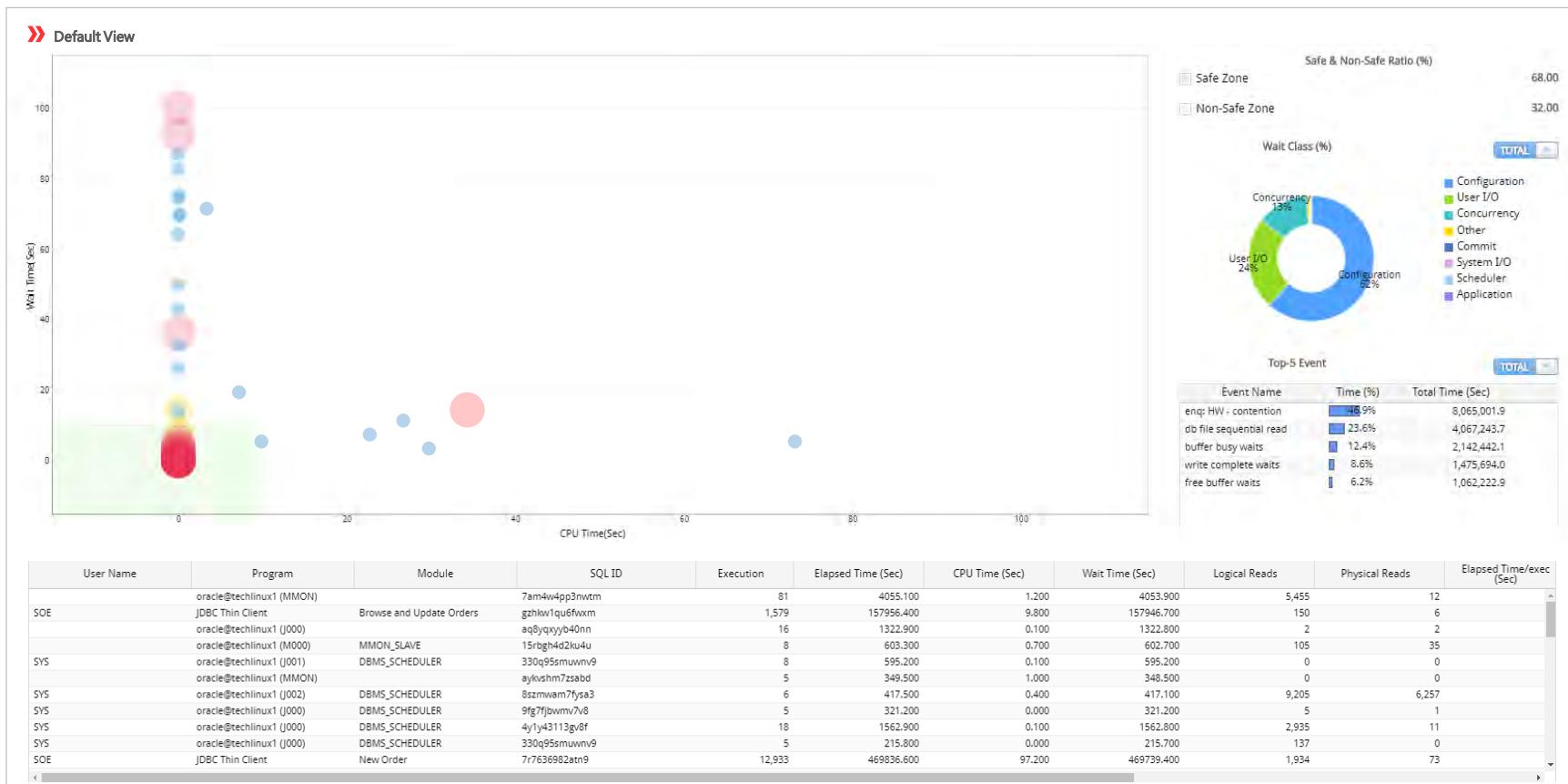
Safe Zone을 사용하여 SQL 투닝
대상을 선정할 때 사용합니다.



Wait Time Analysis?

Wait Time Analysis

- 선택한 날짜에 수행된 Top SQL에 대한 Wait Time & CPU Time을 비교·분석하는 기능입니다.
- SQL별 Wait Class & Top 5 Event 값을 확인할 수 있습니다.
- X축은 CPU Time, y축은 Wait Time으로 설정 화면에서 수행횟수에 따른 점의 크기를 변경할 수 있습니다.
- 해당 기능은 Performance Analyzer ▶ SQL Analysis ▶ Elapsed Time Analysis ▶ Wait Time Analysis 경로에서 사용 가능합니다.



Wait Time 분석 방법 비교

» Maxgauge 분석



03 SQL 선별 및 Wait Event 확인

» ORACLE AWR 분석



05 SQL 실행계획 및 Bind 수집



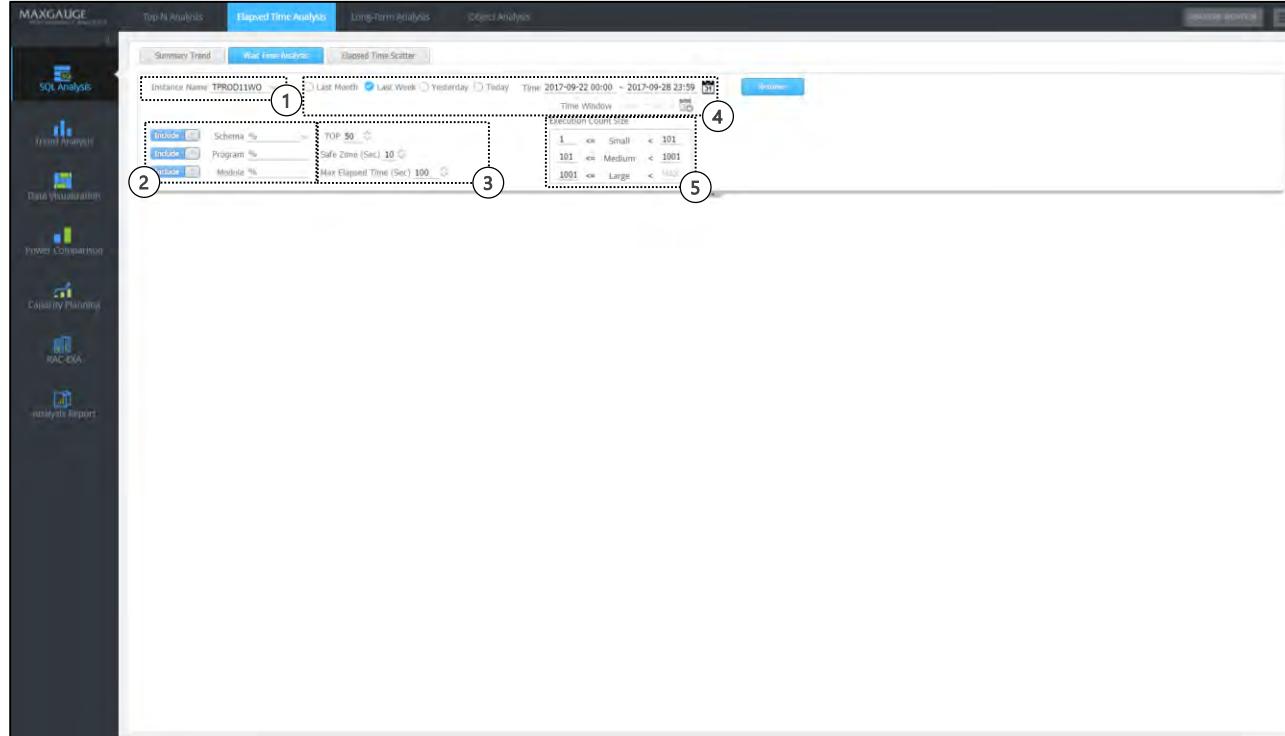
Wait Time Analysis의 활용

Wait Time Analysis의 사용 방법

Wait Time Analysis

- 검색 조건 설정
- 데이터 분석

» Wait Time Analysis 검색 화면



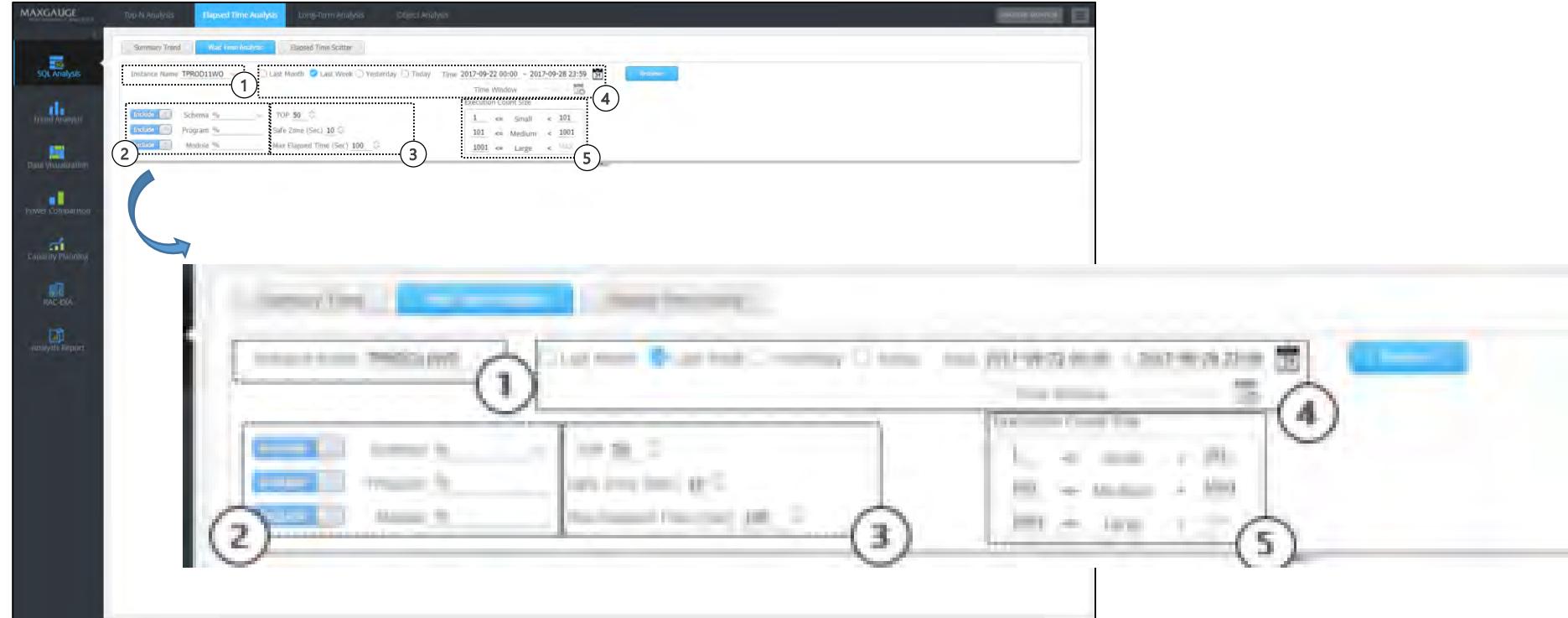
- ① 대상 Instance를 선택합니다.
- ② Schema, Program, Module명을 선택적으로 빼거나 추가할 수 있습니다.
- ③ 출력되는 SQL의 개수 선택 및 Safe Zone 영역 설정과 SQL의 Max 수행시간을 설정합니다.
※ Safe Zone : 안정적인 수행 구간이라 생각되는 영역을 사용자가 임의로 설정
- ④ 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택합니다.
- ⑤ SQL별 수행횟수에 따라 원의 크기를 설정할 수 있습니다. 위의 설정한 조건으로 Wait Time Analysis를 실행합니다.

Wait Time Analysis의 사용 방법

Wait Time Analysis

- 검색 조건 설정
- 데이터 분석

» WaitTimeAnalysis 검색 화면



- ① 대상 Instance를 선택합니다.
- ② Schema, Program, Module명을 선택적으로 추가할 수 있습니다.
- ③ 출력되는 SQL의 개수 선택 및 Safe Zone 영역 설정과 SQL의 Max 수행시간을 설정합니다.
※ Safe Zone : 안정적인 수행 구간이라 생각되는 영역을 사용자가 임의로 설정
- ④ 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택합니다.
- ⑤ SQL별 수행횟수에 따라 원의 크기를 설정할 수 있습니다. 위의 설정한 조건으로 Wait Time Analysis를 실행합니다.

Wait Time Analysis의 사용 방법

Wait Time Analysis

- 검색 조건 설정
- 데이터 분석

» Wait Time Analysis 조회 화면



- 1 x축은 CPU Time, y축은 Wait Time으로 개별 SQL마다 점으로 출력됩니다.
- 2 앞서 설정한 Safe Zone 영역에 포함된 SQL과 그렇지 않은 SQL의 비율 확인합니다.
- 3 Total 또는 개별 SQL의 Wait Class & Top 5 Event를 확인합니다.
- 4 개별 SQL의 일량 정보 확인합니다.
- 5 특정 SQL을 선택한 후, 하단에 위치한 Show Detail을 클릭합니다.

Wait Time Analysis의 사용 방법

Wait Time Analysis

- 검색 조건 설정
- 데이터 분석

» Wait Time Analysis 조회 화면

The screenshot shows the MAXGAUGE Performance Analyzer interface with the 'Wait Time Analysis' tab selected. The main area displays a table of wait statistics for sessions, including User Name, Program, Module, SQL ID, Execution, Elapsed Time (Sec), CPU Time (Sec), Wait Time (Sec), Logical Reads, Physical Reads, and Elapsed Time/exec (Sec). A specific session (SQL ID 1844266620) is highlighted. Below the main table, a modal window provides detailed information for this session, including the SQL Text & Execution Plan, Bind variables, and the Execution Plan itself.

User Name	Program	Module	SQL ID	Execution	Elapsed Time (Sec)	CPU Time (Sec)	Wait Time (Sec)	Logical Reads	Physical Reads	Elapsed Time/exec (Sec)
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	9f7fjpwv/v8	5	321.200	0.000	321.200	5	1	
SOE	JDBC Thin Client	New Order	gphw1qsfwem	13,523	1350008.600	90.000	1349918.600	1,809	47	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	4ylyk3113gdf	18	1562.900	0.100	1562.900	2,935	11	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	adlyppyy640n	16	122.300	0.100	1322.600	2	2	
SYS	oracle@technik1 (000)	MMACH_SLAVE	1qy0113gdf	8	403.300	0.700	403.300	165	35	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	350099mnuv9	8	595.200	0.100	595.200	0	0	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	aylykhn7xs8t	5	349.500	1.000	348.500	0	0	
SYS	oracle@technik1 (002)	DBMS_SCHEDULER	82zmvam7ysa3	6	417.500	0.400	417.100	9,205	6,257	
SOE	JDBC Thin Client	Browse Products	gphw1qsfwem	17,852	1638111.900	126.200	1638115.300	1,728	54	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	7am4q4p320nt	81	4055.100	1.200	4053.900	5,455	12	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	350099mnuv9	5	115.000	0.000	215.700	177	0	
SOE	JDBC Thin Client	New Order	771048981cans	13,244	480377.600	99.200	480378.600	1,961	74	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	63mtan2y9ic	23	604.500	0.100	604.500	1,622	138	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	66y1129x44	175	2336.700	2.200	2334.500	39,080	1,279	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	63tkn2y0plic	53	737.200	0.100	737.100	4,700	475	
SOE	JDBC Thin Client	Browse and Update Orders	gphw1qsfwem	1,625	161605.000	10.100	161595.000	150	6	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	82zmvam7ysa3	144	1194.500	10.400	1183.900	42,917	38,426	
SOE	JDBC Thin Client	Update Customer Details	gphw1qsfwem	196,079	987010.100	189.700	987420.500	3,646,231	1,003,660	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	63kmqdhrfg	66	312.200	0.100	312.200	1,924	55	
SOE	JDBC Thin Client	Update Customer Details	gphw1qsfwem	3,565	14224.300	27.600	14196.700	394	30	

1 선택된 SQL의 일별 SQL 성능 Sum 정보를 확인할 수 있습니다.
 2 선택된 SQL에 대한 SQL Text 및 Bind, Execution Plan 정보를 제공합니다.

Wait Time Analysis의 활용

Wait Time Analysis의 사용 방법

Wait Time Analysis

- 검색 조건 설정
- 데이터 분석

» WaitTimeAnalysis 조회 화면

MAXGAUGE
PERFORMANCE ANALYZER

Top-N Analysis Elapsed Time Analysis Long Term Analysis Object Analysis

Summary Trend Wait Time Analysis Elapsed Time Scater

Instance Name : TPROD11WO Time : 2017-09-22 00:00 ~ 2017-09-28 23:59

User Name	Program	Module	SQL ID	Execution	Elapsed Time (Sec)	CPU Time (Sec)	Wait Time (Sec)	Logical Reads	Physical Reads	Elapsed Time/exec (Sec)
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	9tg7jpwv7v8	5	321.200	0.000	321.200	5	1	62.57
SOE	JDBC Thin Client	New Order	gphnvlqsfwem	13,523	1350008.600	90.000	134918.600	1,809	47	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	4ylyk3113gdf	18	1562.900	0.100	1562.900	2,935	11	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	adlyppyy640n	16	122.300	0.100	1322.600	2	2	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	1m401l11111	8	403.300	0.700	403.300	165	35	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	3500979msuiv9	8	595.200	0.100	595.200	0	0	
SYS	oracle@technik1 (IMMON)	DBMS_SCHEDULER	aylykhn7ysx8	5	349.500	1.000	348.500	0	0	
SYS	oracle@technik1 (002)	DBMS_SCHEDULER	8izmzwam7ysa3	6	417.500	0.400	417.100	9,205	6,257	
SOE	JDBC Thin Client	Browse Products	gphnvlqsfwem	17,852	1638111.900	126.200	1638115.300	1,728	54	
SYS	oracle@technik1 (IMMON)	DBMS_SCHEDULER	7am4kq4g3n0t0	81	4055.100	1.200	4053.900	5,455	12	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	3500979msuiv9	5	119.000	0.000	215.700	177	0	
SOE	JDBC Thin Client	New Order	771548862ans	13,244	480377.600	99.200	480378.600	1,961	74	
SYS	oracle@technik1 (000)	DBMS_SCHEDULER	63M7an2jyslc	23	604.500	0.100	604.500	1,622	138	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	66y2129jx44	175	2336.700	2.200	2334.500	29,080	1,279	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	63f3k2en2j0plc	53	737.200	0.100	737.100	4,700	475	
SOE	JDBC Thin Client	Browse and Update Orders	gphnvlqsfwem	1,625	161605.000	10.100	161595.000	150	6	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	8izmzwam7ysa3	144	1194.500	10.400	1183.900	42,917	38,426	
SOE	JDBC Thin Client	Update Customer Details	gphnvlqsfwem	196,079	987010.100	189.700	987420.500	3,646,231	1,003,660	
SYS	oracle@technik1 (001)	DBMS_SCHEDULER	03mmpd0hrfg	66	312.200	0.100	312.200	1,924	55	
SOE	JDBC Thin Client	Update Customer Details	gphnvlqsfwem	3,565	1424.300	27.600	14196.700	394	30	

SQL Text & Execution Plan [500.l0.apj(y3113gdf)] Hide Details

1

2

- 선택된 SQL의 일별 성능 정보를 확인할 수 있습니다.
- 선택된 SQL에 대한 Text 및 Bind, Execution Plan 정보를 확인할 수 있습니다.

실전 분석 사례 Case 1.
Real Time-Monitor에서 SQL Elapsed Time이 급증하였습니다.
Wait Time이 긴 SQL들을 확인하고 싶습니다.

Wait Time Analysis의 활용

STEP

1. RTM – Wait Time 급증 감지

2. 검색 조건 설정

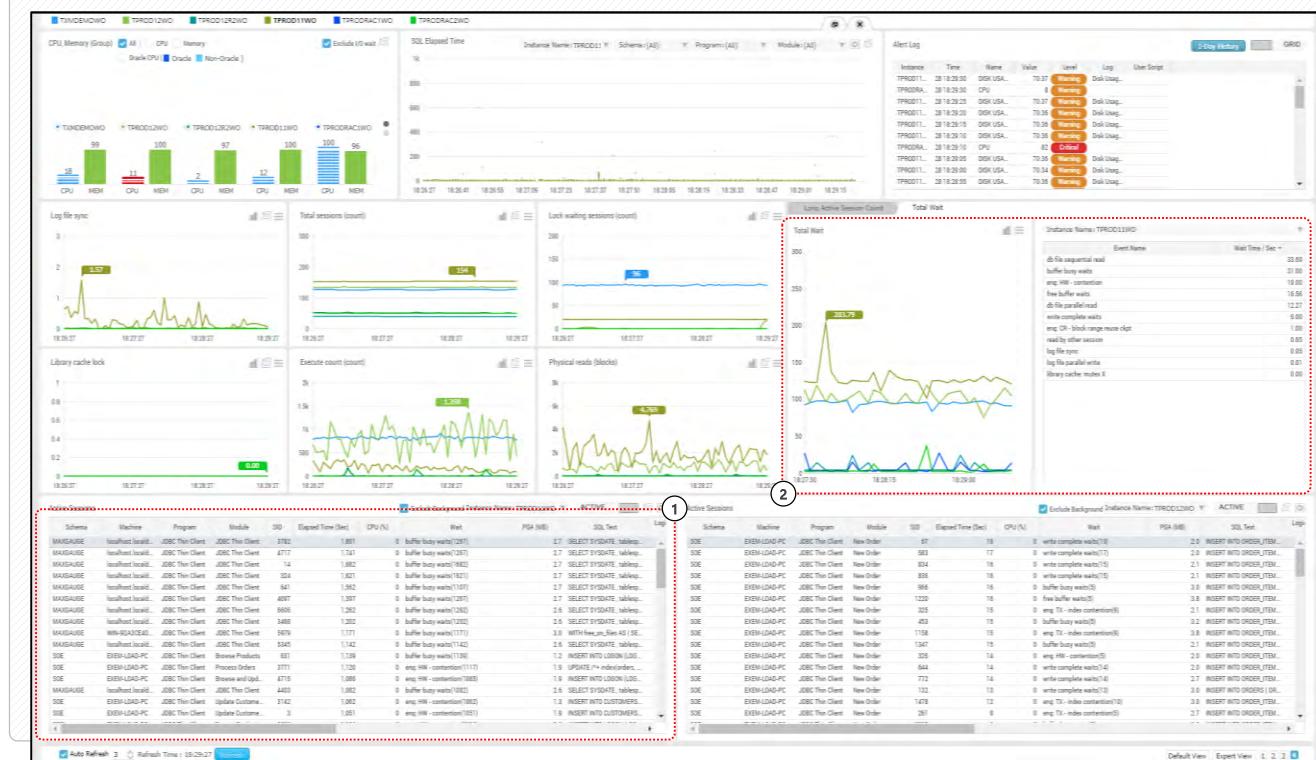
3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 주이 확인

└ Long-Term Analysis 연계

"Wait Time이 긴 SQL들을 확인하고 싶습니다."

» Real Time Monitor 화면



1 Active Session Grid에서 Wait Event가 증가한 것을 확인할 수 있습니다.

2 Total Wait Grid에서 Total Wait이 증가한 것을 확인할 수 있습니다.

✓ 장애발생 감지

RTM – 개별 Session의 Wait Event를 대기하면서, Total Wait Time의 급증 확인됨

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

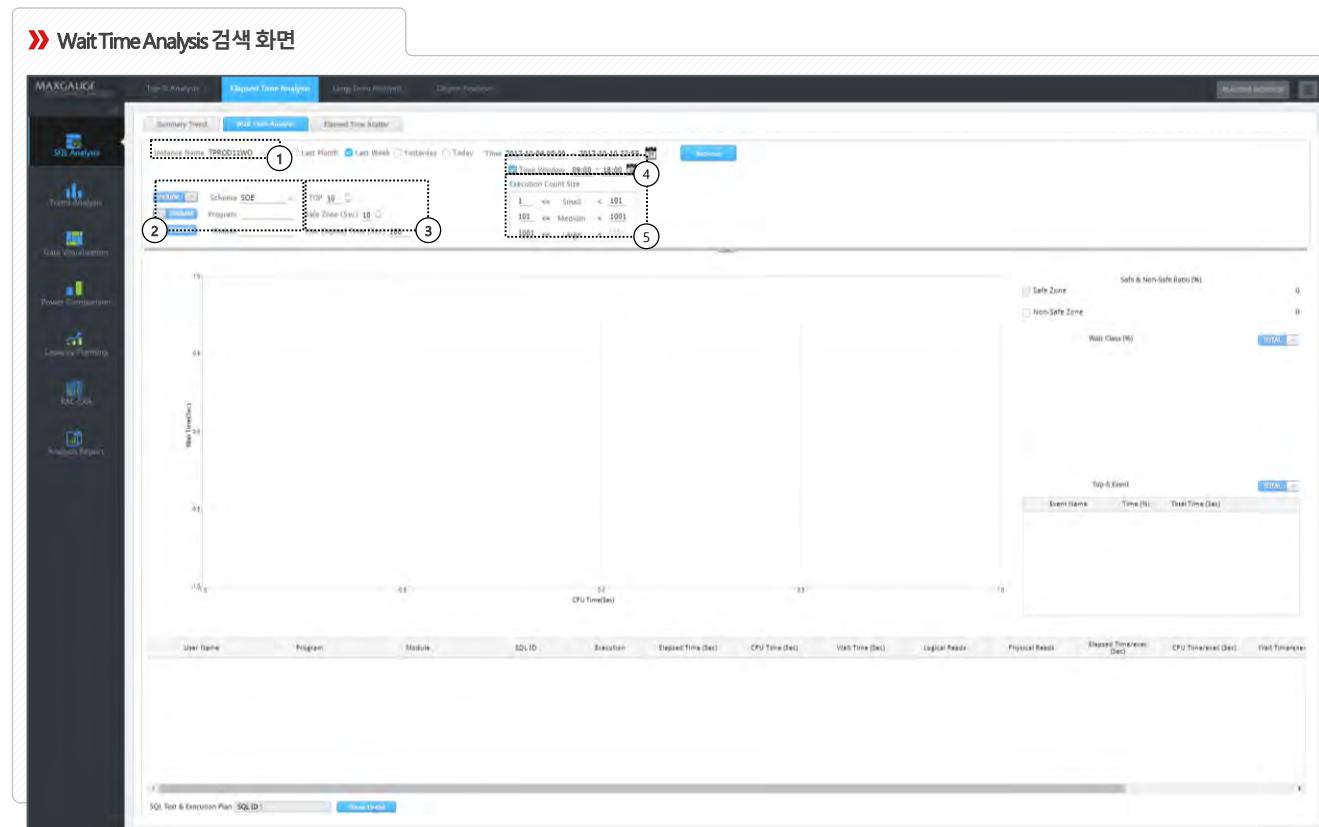
STEP

1. RTM – Wait Time 급증 감지

2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 주이 확인
 - └ Long-Term Analysis 연계



- 1 대상 Instance (TPROD11WO)를 선택합니다.
- 2 Schema (SOE), Program, Module명을 선택적으로 빼거나 추가할 수 있습니다.
- 3 수행시간이 100초 이하의 SQL 중 Top-10 SQL만 출력합니다.
- 4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (09:00 ~ 18:00)을 선택적으로 선택할 수 있습니다.
- 5 SQL별 수행횟수에 따라 보여지는 원의 크기를 설정할 수 있습니다. 위의 설정한 조건으로 Wait Time Analysis를 실행합니다.

✓ 시나리오

INSTANCE	: TPROD11WO
SCHEMA	: SOE
PROGRAM	: (null)
분석 상세 일자	: 9/10 ~ 10/10

✓ 분석 방향 및 의도

지난 한달 간, Wait Time이 긴 SQL을 대상으로 SQL 일량 분석 및 Wait Event를 분석하고 싶음

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

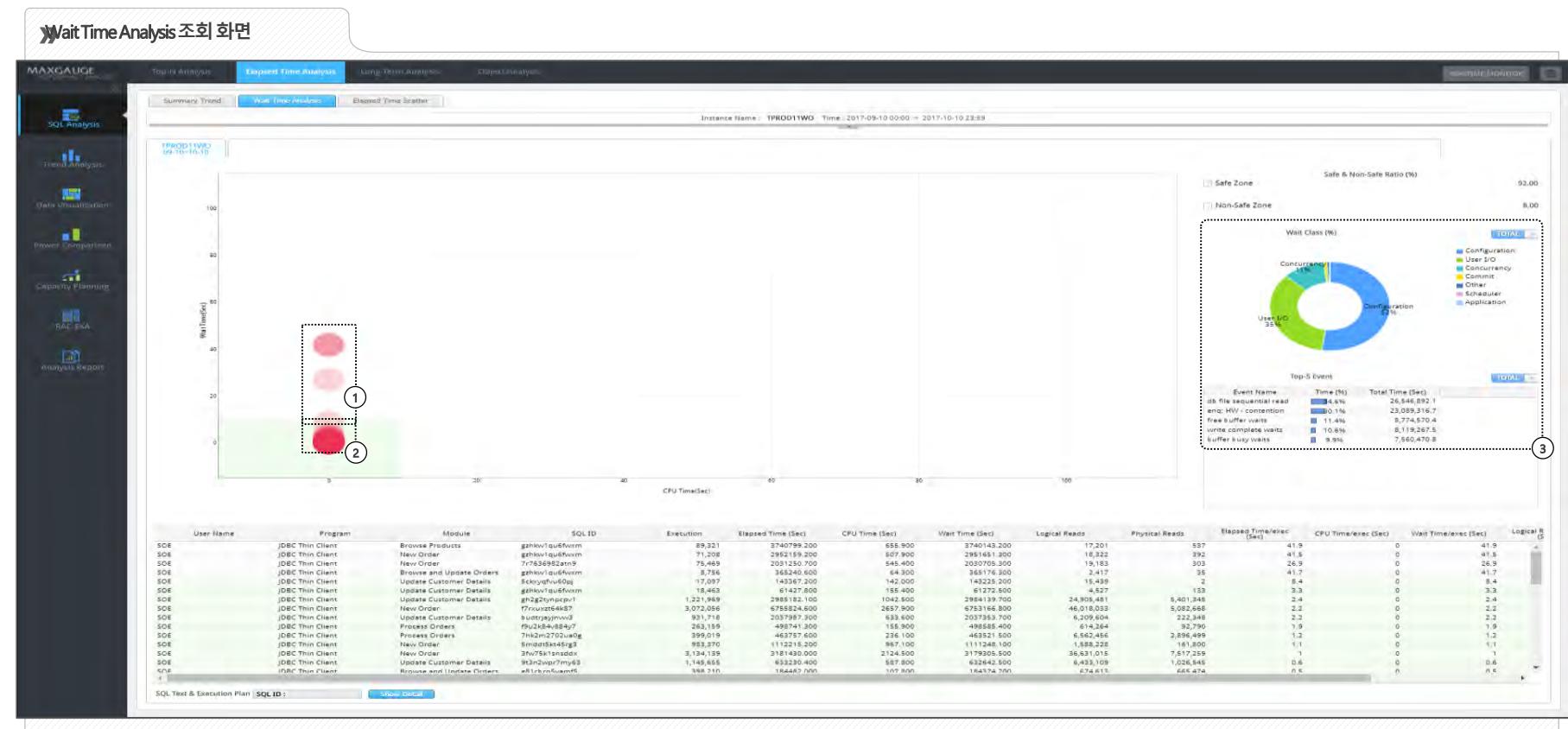
STEP

- RTM – Wait Time 급증 감지
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 주이 확인

↳ Long-Term Analysis 연계



- 1 Wait Time 30초 이상인 SQL을 차트에서 확인할 수 있습니다. (4건)
- 2 Safe Zone에 위치한 SQL은 Wait Time이 짧아 성능상 문제가 없습니다. (6건)
- 3 추출된 Top-10 SQL에 대한 Total Wait Class와 Top 5-Event를 확인할 수 있습니다.

Wait Time Analysis의 활용

"Wait Time이 긴 SQL들을 확인하고 싶습니다."

STEP

- RTM – Wait Time 급증 감지
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석**
- 1-SQL Detail 정보 확인
- SQL 시간대별 주이 확인
 - Long-Term Analysis 연계



- Wait Time 30초 이상인 SQL들을 마우스로 드래그합니다.
- 추출된 4건의 SQL에 대한 Selected Wait Class와 Top 5-Event를 확인할 수 있습니다.
- 추출된 4건의 SQL에 대한 개별 SQL의 일량 정보 확인할 수 있습니다.
- 1-SQL을 선택한 다음, Show Detail을 선택합니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

STEP

1. RTM – Wait Time 급증 감지
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인

↳ Long-Term Analysis 연계

The screenshot shows the MAXGAUGE Wait Time Analysis software interface. The main window displays a table of SQL statements (SOE) with columns including User Name, Program, Module, SQL ID, Execution, Elapsed Time (Sec), CPU Time (Sec), Wait Time (Sec), Logical Reads, Physical Reads, Elapsed Time/exec, CPU Time/exec (Sec), Wait Time/exec (Sec), and Logical R/O. Below this, a detailed view for SQL ID 3241608609 shows the SQL Text & Execution Plan, including the execution history over time, the execution plan, and the bind values.

User Name	Program	Module	SQL ID	Execution	Elapsed Time (Sec)	CPU Time (Sec)	Wait Time (Sec)	Logical Reads	Physical Reads	Elapsed Time/exec	CPU Time/exec (Sec)	Wait Time/exec (Sec)	Logical R/O
SOE	JDBC Thin Client	Browse Products	gthkv1qu6wem	395	3740798.200	656.800	3740798.200	77.200	537	41.9	0	41.9	
SOE	JDBC Thin Client	New Order	gthkv1qu6wem	4,856	2952159.200	64.200	2952159.200	2,177	0	41.7	0	41.7	
SOE	JDBC Thin Client	New Order	77636952am9	71,208	2031250.700	507.900	2031250.700	18,322	392	41.5	0	41.5	
SOE	JDBC Thin Client	Update Customer Details	scixyqvvu6opj	75,469	2030708.300	548.400	2030708.300	19,183	303	26.9	0	26.9	
SOE	JDBC Thin Client	Update Customer Details	scixyqvvu6opj	17,097	143267.200	142.000	143267.200	16,439	2	8.4	0	8.4	

✓ 해당 SQL 일량 정보 및 SQL Plan과 Bind 정보를 확인할 수 있습니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

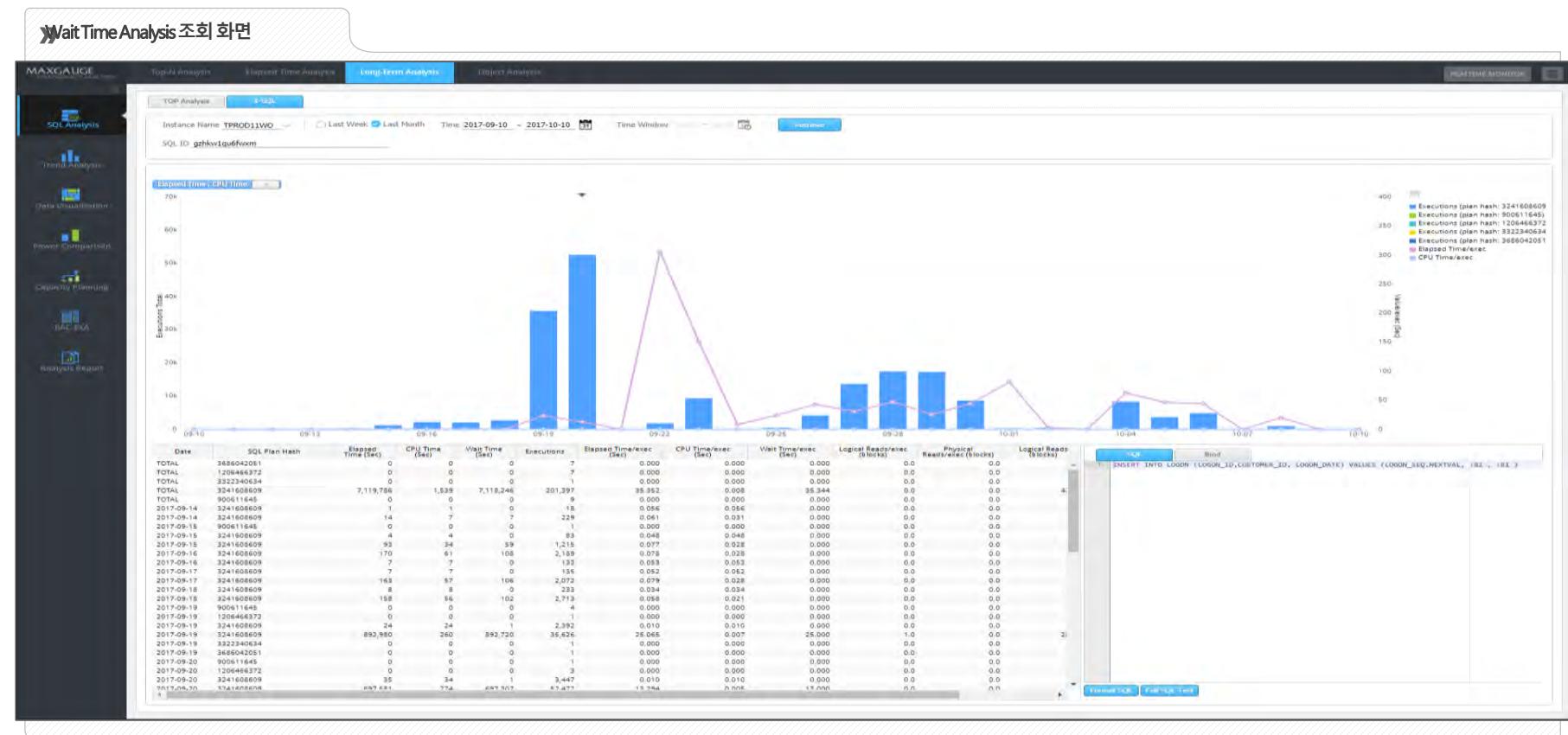
STEP

1. RTM – Wait Time 급증 감지
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인

└ Long-Term Analysis 연계



✓ 문제가 되는 SQL ID를 복사하여, Long-Term Analysis 탭으로 이동합니다.

✓ 해당 SQL의 보다 상세한 일량 정보와 추이를 확인할 수 있습니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

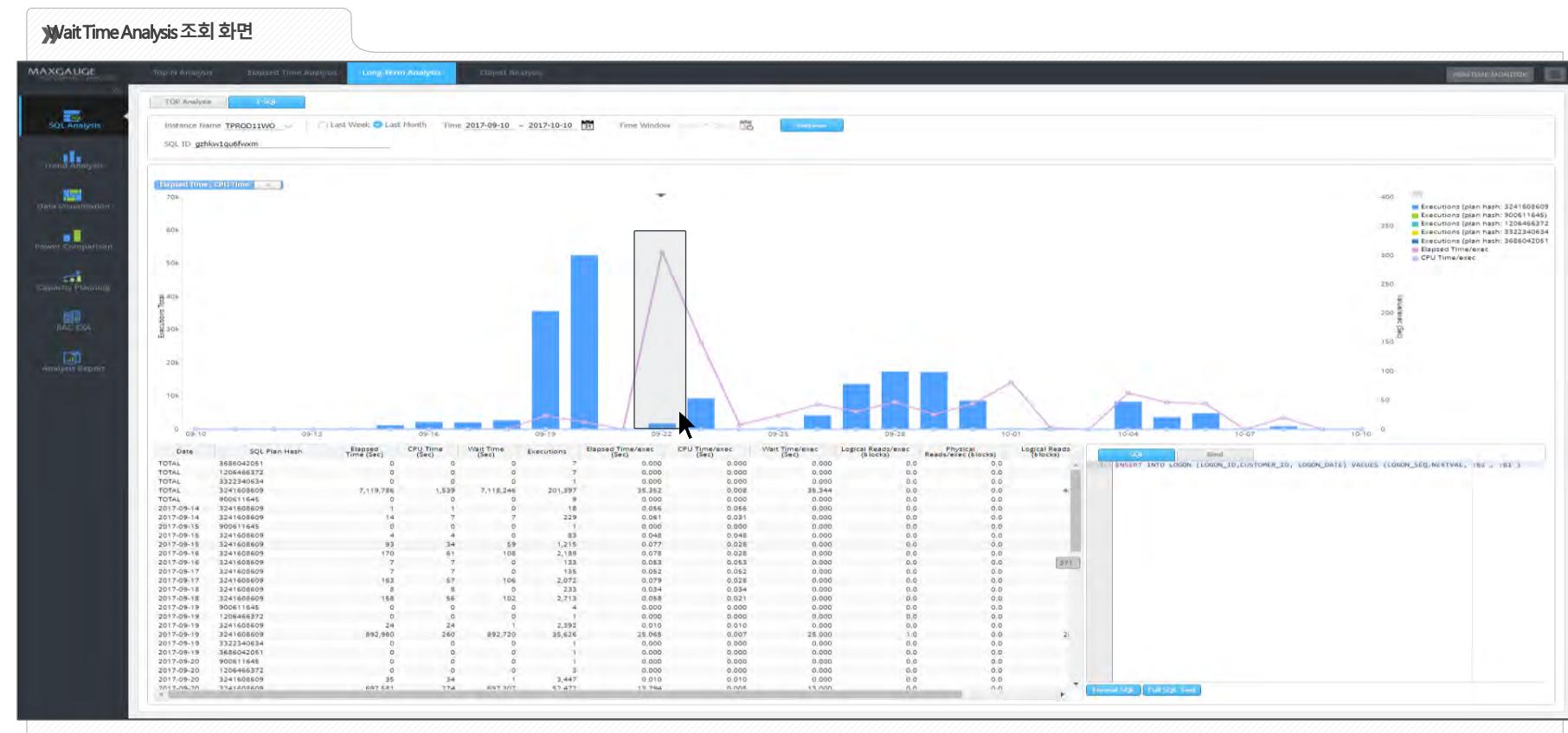
STEP

1. RTM – Wait Time 급증 감지
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인

└ Long-Term Analysis 연계



✓ Elapsed Time(분홍 실선)이 가장 컸던 9/22일을 상세 분석하기 위해 더블클릭 합니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

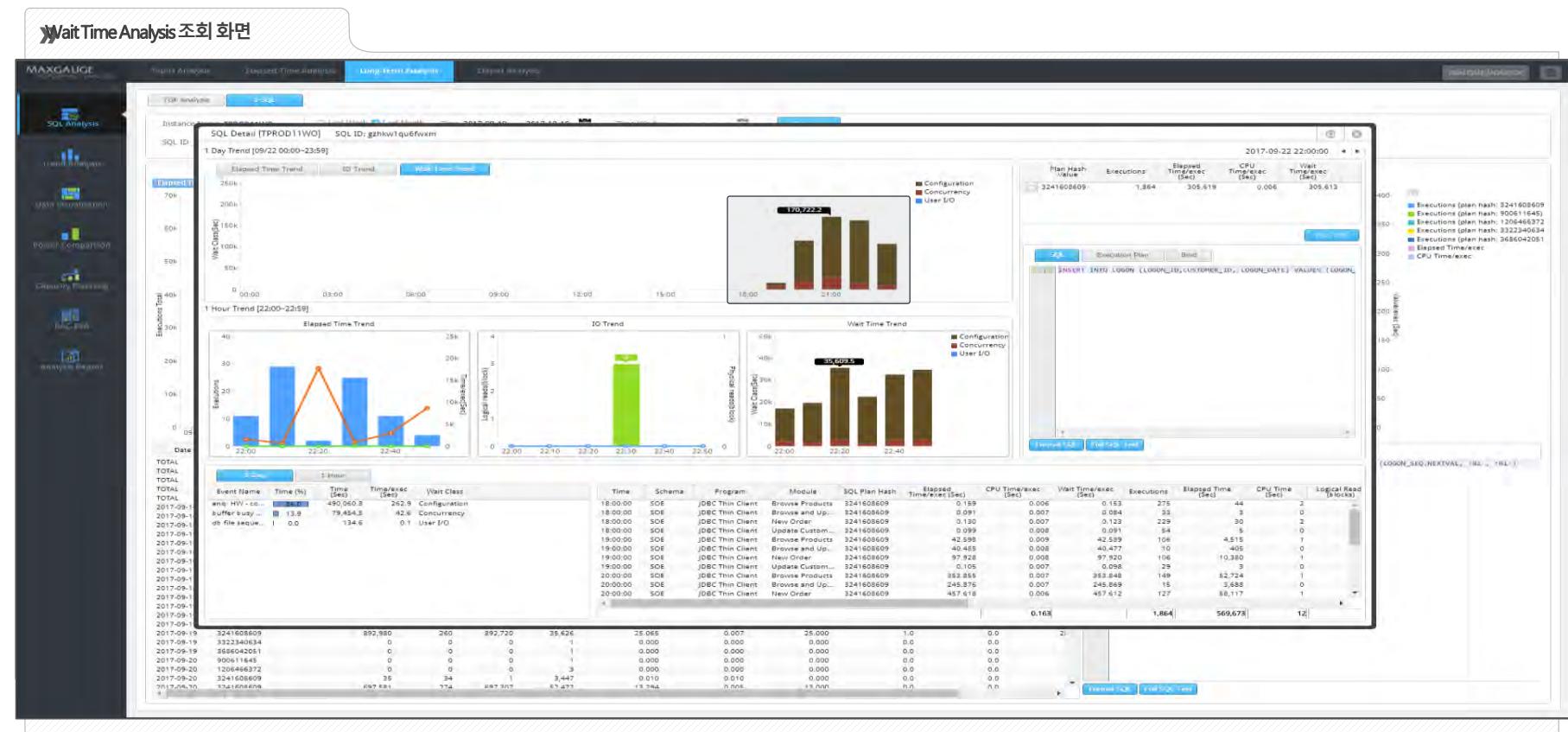
STEP

- RTM – Wait Time 급증 감지
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인**

└ Long-Term Analysis 연계



- ✓ SQL Detail을 통해 선택한 날짜에 SQL의 상세한 정보를 확인할 수 있습니다.
- ✓ ‘Wait Time Trend Tab’을 선택하면 시간대별 주요 대기 Class와 Event를 확인할 수 있습니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

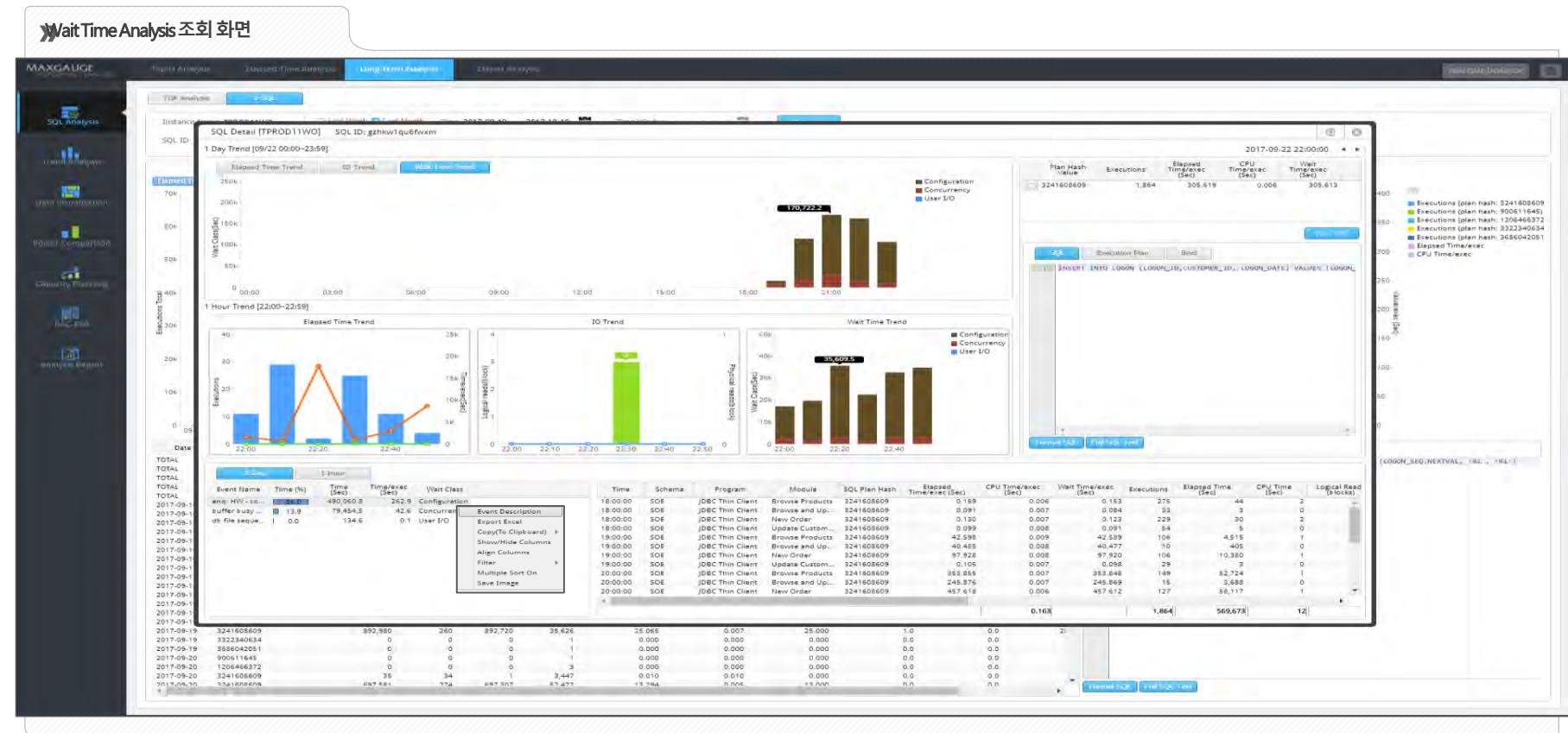
STEP

1. RTM – Wait Time 급증 감지
2. 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인

└ Long-Term Analysis 연계



- ✓ “enq : HW – contention” Event를 대기하는 시간이 긴 것을 알 수 있습니다.
- ✓ 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 Event Description으로 연계합니다.

Wait Time Analysis의 활용

“Wait Time이 긴 SQL들을 확인하고 싶습니다.”

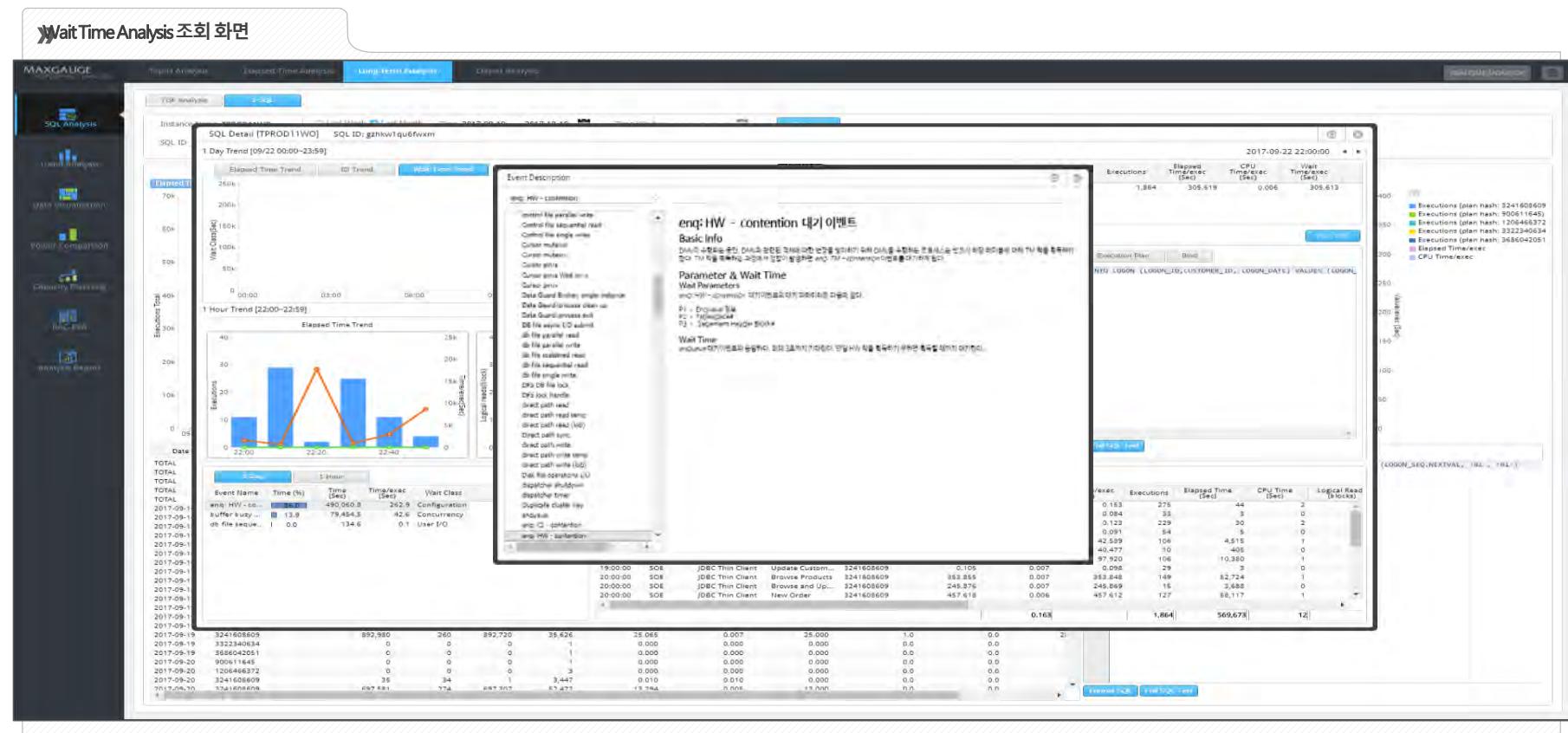
STEP

- RTM – Wait Time 급증 감지
- 검색 조건 설정

3. 데이터 분석

- 차트 확인 및 설명
- Wait Time 긴 SQL 분석
- 1-SQL Detail 정보 확인
- SQL 시간대별 추이 확인**

└ Long-Term Analysis 연계



✓ Event Help Description 기능을 통해 해당 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

MAXGAUGE



Practical Guide

Trend Analysis [Performance Analyzer]

Contents

Trend Analysis ?

Trend Analysis의 활용

Performance Trend의 사용 방법

EXA_RAC Trend의 사용 방법

Long Term Trend의 사용 방법

Case 1. CPU가 급격히 증가한 원인을 알고 싶어요

Case 2. RAC 간의 Load Balance 및 성능을 확인하고 싶어요

Case 3. 한 달간 가장 큰 부하의 원인을 알고 싶어요

Trend Analysis?

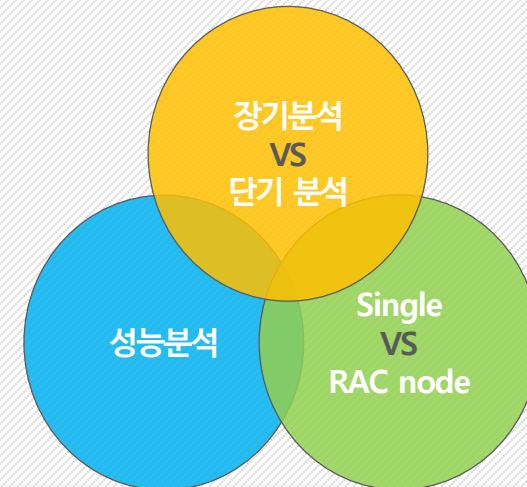
Performance Trend

Trend Analysis는 언제 사용하나요?

DB 서버부하 시점의
원인 분석을 해야 하는 경우 «

RAC node간의
성능 분석을 해야 하는 경우 «

» 장기간 DB 성능 추이를 확인 하고 싶은
경우



Performance Trend ?

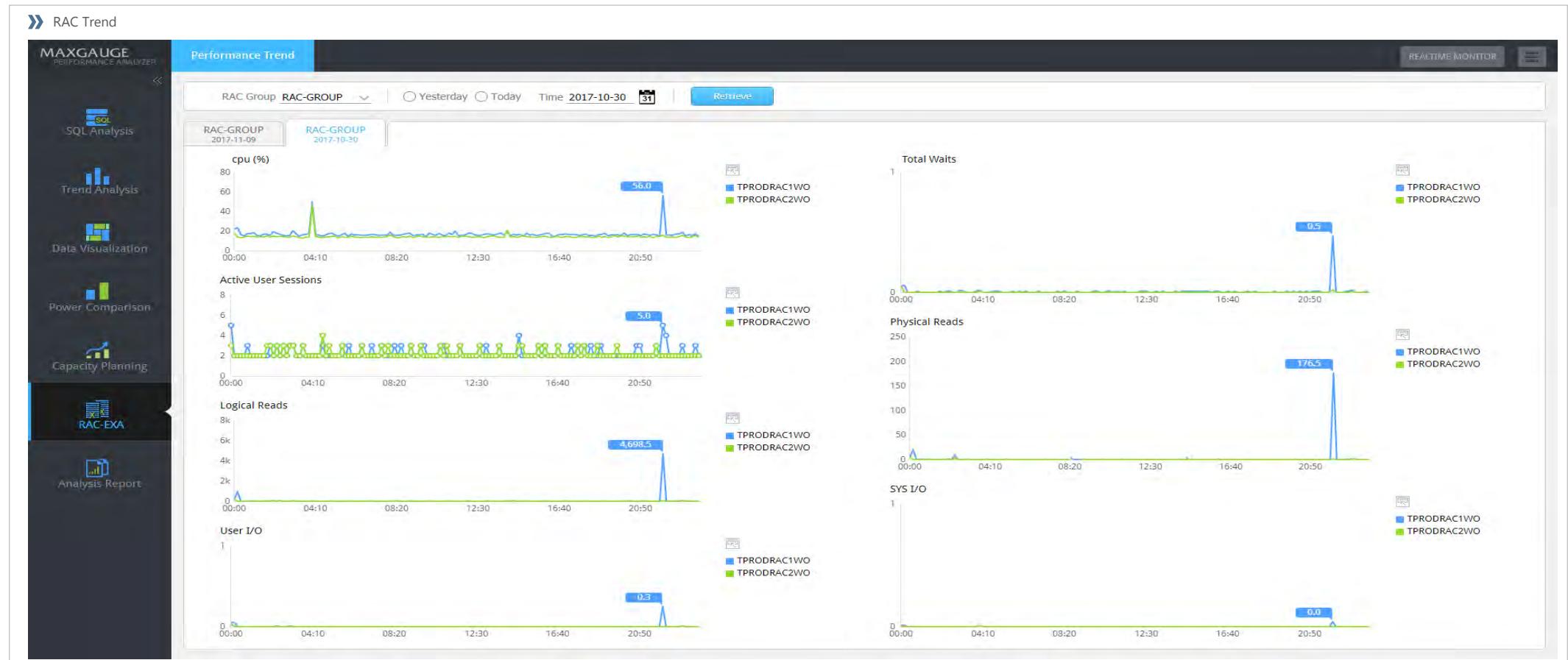
- 시간대별 성능 지표 제공을 통해 부하 시점에 대해 직관적인 파악이 가능하다. (시간대별 성능 지표 비교를 통해 급격한 지표 변동을 통해 부하시점을 한눈에 파악가능)
- 시스템의 성능 부하 시점 파악은 물론이고, 부하 발생 시점의 상세 분석까지 가능하다
- 즉 분 단위 성능 지표와 초 단위 active session 정보를 통해, 당시 DB 부하의 원인과 SQL문 까지도 파악이 가능하다.
- 해당 기능은 Performance Analyzer ▶ Trend Analysis ▶ Performance Trend 에서 사용 할 수 있다.



Trend Analysis?

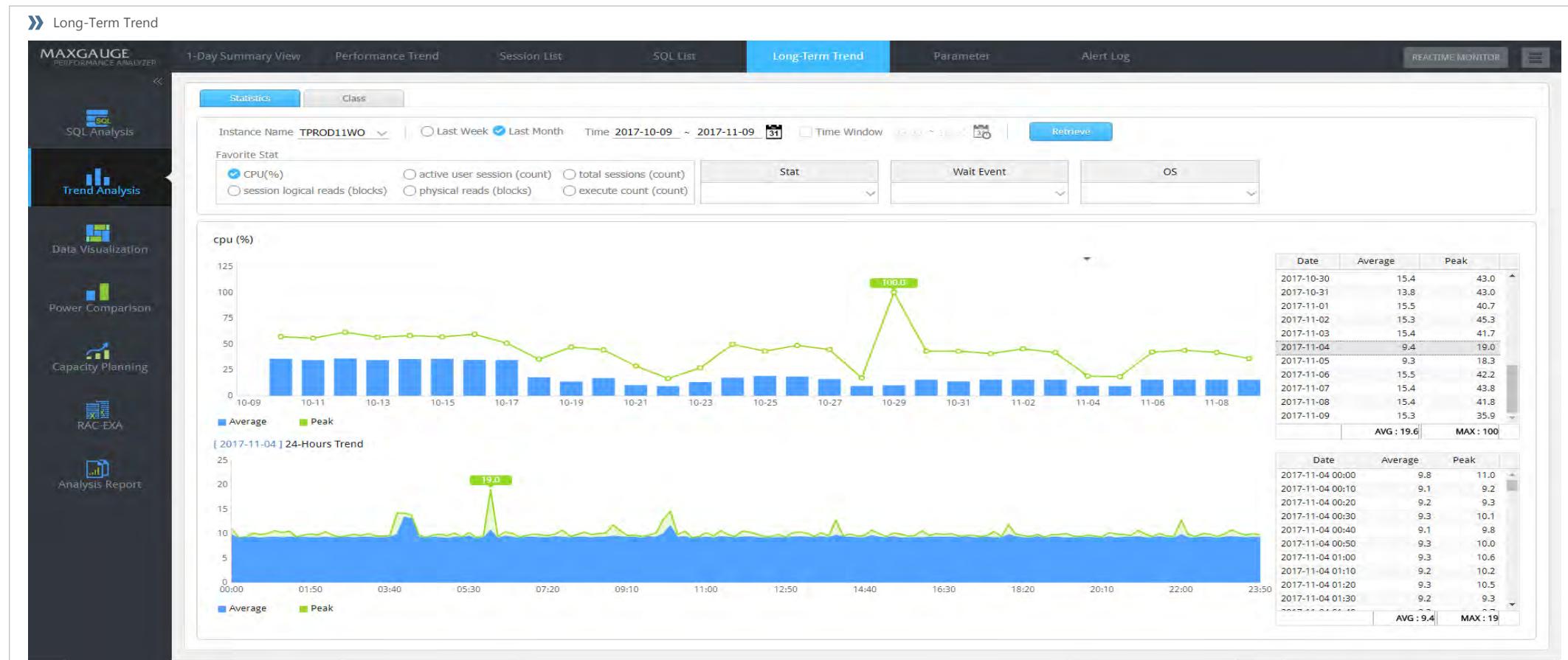
RAC Trend ?

- RAC로 운영 중인 서버간에 성능 지표를 보다 직관적으로 비교 할 수 있다.
- 상세분석 연동을 통해 동일 시간 때 RAC 노드의 성능 지표, Active Session 정보를 비교 할 수 있다.
- 해당 기능은 Performance Analyzer ▶ RAC-EXA ▶ Performance Trend에서 사용 할 수 있다.



Long-Term Trend ?

- 선택한 지표의 일별 성능 추이가 제공되어, 장기간의 자원 사용량 추이를 확인할 경우 용이합니다. (주 단위, 월 단위 Trend 비교를 통해 대상 서버의 자원 사용 파악이 가능 함)
- 부하가 높은 날자 선택 시, 하단에 있는 24 Hours Trend를 통해 하루 동안의 성능 추이를 확인 할 수 있다. (10분 단위로 하루 동안의 성능 지표의 avg 값, max 값 제공)
- 24시간 중 성능이 올라간 시간대를 선택하고 드래그 하면 Time Slice 기능과 연동되어 DB의 부하 원인은 물론 유발한 SQL까지 알 수 있다.
- 해당 기능은 Performance Analyzer ▶ Trend Analysis ▶ Long-Term Trend에서 사용 할 수 있다.



Trend 비교를 통해 DB 성능 분석을 할 수 있습니다!



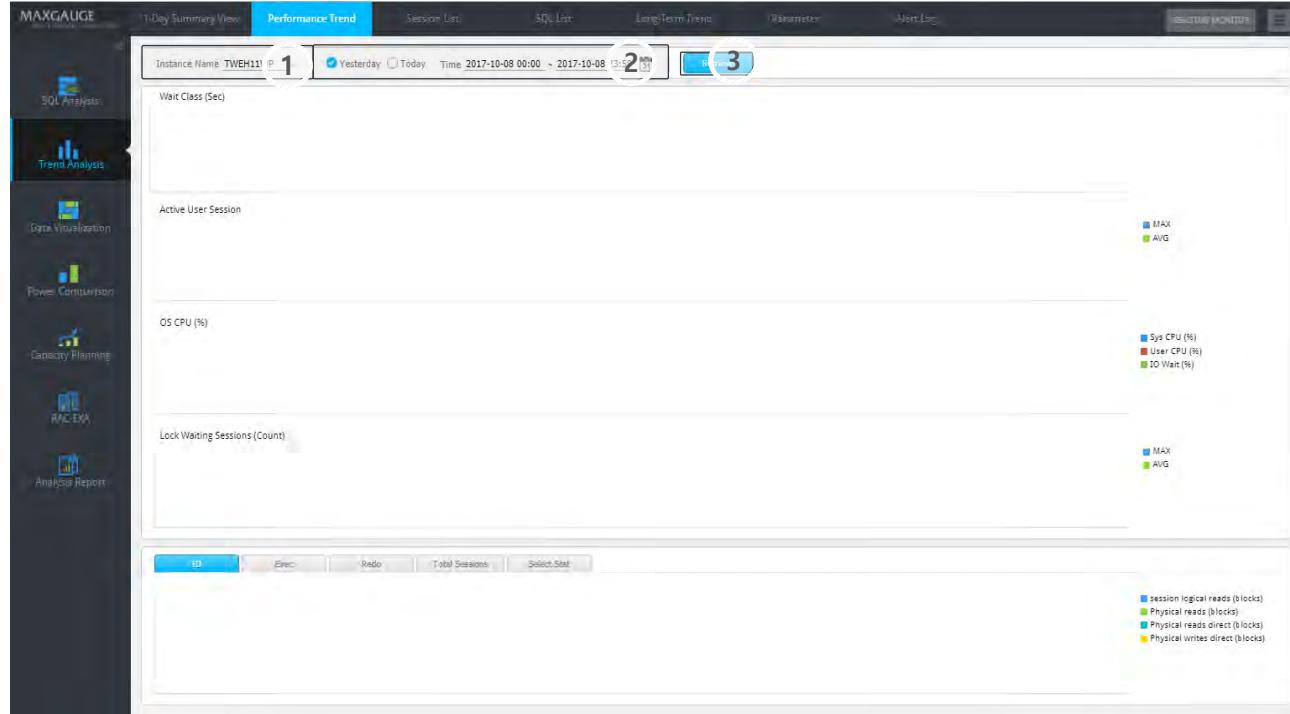
Trend Analysis의 활용

Performance Trend 의 사용 방법

Performance Trend

- 검색 조건 설정
- 데이터 분석

» Performance Trend 조회 화면



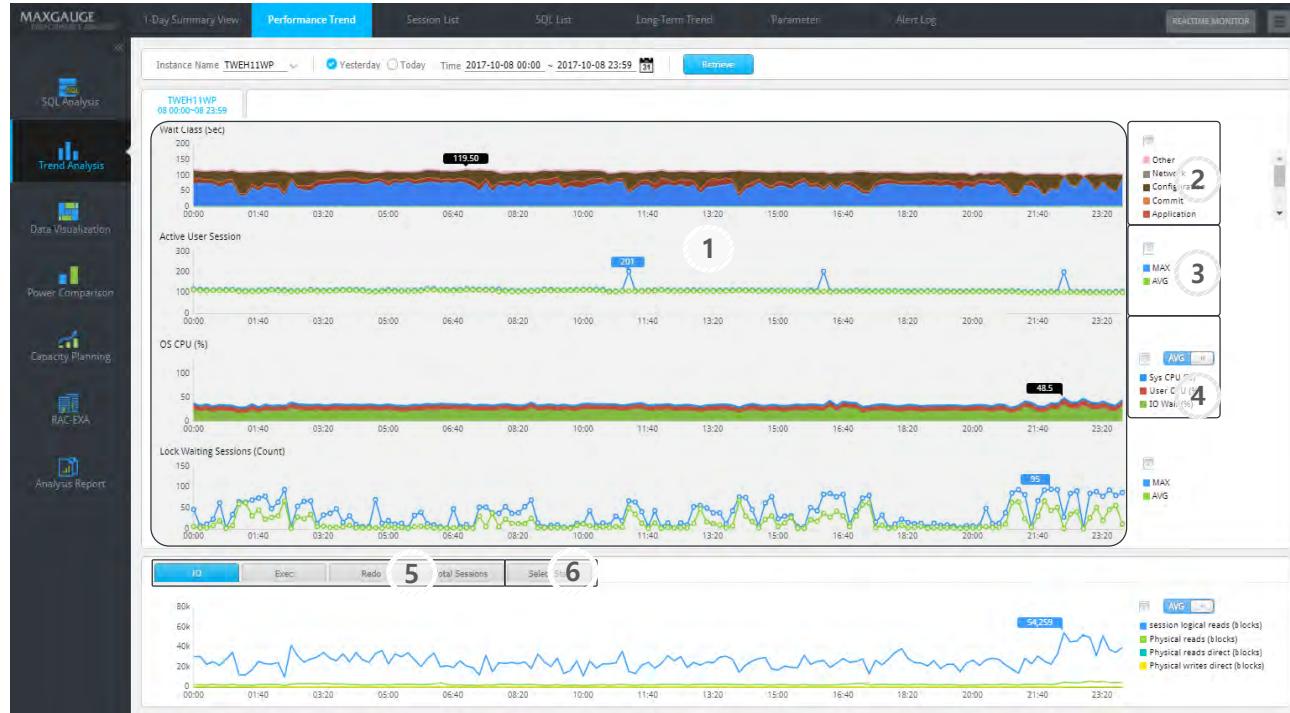
- ① 성능 분석을 원하는 인스턴스를 선택 합니다.
- ② 분석을 원하는 날짜를 선택 합니다. (Performance Trend는 한 서버에 대해 **최대 2일**에 대한 성능 비교가 가능 합니다.)
- ③ 위의 설정한 조건으로 **Performance Trend**를 실행합니다.

Performance Trend 의 사용 방법

Performance Trend

- 검색 조건 설정
- 데이터 분석

» Performance Trend 검색 화면



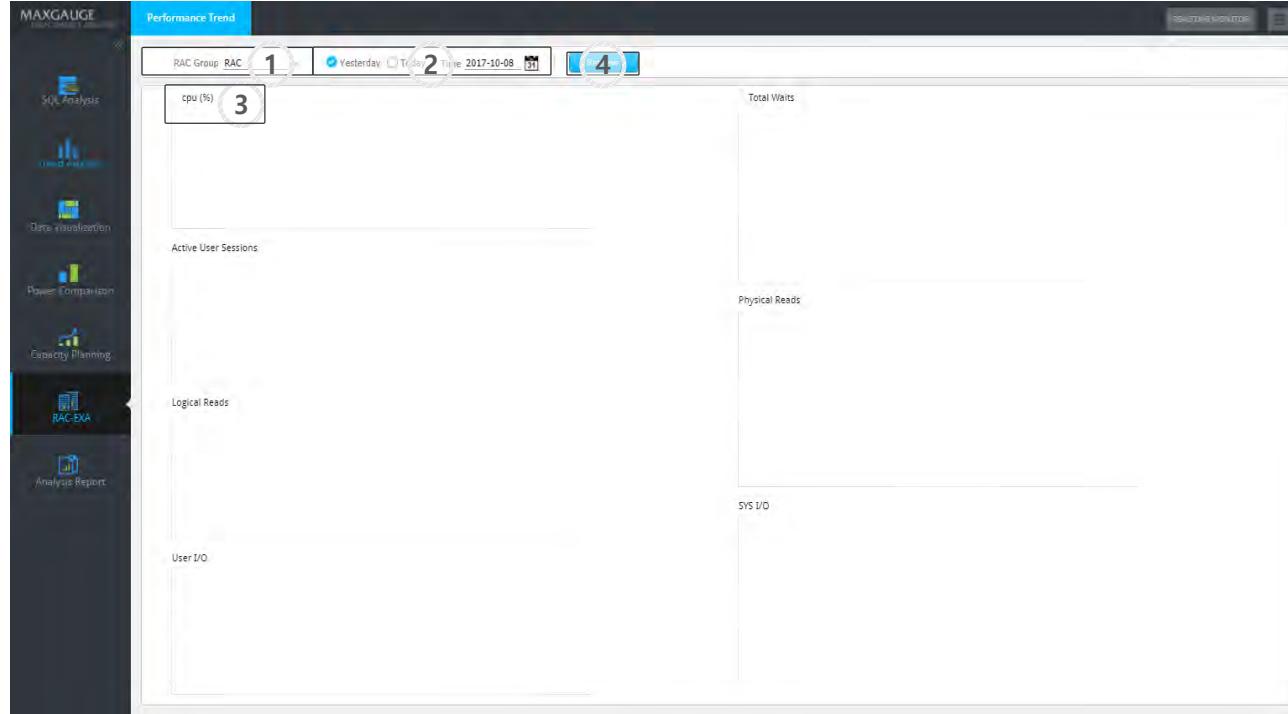
- 선택한 날짜의 성능 지표 Wait Class, Active User Session, On CPU, Lock Waiting Session(Count)가 출력 됩니다.
- 팝업 메뉴 선택 시 Wait Class의 10분 단위 AVG 값들을 확인 할 수 있습니다.
- 팝업 메뉴 선택 시 Active Session의 10분 단위 AVG, MAX 값을 확인 할 수 있습니다.
- Toggle 버튼 선택을 통해 SYS CPU, User CPU, IO Wait의 MAX 값을 확인 할 수 있습니다.
- 하단에 IO, Execute Count, Redo, Total Session 지표를 제공합니다.
- 기본 화면 지표 외에 Select Stat 선택을 통해 비교하고자 하는 성능 지표를 선택 할 수 있습니다.

RAC-EXA Trend 의 사용 방법

RAC-EXA Trend

- 검색 조건 설정
- 데이터 분석

» RAC-EXA Trend 검색 화면



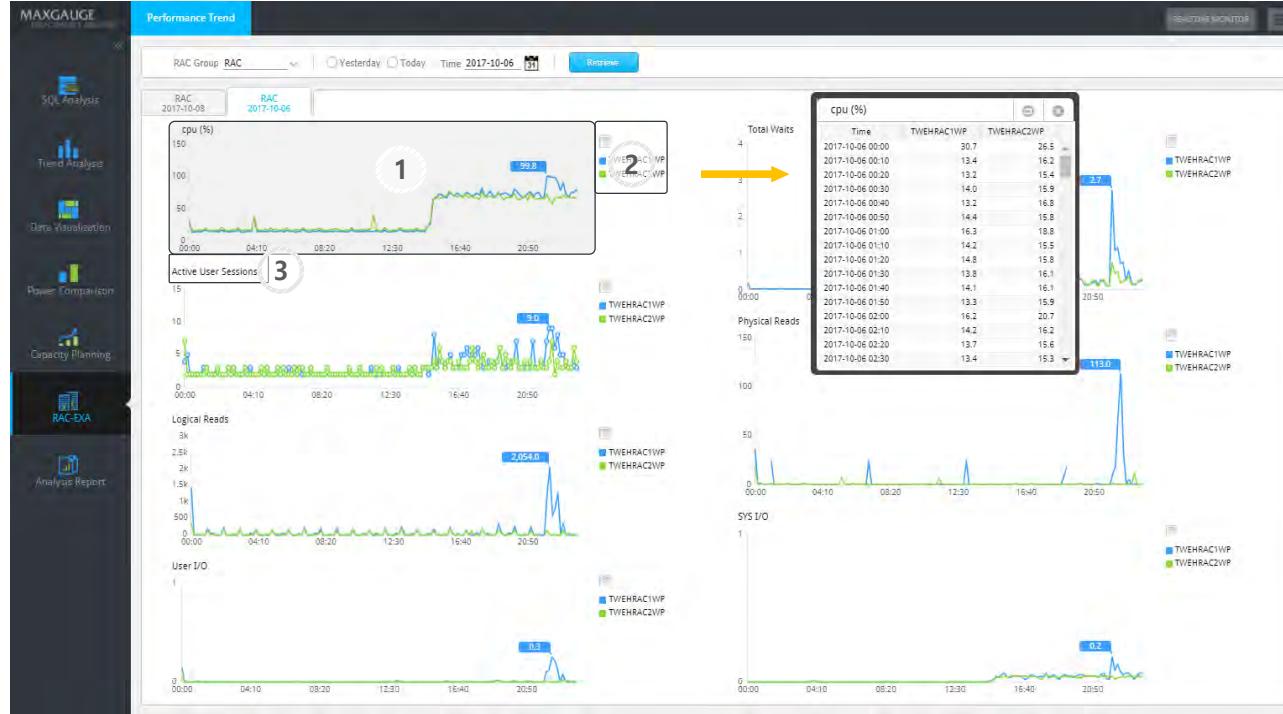
- 1 분석을 원하는 RAC 그룹을 선택합니다.
- 2 분석을 원하는 날짜를 선택 합니다. (달력 선택을 통해 데이터가 쌓이는 모든 날짜의 RAC 그룹 비교가 가능 합니다.)
- 3 성능 지표를 더블 클릭 시 (추천 지표와 DB stat , Wait Event, OS stat의 비교분석이 가능합니다.)
- 4 위의 설정한 조건으로 RAC-EXA Trend를 실행합니다.

RAC-EXA Trend 의 사용 방법

RAC-EXA Trend

- 검색 조건 설정
- 데이터 분석

» RAC-EXA Trend 검색 화면



1 RAC 그룹의 성능 지표 (CPU, Active User Session, Logical Reads , User I/O, Total Waits, Physical Reads , Sys I/O) 가 출력 됩니다.

2 팝업 아이콘 선택 시 성능 지표의 AVG 값이 리스트 형태로 출력 됩니다.

3 다른 지표를 확인하고 싶은 경우 지표 명을 더블 클릭을 하면, 원하는 성능 지표로 변경이 가능 합니다.

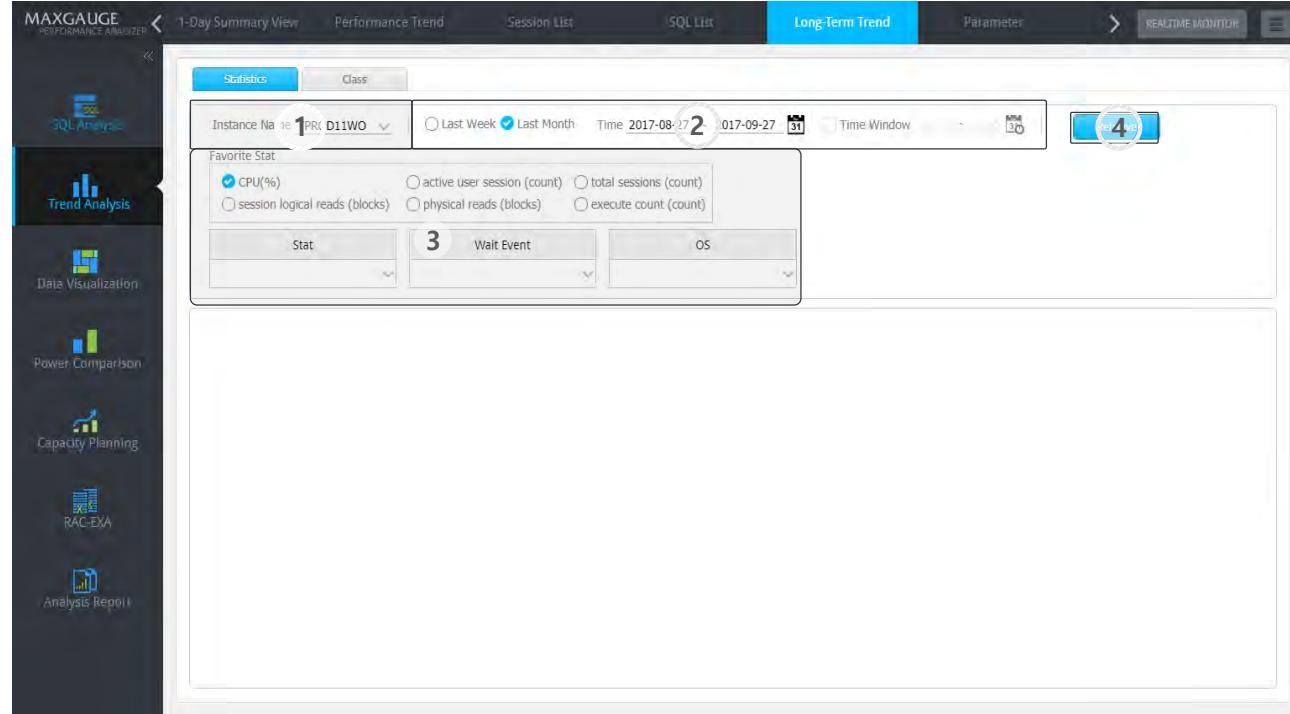
Cf) RAC 그룹의 성능 지표 비교를 통해 각 노드의 load Balancing 여부를 확인 하실 수 있습니다.

Long-Term Trend 의 사용 방법

Long-Term Trend

- 검색 조건 설정
- 데이터 분석

» Long-Term Trend 검색 화면



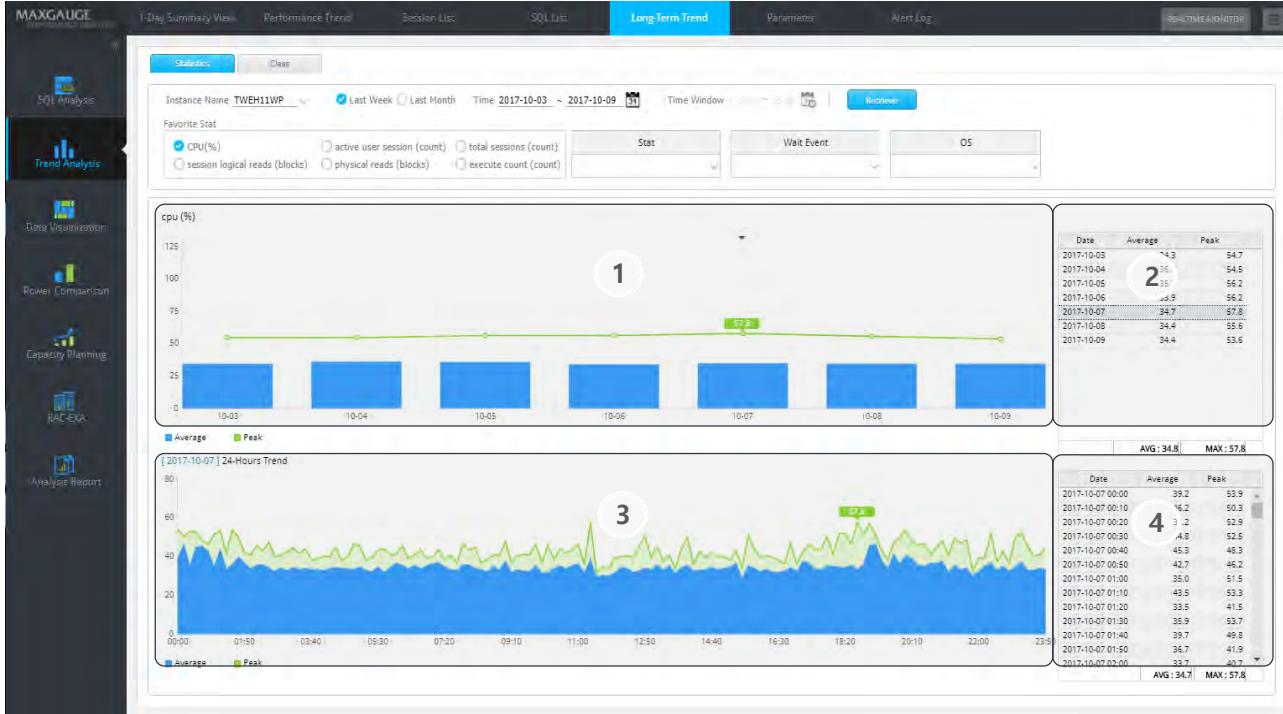
- 1 분석 대상 Instance를 선택합니다.
- 2 분석을 하고자 하는 기간 설정을 합니다. (Time Window 설정을 통해 업무 시간의 분석이 가능 합니다.)
- 3 분석을 원하는 지표를 선택합니다. (추천 지표 외에도 DB stat, Wait Event, OS stat의 분석이 가능합니다.)
- 4 위의 설정한 조건으로 Long-Term Trend를 실행합니다.

Long-Term Trend 의 사용 방법

Long-Term Trend

- 검색 조건 설정
- 데이터 분석

» Long-Term Trend 조회 화면



- 분석을 원하는 지표의 일별 MAX, AVG 값이 **그래프 형태**로 출력 됩니다. (최초 출력 일자는 지표의 MAX값을 기록한 날짜가 출력 됩니다.)
- 분석을 원하는 지표의 일별 MAX, AVG 값이 **리스트 형태**로 출력 됩니다.
- 선택된 날짜의 일일 지표 추이가 **그래프 형태**로 제공 됩니다.
- 선택된 날짜의 일일 지표 추이가 **리스트 형태**로 제공 됩니다.

Cf) 분석을 원하는 날짜의 출력은 1,2번 화면에서 날짜 선택을 통해 확인 하실 수 있습니다.

실전 분석 사례 Case 1.
**CPU가 급격히 증가한 원인
을 알고 싶어요**

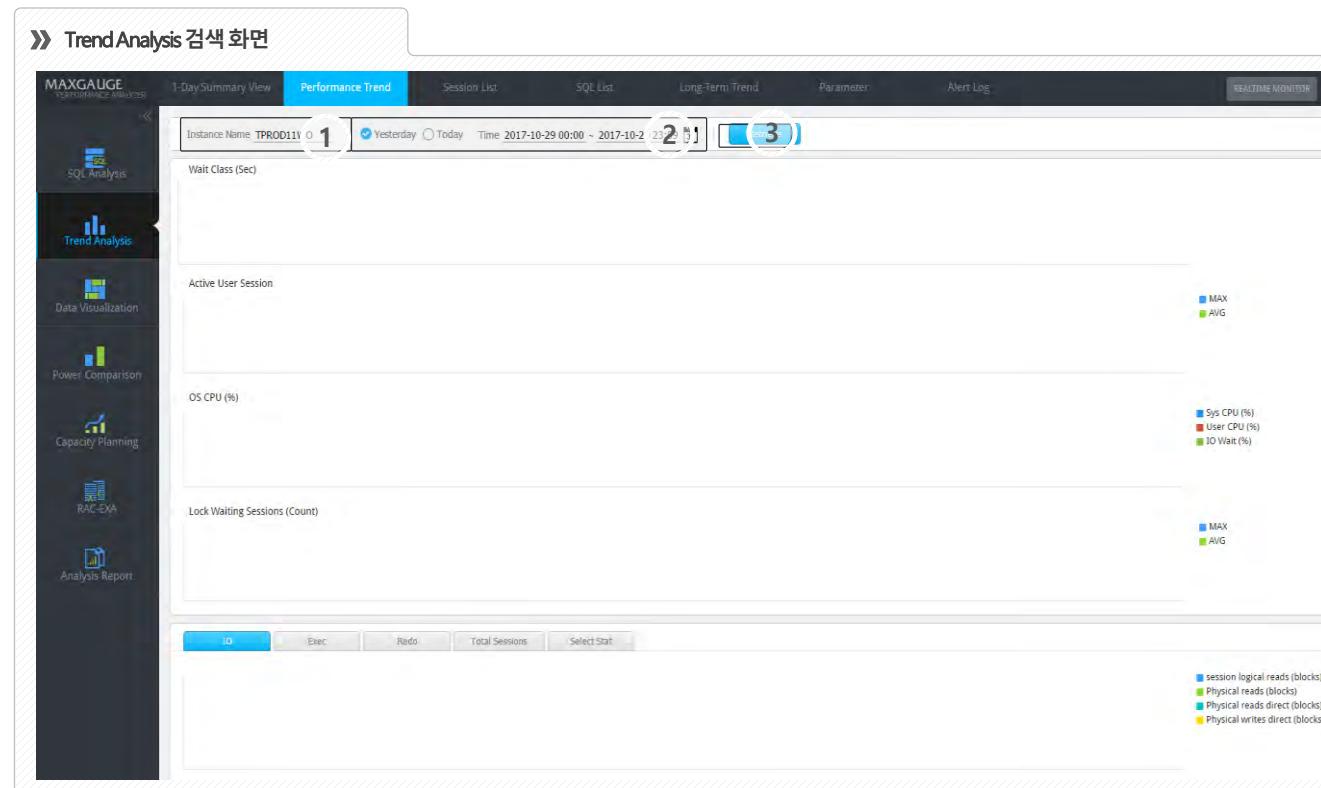
“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계



✓ 시나리오

INSTANCE : TPROD11WO
문제 발생일 : 2017년 10월 29일

✓ 목표

높은 CPU를 기록한 시점의 문제 상황 확인 및 원인을 파악을 합니다.

- 1 성능 분석을 진행할 인스턴스를 선택 합니다.
- 2 성능 분석을 진행할 날짜를(10월 29일) 선택합니다. (Performance Trend는 한 서버에 대해 최대 2일에 대한 성능 비교가 가능 합니다.)
- 3 위의 설정한 조건으로 Performance Trend를 실행합니다.

“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 성능 추이 확인

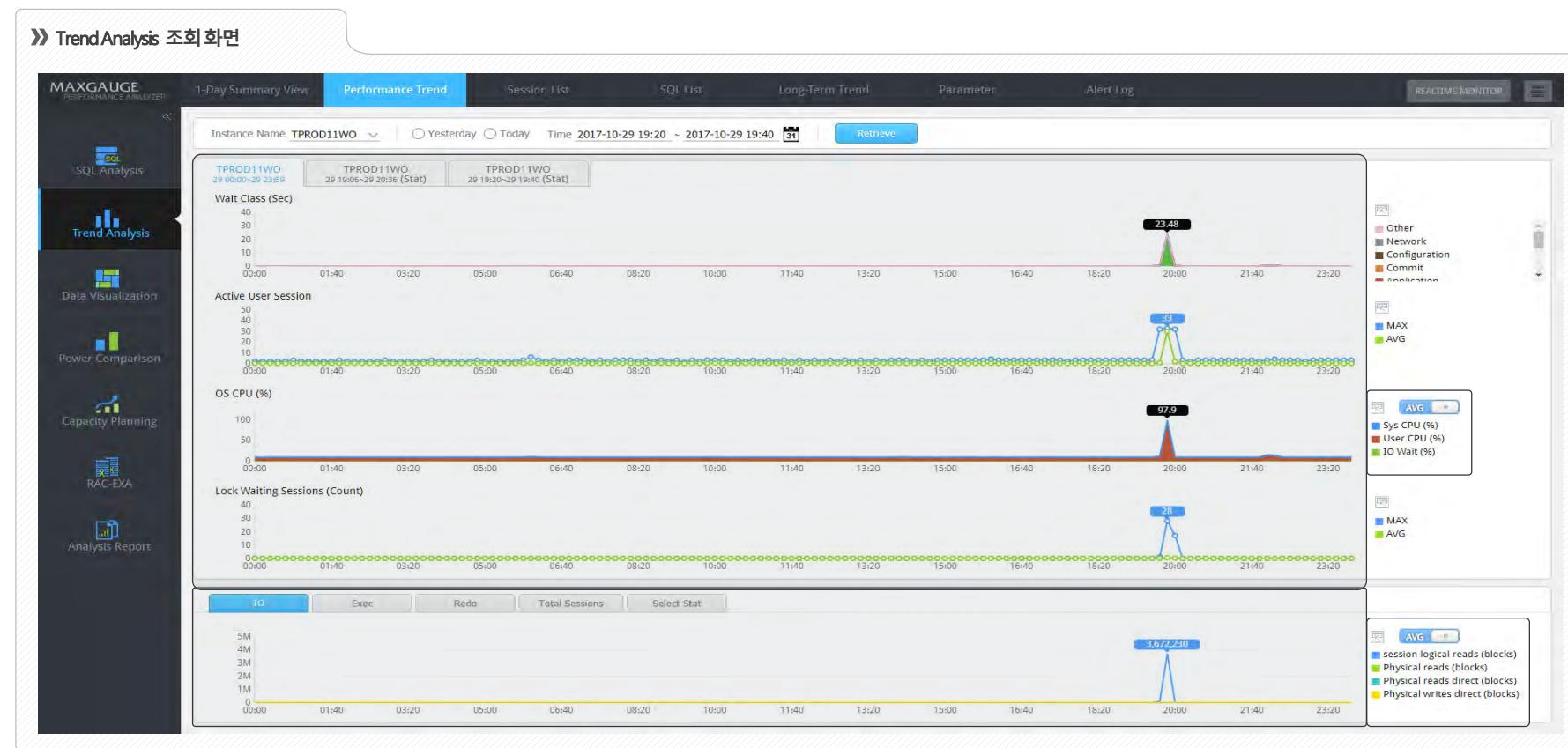
■ “DB 장애 시점” 확인

■ “상세분석”

└ Trend Analysis 연계

└ SQL Detail 연계

└ Stat Analysis 비교



- ✓ 하루의 성능 추이 중 OS 자원 사용률, Wait Class, Active User Session, IO, Lock Waiting Session등의 추이 그래프를 확인 합니다.
- ✓ CPU와 같은 몇몇 지표들은 Toggle 버튼을 클릭하여, Trend를 AVG 값으로 볼지 MAX 볼지를 선택할 수도 있습니다.
- ✓ 19시 40분경 OS CPU 지표와 DB 성능지표의 추이가 급격히 증가 된 것으로 보아, 해당 시점에 어떤 문제가 있다는 것을 **직관적으로** 예상할 수 있습니다.

“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 성능 추이 확인

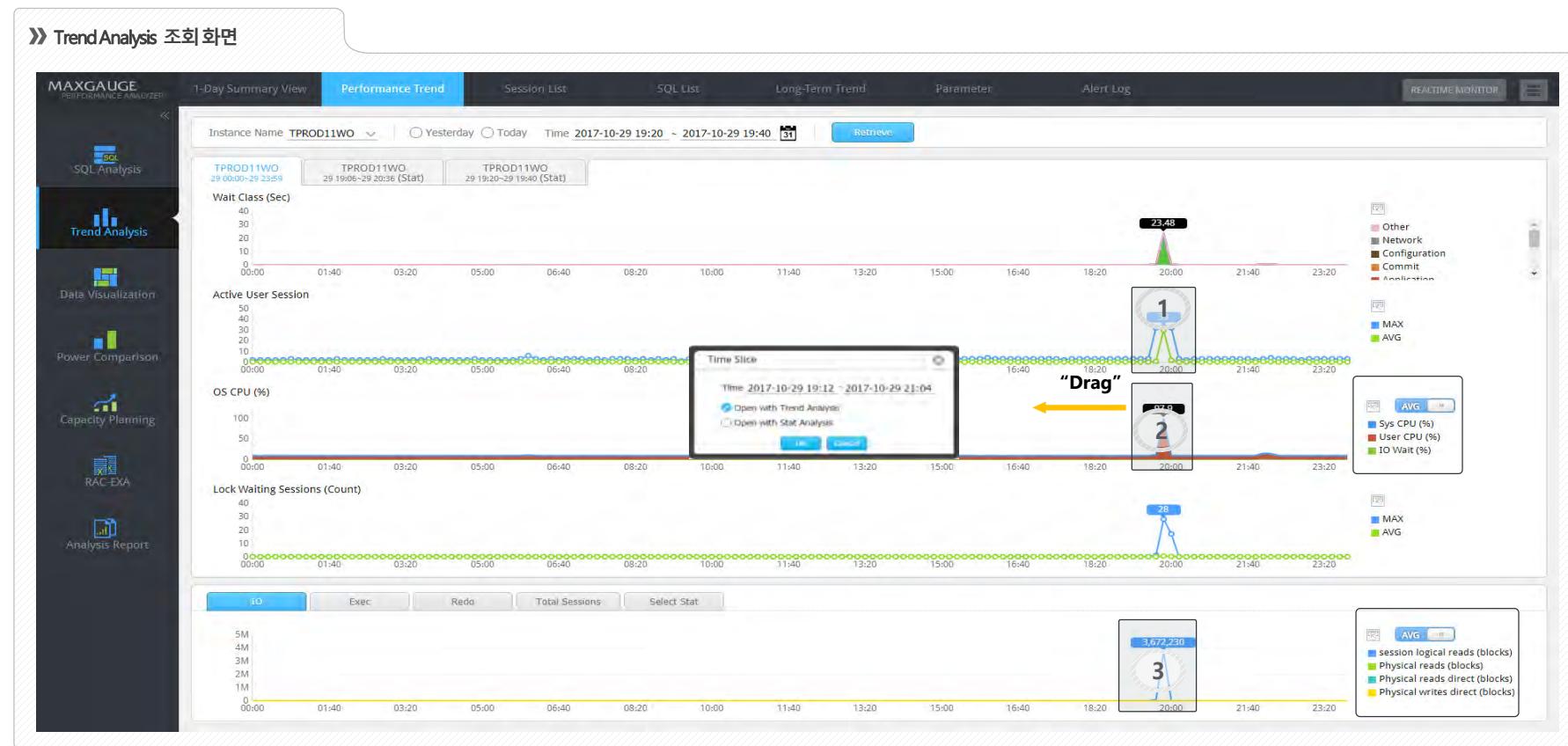
■ “DB 장애 시점” 확인

■ “상세분석”

└ Trend Analysis 연계

└ SQL Detail 연계

└ Stat Analysis 비교



✓ 10월19일 19시40분 부터, CPU, Active Session, Session Logical Reads 등 부하의 척도가 되는 주요 지표 3가지가 동시에 증가 된 것이 확인 됩니다.

✓ 이제 해당 시간 동안 무슨 일이 발생했는지 확인하기 위해, 문제의 구간을 “Drag” 하여, 당시의 상황을 상세히 분석해 보도록 하겠습니다.

(Drag시 Time Slice 기능이 활성 되며, 당시 Session들의 정보까지 확인이 가능합니다.)

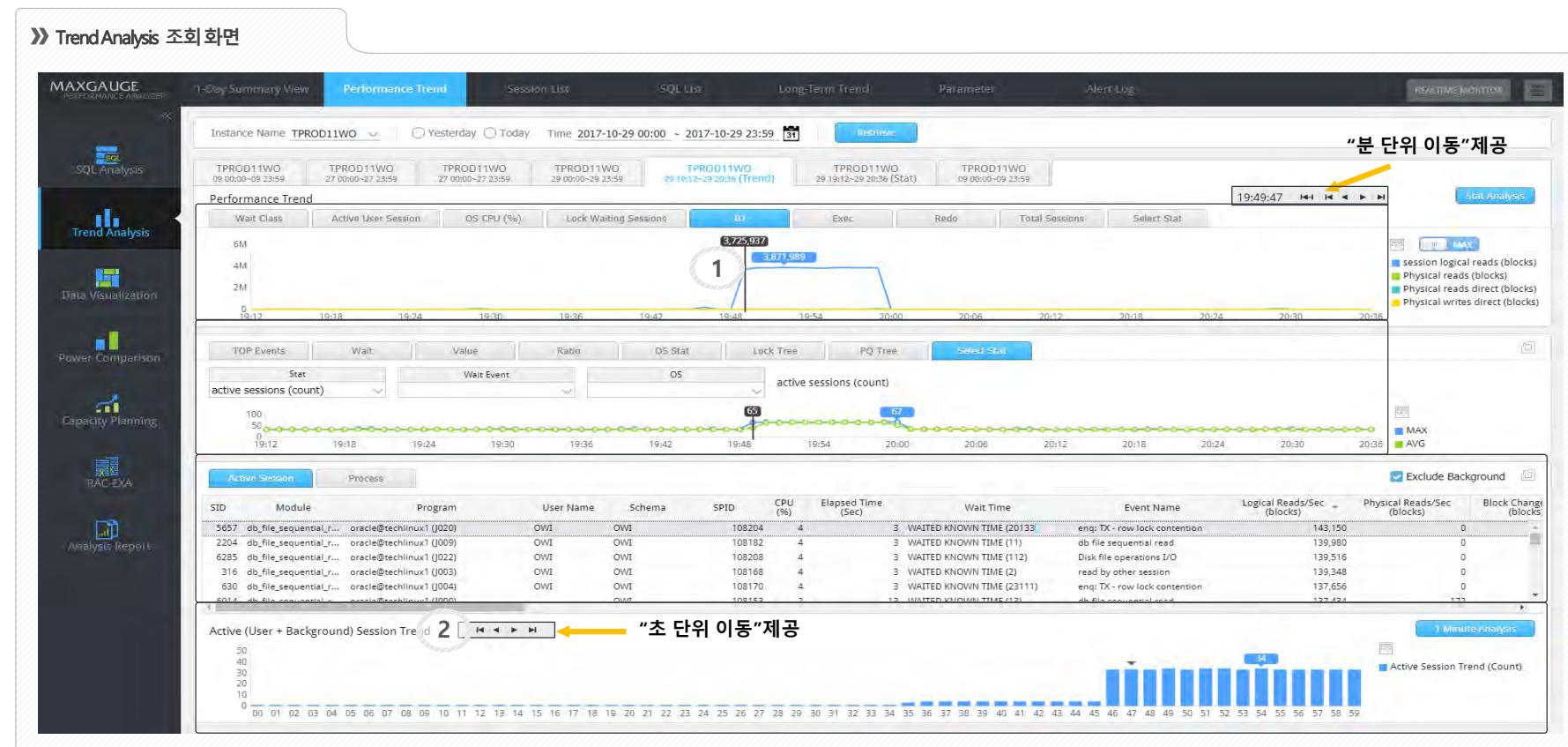
“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계
 - └ Stat Analysis 비교



1 Time Slice는 Drag 한 구간까지의 Trend만 보여집니다. 그리고 하단에는 당시의 수행된 Active Session 정보가 보여집니다. 우선 CPU가 급격히 증가한 시점 이동한 후

Active Session Trend에서 Active Session이 급격하게 증가한 초를 확인한 후, 막대 바를 클릭해 해당 시점으로 이동합니다.

2 성능 이슈 발생직전의 초(19시49분 45초)와 발생 후(19시 49분 46초)를 비교하여, 문제를 일으키는 Session을 파악합니다. 이 때 세션들이 겪는 이벤트와 I/O량 등은 문제를 분석

Key Point가 됩니다. (Session의 초 단위 정보 확인)

- ✓ 즉 성능 분석 시 Trend는 물론이고, 장애 발생 전후를 초 단위로 이동하여, Session이 급격히 증가하는지, 서서히 증가하는지, 어떤 Event와 어떤 SQL로 인해 지연이 되는지를 모두 파악해야 정확한 분석이 가능합니다.

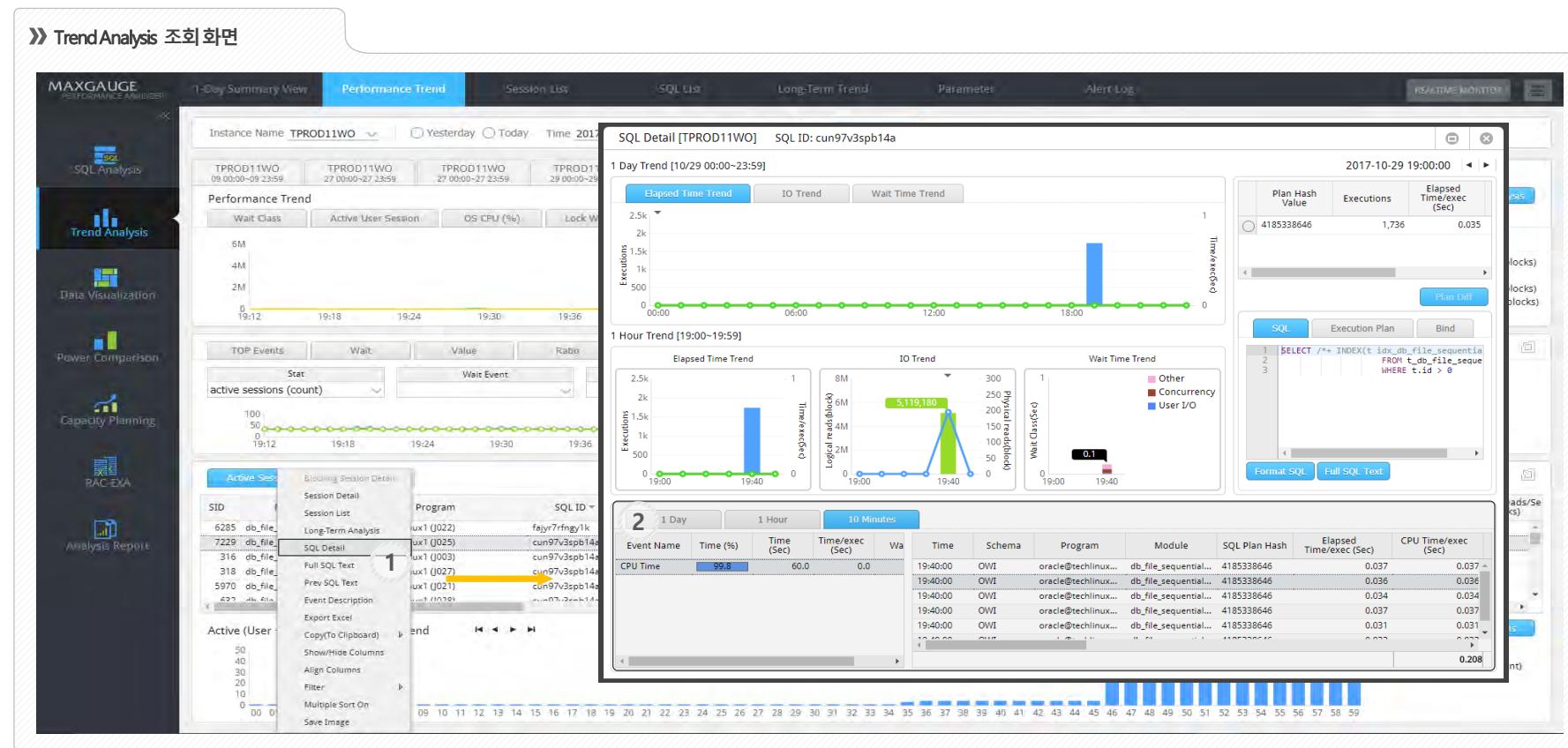
“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계
 - └ Stat Analysis 비교



- 1 I/O량이 높아, 부하의 원인으로 의심이 되는 Active Session의 경우, 우 클릭 후 SQL Detail로 연계하여, SQL에 대한 Detail한 분석을 진행합니다.
 - 2 SQL 단위의 성능 지표 확인 후, CPU 증가 시점 가장 많이 수행 된 SQL의 경우, 대기 이벤트 없이 CPU time에 대부분의 시간을 사용하였습니다. 이와 같은 경우에는 Wait Event는 없기 때문에, 왜 CPU 사용을 많이 했는가를 초점으로 분석하면 됩니다.
(악성 Plan으로 인해 I/O가 높나? Function이 존재하는가?, 비정상적으로 많이 수행되는 것은 아닌가? 정도를 관점으로 둡니다.)
- ✓ 문제 SQL은 1회 수행당 부하는 작았으나 Execution이 높았습니다. 즉 부하 시점에 해당 SQL이 많이 수행된 것이 CPU가 급격히 증가한 원인임을 알 수 있습니다

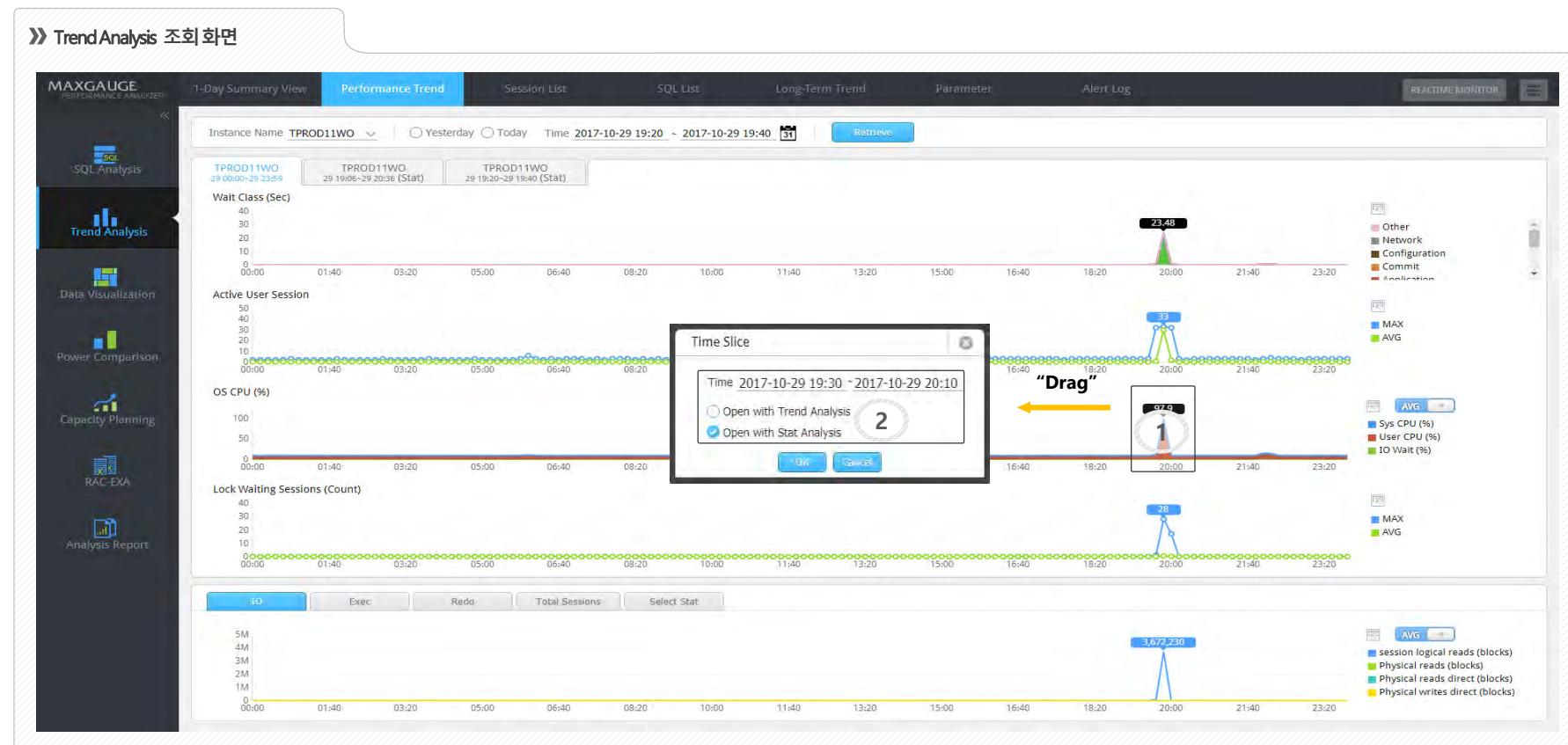
“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계
 - └ Stat Analysis 비교



- 1 분석 시점에 동일한 SQL만 수행하는 경우라면, 손쉽게 부하 SQL을 찾아 낼 수 있습니다. 하지만 SQL이 여러 개로 나뉘고, 이것이 동시 다발적으로 수행된다면 분석이 어렵습니다. 이와 같은 상황에서는 문제가 있는 구간과 없는 구간의 Top SQL 정보를 비교함으로써, 부하가 되는 SQL을 손쉽게 찾을 수 있습니다.
- 2 우선 성능 지표가 급격히 증가한 20시 부근을 “Drag”한 후 팝업 창에서는 Top SQL을 바로 볼 수 있게 Stat Analysis을 수행할 수 있도록 합니다.

“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계
 - └ Stat Analysis 비교

» Trend Analysis 조회 화면

SQL ID	Schema	Program	Module	SQL Plan Hash	Elapsed Time (%)	CPU Time (Sec)	CPU Time (%)	Logical Reads (%)	Logical Reads (blocks)	Physical Reads (%)	Executions	Elapsed Time (Sec)	Wait Time
Sudxh0ykgshkb	OWI	oracle@technix1 (Q...	Streams	0	68.1%	598	83.0%	0.0%	0	0.0%	0	600	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.1%	10	1.4%	10.0%	618,434	0.0%	265	10	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.1%	10	1.4%	15.9%	981,586	0.0%	320	10	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.1%	10	1.4%	12.7%	782,467	0.0%	298	10	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.1%	10	1.4%	15.0%	926,943	0.0%	277	10	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.1%	10	1.4%	11.5%	711,288	0.0%	269	10	
cun97v3spb14a	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	1.2%	10	1.5%	17.8%	1,098,462	0.3%	307	10	
8guhtcnlk660	MAXGAUGE	JDBC Thin Client		2905781256	0.9%	8	1.1%	0.0%	0	0.0%	13	8	
InqhyTr9xmxfy	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	0.5%	5	0.5%	2.3%	143,322	0.0%	254	5	
cpm8tdmwf8ha	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	0.5%	5	0.7%	3.1%	189,208	0.0%	298	5	
gmzuawvn364dp	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	0.5%	5	0.7%	3.5%	216,854	0.0%	321	5	
g373c5ddsc9m	OWI	oracle@technix1 (J...	db_file_sequential_read	4185338646	0.4%	4	0.5%	2.0%	124,070	0.0%	279	4	

SQL ID : Sudxh0ykgshkb

Stat Name	Avg	Sum	Wait Time	Sum
Executions	1	1		
Elapsed Time	600.050	600	library cache: mutex X	2
CPU Time	598.070	598		2
Logical Reads	0	0		
Physical Reads	0	0		
Redo Size	0	0		
Sort Disk	0	0		
Sort Rows	0	0		
Table Scan Blocks Gotten	0	0		
Table Scan Rows Gotten	0	0		
Table Fetch By Rowid	0	0		
Table Fetch Continued By Rowid	0	0		

SQL: select location_name, max(r.reg_id) from reg\$ r left outer join gv\$subscriber\$ s on r.reg_id = s.subscriber_id where location_name like '%서울%' and s.subscriber_id is null

Execution Plan: ...

Bind: ...

1 CPU가 높았기 때문에 우선 SQL 정보에서 CPU Time 기준으로 내림 차순 정렬을 합니다.

✓ CPU Time과 Logical Reads 지표를 동시에 비교를 해 보면, CPU Time이 높은 SQL이 Logical Reads도 높은 것을 확인 할 수 있습니다.

(Session Logical Reads가 높았다는 것은 실제 DB에서 Memory I/O가 높은 것을 의미하는데 이는 작업량이 높은 것을 의미합니다.)

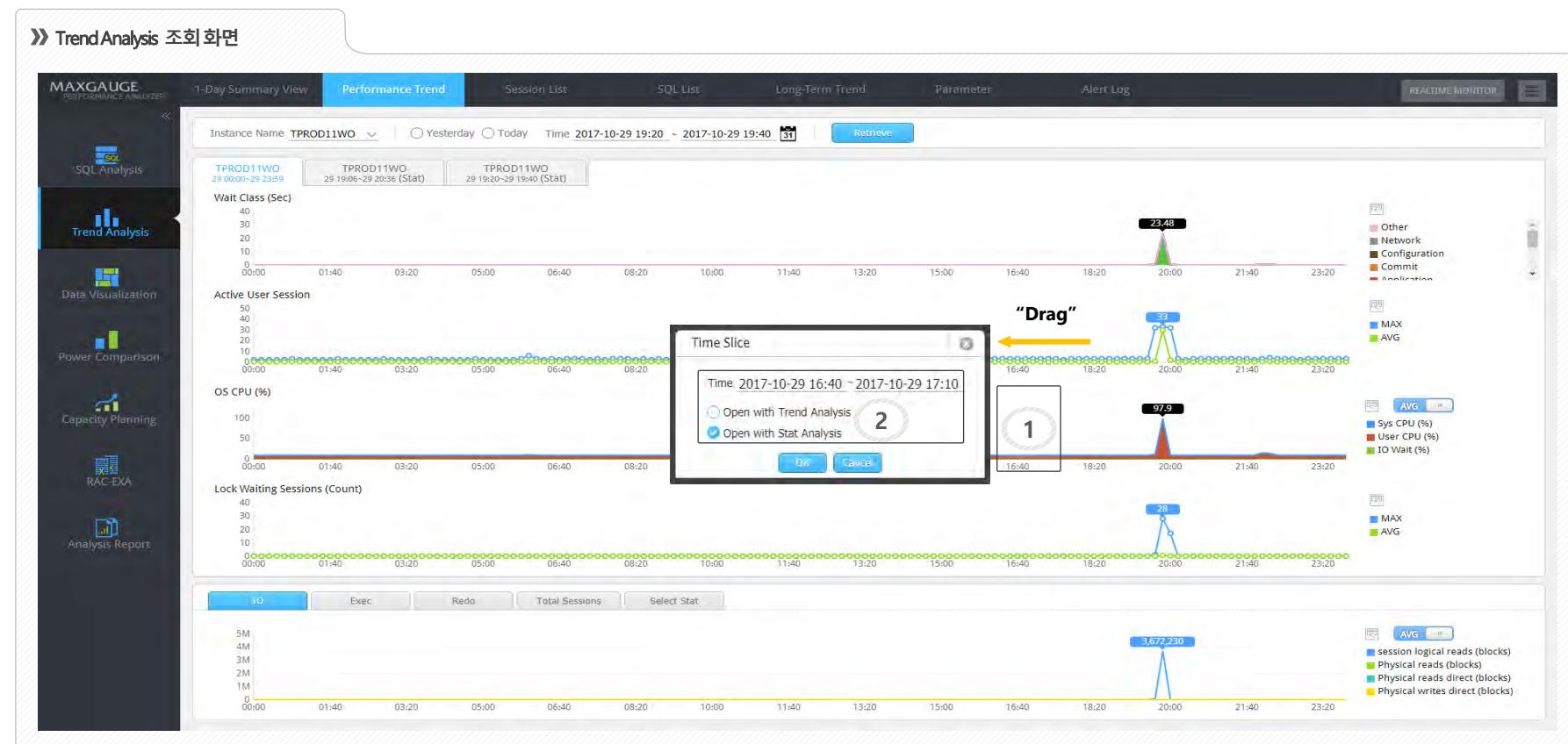
“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인
- “상세분석”
 - └ Trend Analysis 연계
 - └ SQL Detail 연계
 - └ Stat Analysis 비교



- 1 부하가 된 시점을 확인했다면, 이제는 **부하가 없던 시점을 확인**하여, 평상시에는 수행되지 않았던 SQL들이 존재하는지 파악하면 됩니다.
- 2 그럼 “Drag”를 통해 정상적으로 DB가 운영된 시점 시점의 Stat Analysis를 확인해 보도록 하겠습니다.

“CPU가 급격히 증가한 원인을 알고 싶어요”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- “DB 장애 시점” 확인

▪ “상세분석”

- └ Trend Analysis 연계
- └ SQL Detail 연계
- └ Stat Analysis 비교

» Trend Analysis 조회 화면

SQL	Schema	Program	Module	Machine	OS User			
Elapsed Time	CPU Time	Logical Reads	Physical Reads	Executions				
5udxh0ykgshkb	oracle@techlinux1 (Q... Streams	0	53.2%	598	53.6%	0.0%	0	
cb21bacyh3c7d	oracle@techlinux1 (Q... Streams	0	45.1%	504	45.2%	0.0%	0	
8guhtcnkk650	MAXGAUGE JDBC Thin Client	JDBC Thin Client	2905781256	0.5%	6	0.5%	0.0%	0
bglyynbb8g9v2	DBSNMP JDBC Thin Client	emagent_SQL_oracle_...	85920909	0.1%	2	0.1%	0.0%	0
0uuuzurv6jnj	DBSNMP JDBC Thin Client	emagent_SQL_oracle_...	2474890404	0.1%	1	0.1%	0.0%	0
196mqnmmpxpv1	DBSNMP JDBC Thin Client	emagent_SQL_oracle_...	2501378660	0.1%	1	0.1%	0.0%	0
5wwmny5t3676	DBSNMP JDBC Thin Client	emagent_SQL_oracle_...	4274421523	0.1%	1	0.1%	0.0%	0
a1dkahs57u6k	DBSNMP perl@techlinux1 (TNS...	Oracle Enterprise Ma...	1743960565	0.0%	0	0.0%	95.1%	64,259
bunssq950snhf	DBSNMP oracle@techlinux1 (M...	MMON_SLAVE	2694099131	0.0%	0	0.0%	0.0%	0
4dy1xm4nxclgf	DBSNMP oracle@techlinux1 (M...	MMON_SLAVE	3821145811	0.0%	0	0.0%	0.0%	0
0ogss90yyrn57	DBSNMP oracle@techlinux1 (M...	MMON_SLAVE	82777415	0.0%	0	0.0%	0.0%	0
84k66rt2s7y1c	DBSNMP oracle@techlinux1 (M...	MMON_SLAVE	3821145811	0.0%	0	0.0%	0.0%	0

10월29일 DB 정상 시점 SQL(16시40~17시10분)

SQL	Schema	Program	Module	Machine	OS User			
Elapsed Time	CPU Time	Logical Reads	Physical Reads	Executions				
Sudxh0ykgshkb	oracle@techlinux1 (Q... Streams	0	69.1%	598	83.0%	0.0%	0	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.1%	10	1.4%	10.0%	616,434	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.1%	10	1.4%	15.9%	981,586	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.1%	10	1.4%	12.7%	782,467	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.1%	10	1.4%	15.0%	926,943	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.1%	10	1.4%	11.5%	711,288	
cun97v3spb14a	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	1.2%	10	1.5%	17.8%	1,098,462	
8guhtcnkk660	MAXGAUGE JDBC Thin Client	JDBC Thin Client	2905781256	0.9%	8	1.1%	0.0%	0
1nqhy1h9xmfxy	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	0.5%	5	0.6%	2.3%	143,922	
cpm8fdnw8ha	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	0.5%	5	0.7%	3.1%	189,208	
gmzuawtn504ep	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	0.5%	5	0.7%	3.5%	216,854	
g373c9sdscsp9m	OWI oracle@techlinux1 (Q... db_file_sequential_read	4185338646	0.4%	4	0.5%	2.0%	124,070	

10월29일 CPU 부하 시점SQL(19시40~20시10분)

- ✓ 정상적 운영 시점과 CPU 부하 시점에 대해 CPU Time으로 정렬을 하고, 두 시점에 대해 SQL문을 비교할 수 있습니다.
- ✓ 부하시점과 정상시점의 Stat 비교하면, 문제가 되는 이벤트, 부하를 일으킨 SQL문을 쉽고 정확하게 파악할 수 있습니다.
- ✓ 확인 결과 정상적인 구간에서는 보이지 않던 OWI Schema에서 수행된 SQL이 CPU 부하 시점 전체 CPU Time의 83%를 점유하고 있는 것으로 보아

해당 SQL이 많이 수행된 것이 CPU증가의 원인임을 확인 할 수 있었습니다.

실전 분석 사례 Case 2.
**RAC 간의 Load Balance 및 성
능을 확인하고 싶어요.**

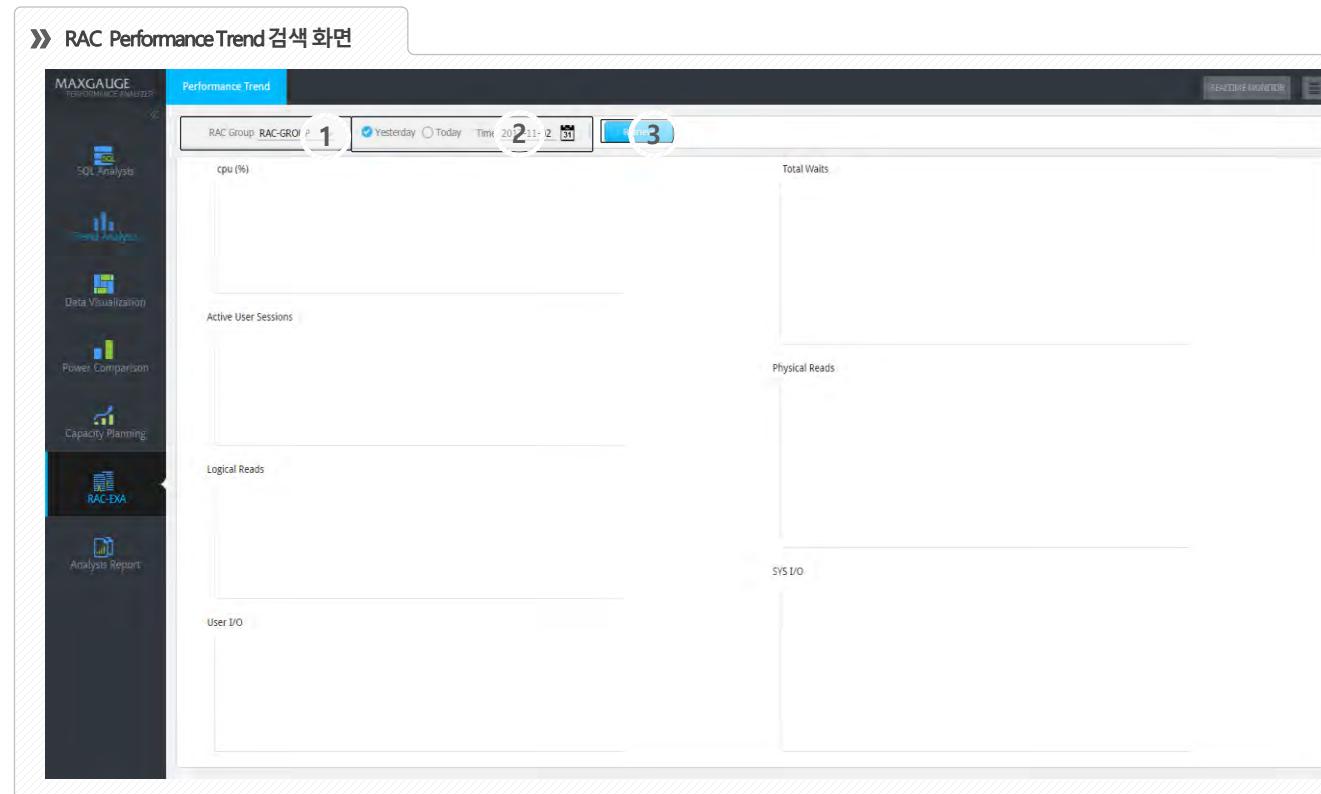
“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
- Trend Analysis



- 1 성능 분석을 원하는 RAC 그룹을 설정 합니다. (설치 시 RAC node간의 그룹 설정 가능)
- 2 분석을 원하는 날짜를 선택 합니다. (RAC 그룹에 속한 서버의 하루간의 성능 비교가 가능)
- 3 위의 설정한 조건으로 Performance Trend를 실행합니다.

✓ 시나리오

INSTANCE : TRODRAC1WO, TRODRAC2WO

문제 발생일 : 2017년 11월 10일

✓ 목표

RAC 구성 후 서비스 오픈 테스트 후 node 별 작업량을 비교 할 수 있다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

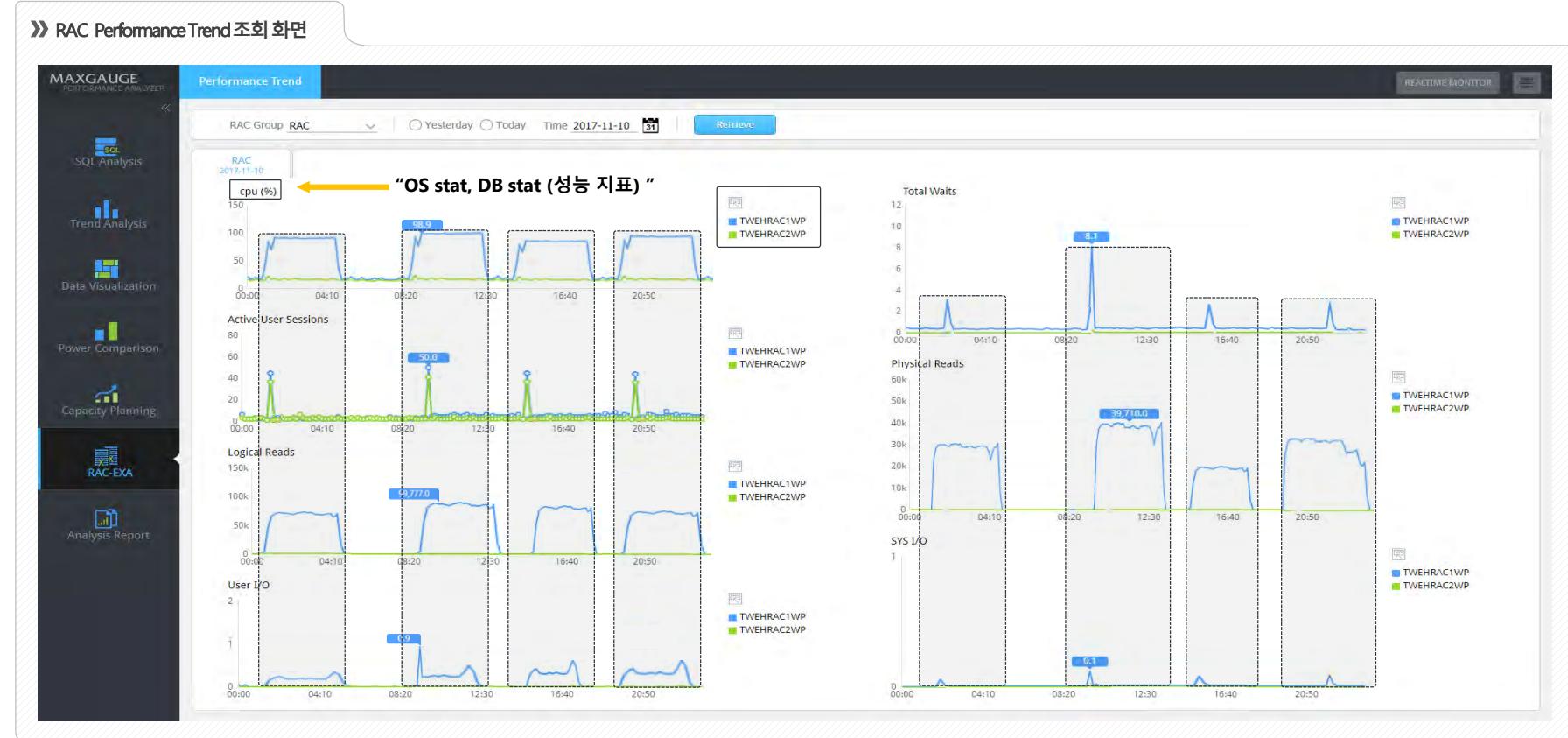
1. 검색 조건 설정

2. 데이터 분석

▪ 성능 추이 확인

▪ Trend Analysis

- └ Trend Analysis 연계
- └ 1 minute Analysis
- └ Stat Analysis 비교



- ✓ 위 화면은 RAC-Group에 속한 서버들의 OS stat, DB stat 의 성능 추이를 보여주고 있습니다.
- ✓ 두 Node를 동시에 각기 다른 색으로 표기함으로써, Node 별 성능 추이를 빠르고 손쉽게 확인할 수 있습니다.
- ✓ 11/10일 로그를 확인해 본 결과 Node1(TWEHRAC1WP)에서만 주기적으로 CPU 와 DB Stat 값이 크게 증가 했다. 감소하기를 반복하는 것을 확인 할 수 있습니다.

(node1에 DB부하가 있는 걸로 예상 됨)

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

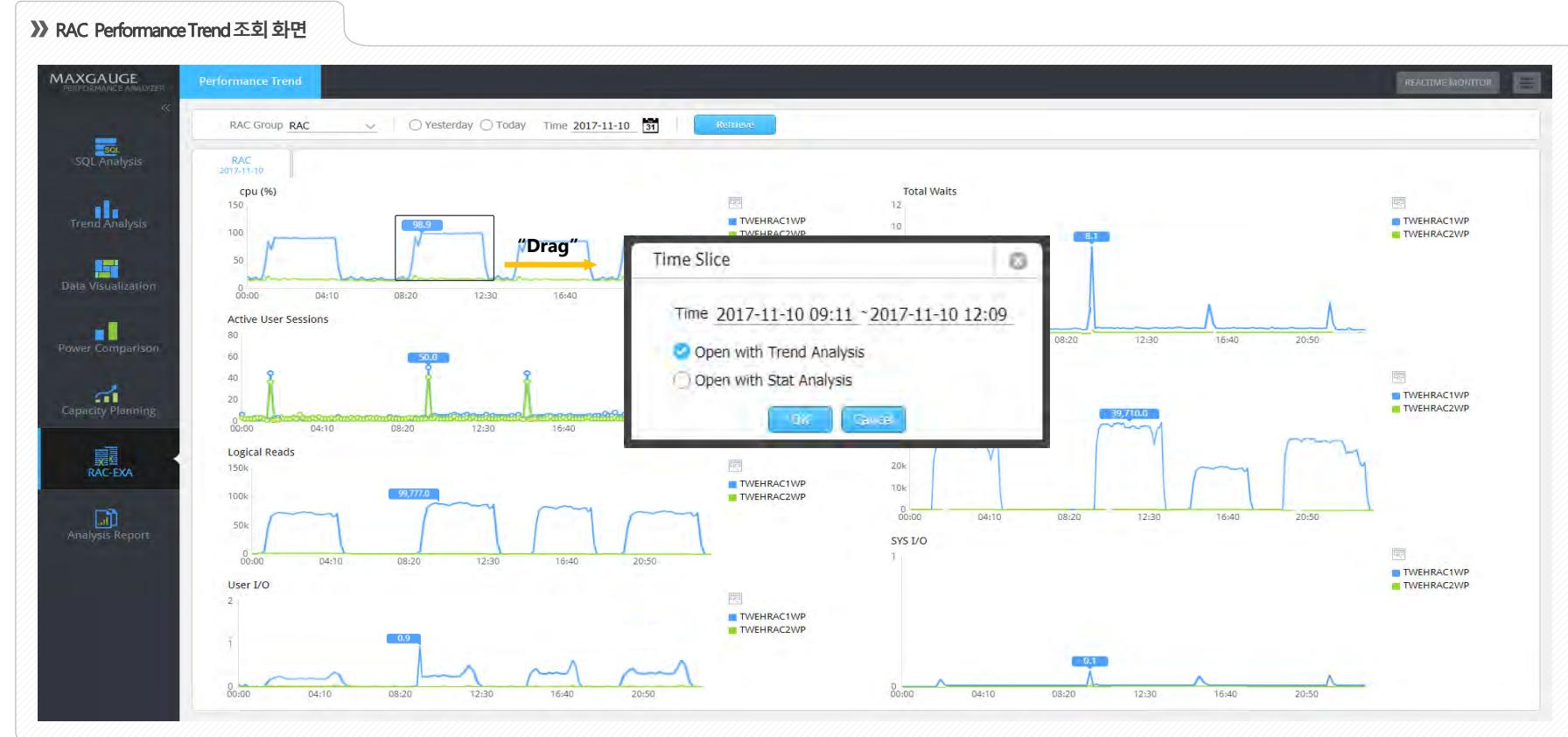
- 성능 추이 확인

- Trend Analysis

 - └ Trend Analysis 연계

 - └ 1 minute Analysis

 - └ Stat Analysis 비교



✓ 부하가 되는 시점의 무슨 일이 벌어졌는지 좀 더 자세히 확인하기 위해서 “Drag”를 통해 Trend Analysis 연계 합니다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

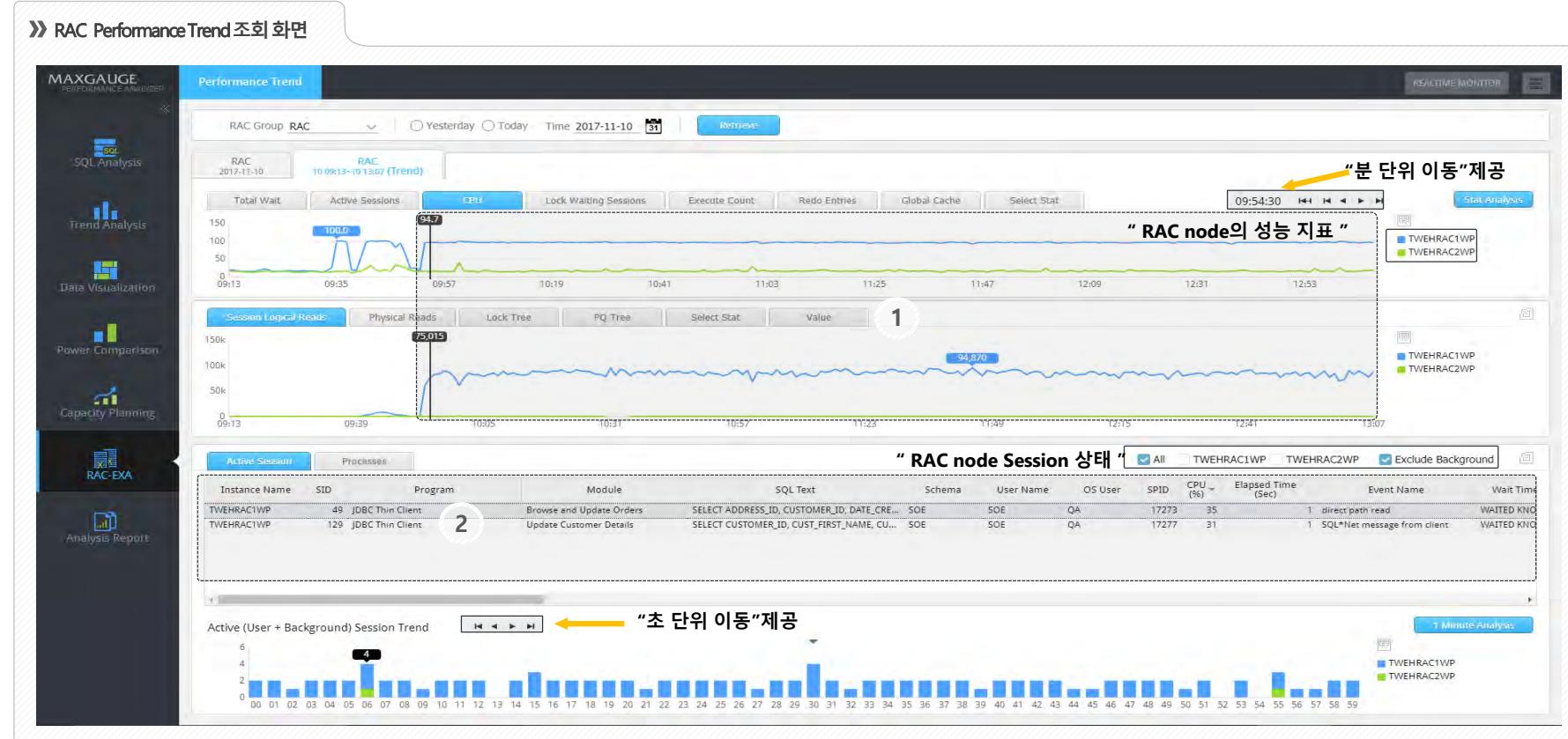
■ 성능 추이 확인

■ Trend Analysis

└ Trend Analysis 연계

└ 1 minute Analysis

└ Stat Analysis 비교



1 로그를 확인한 결과 항상 1번 node만 CPU와 Logical Reads가 급증 한 것을 확인 할 수 있습니다.

2 1번에서만 CPU와 Logical Reads가 증가한 원인을 파악하기 위해 Active Session 탭을 클릭하여, 당시에 수행된 Session과 SQL문을 확인 할 수 있습니다.

(동 시간 때 2번 node에 Session 접속이 거의 없음)

" RAC 간의 Load Balance 및 성능을 확인하고 싶어요 "

STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

- #### ■ 성능 추이 확인

▪ Trend Analysis

- ## ↳ 1 minute Analysis



1 Active Session에서 초 단위로 이동하며 로그를 분석할 수 있으나, 다양한 Event가 발생할 경우 이를 분석하기 위해 많은 시간이 소요됩니다.

이 때 화면 하단의 **1minute Analysis** 버튼을 클릭하면 1분간 Stat / Wait에 대한 통계 정보를 확인함으로써 빠르고 정확하게 분석할 수 있습니다.

2 Wait Class와 Wait Event를 확인해 보면, 당시 CPU Time과 I/O 관련 대기 이벤트가 높았던 것을 알 수 있습니다. (Direct Path I/O, Db file sequential reads)

3. 또한 Active Session 정보를 **Sum** 기능을 활용해 CPU를 많이 사용한 Session은 물론이고 SQL까지 분석 가능합니다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

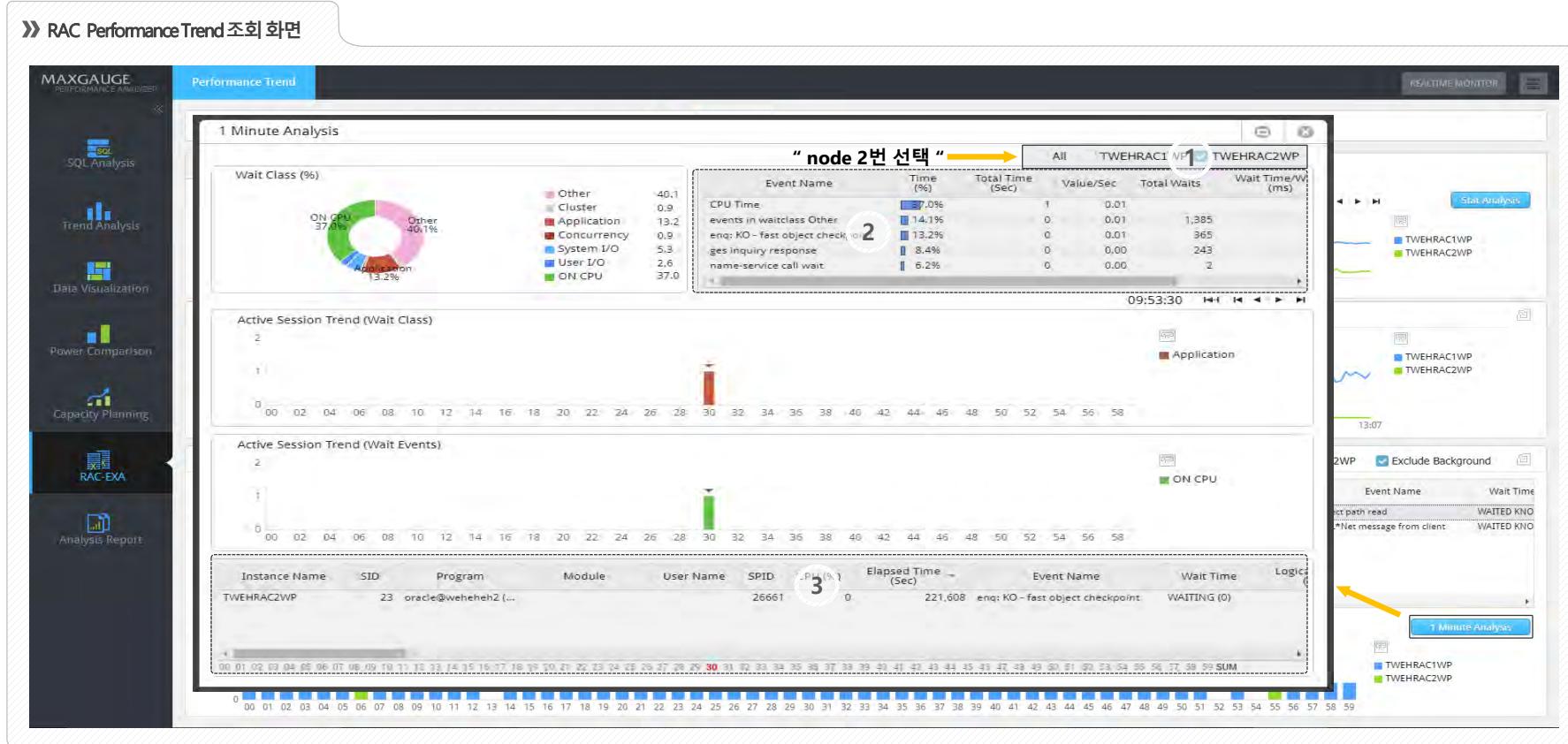
■ 성능 추이 확인

■ Trend Analysis

└ Trend Analysis 연계

└ 1 minute Analysis

└ Stat Analysis 비교



1 RAC 1min 분석의 2번 node Session 정보를 확인 합니다.

2 Event 확인 결과 동시간 때 1번 node에 비해 Total Time이 수치가 낮은 것을 확인 할 수 있습니다. (서버에 작업이 없음을 의미)

3 하단의 Session 확인을 통해 접속한 Session이 거의 없음을 확인 할 수 있습니다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

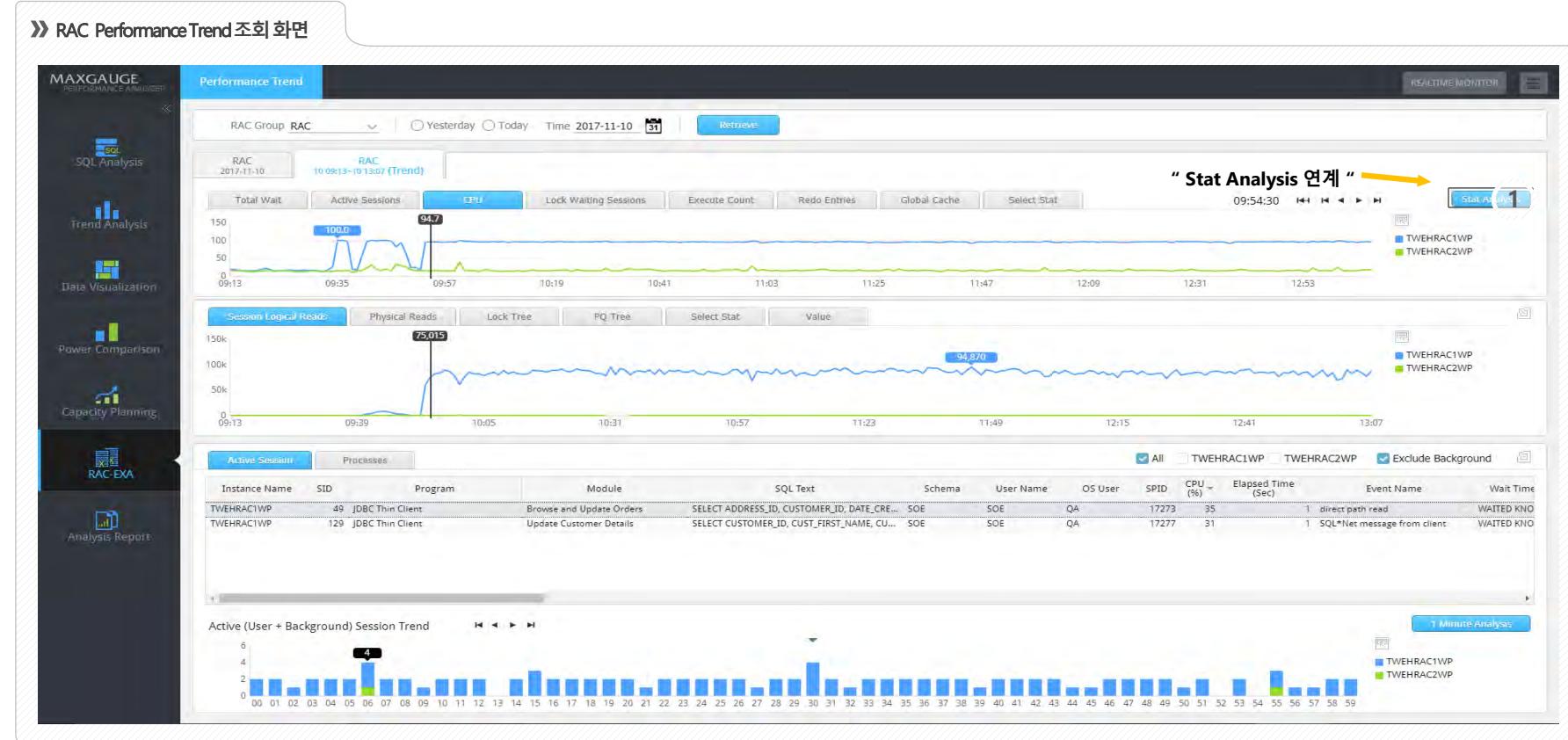
■ 성능 추이 확인

■ Trend Analysis

└ Trend Analysis 연계

└ 1 minute Analysis

└ Stat Analysis 비교



1 하지만 1분 구간만 확인할 경우, 전체에 걸쳐 발생한 것 마냥 오판하는 일반화 오류가 발생할 수도 있습니다.

이러한 우를 범하지 않기 위해서는 Drag한 구간 전체에 대해 분석하는 것입니다. 이는 화면 상단의 Stat Analysis 버튼을 클릭하여 분석이 가능합니다.

✓ Stat Analysis는 24시간 중 Drag 하여 잘라낸 Time Slice 구간 전체에 대해 Top N / Top Event를 분석하는 용도로 사용됩니다. 이를 활용하면 당시 구간의

부하를 유발한 SQL은 물론이고 당시 경합을 겪게 된 주된 원인의 Event를 분석할 수 있습니다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 성능 추이 확인

■ Trend Analysis

└ Trend Analysis 연계

└ 1 minute Analysis

└ Stat Analysis 비교

» RAC Performance Trend 조회 화면

Instance Name	SQL ID	Schema	Program	Module	SQL Plan Hash	Logical Reads/exec (blocks)	Logical Read (%)	Elapsed Time (%)	CPU Time (%)	Physical Reads (%)	Executions	Elapsed Time (Sec)
TWEHRAC1WP	g81cbrq5yamf5	SOE	JDBC Thin Client	New Order	1286489376	12,066	22.4%	29.2%	0.9%	26,579		
TWEHRAC1WP	7ws837ygnplzv	SOE	JDBC Thin Client	New Order	2597291669	6,902	17.5%	17.1%	15.5%	53.0%	26,543	
TWEHRAC1WP	8zz6y2ydzqj0p0	SOE	JDBC Thin Client	Update Customer Details	1005345217	16,110	14.2%	24.8%	31.0%	22.8%	6,424	
TWEHRAC1WP	70959moy5g	SOE	JDBC Thin Client	Browse and Update Orders	33541244	16,375	7.5%	3.8%	3.3%	15.9%	3,348	
TWEHRAC1WP	g81cbrq5yamf5	SOE	JDBC Thin Client	Browse and Update Orders	1286489376	12,000	5.5%	2.8%	3.7%	0.1%	3,354	
TWEHRAC1WP	29qp10usqkjh0	SOE	JDBC Thin Client	Sales Rep Query	3619984409	17,419	3.2%	2.0%	1.8%	6.5%	1,355	
TWEHRAC1WP	d1w3yvqzq3qn	SOE	JDBC Thin Client	Datagenerator Worker Thread	0	23	0.0%	5.2%	3.4%	0.0%	3,462	
TWEHRAC1WP	99p0z8u009916	SOE	JDBC Thin Client	Datagenerator Worker Thread	0	5	0.0%	7.3%	3.4%	0.0%	13,720	
TWEHRAC1WP	c13sm46kr27c	SOE	JDBC Thin Client	New Order	1197921474	15	0.0%	0.9%	0.7%	0.0%	4,234	
TWEHRAC1WP	cg9mh3ns0na0	SOE	JDBC Thin Client	Datagenerator Management Thread	0	49,500	0.0%	0.1%	0.0%	0.0%	1	
TWEHRAC1WP	1qw8j8ydtas5	SOE	JDBC Thin Client	Datagenerator Worker Thread	0	51	0.0%	0.7%	0.5%	0.0%	919	

Filter : (Module Not Like (%MMON_SLAVE))

SQL ID : 5xy32f0pphbpp

Stat Name	Avg	Sum	Wait Time	Sum
Executions	1	1		7
Elapsed Time	9.950	10		
CPU Time	3.450	3		
Logical Reads	24,386	24,386		
Physical Reads	16,760	16,760		
Redo Size	412,884	412,884		
Sort Disk	0	0		
Sort Rows	1,000,000	1,000,000		
Table Scan Blocks Gotten	16,867	16,867		
Table Scan Rows Gotten	7	0		
Table Fetch By Rowid	7	7		
Table Fetch Continued By Rowid	0	0		

SQL

```

1 ALTER TABLE customers
2 ADD ( CONSTRAINT customers_pk PRIMARY KEY (customer_id) NOVALIDATE)
3

```

Execution Plan

Bind

Format SQL Full SQL Text

1 Stat 분석의 1번 node를 선택 합니다.

2 1번 node Session의 부하를 발생시킨 SQL이 어떤 것인지 알기 위해 Session Logical Reads 기준으로 정렬 합니다.

만약 Top 1, 2에 해당하는 module명이 이번에 추가된 업무 였다고 한다면, 예전과는 달리 성능 지표가 급격히 증가했다가 다시 내려가는 현상은

해당 업무가 추가되어 발생한 것이라 판단 할 수 있습니다.

✓ 좀 더 확실하게 확인하기 위해 2번 node도 살펴 보도록 하겠습니다.

“ RAC 간의 Load Balance 및 성능을 확인하고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

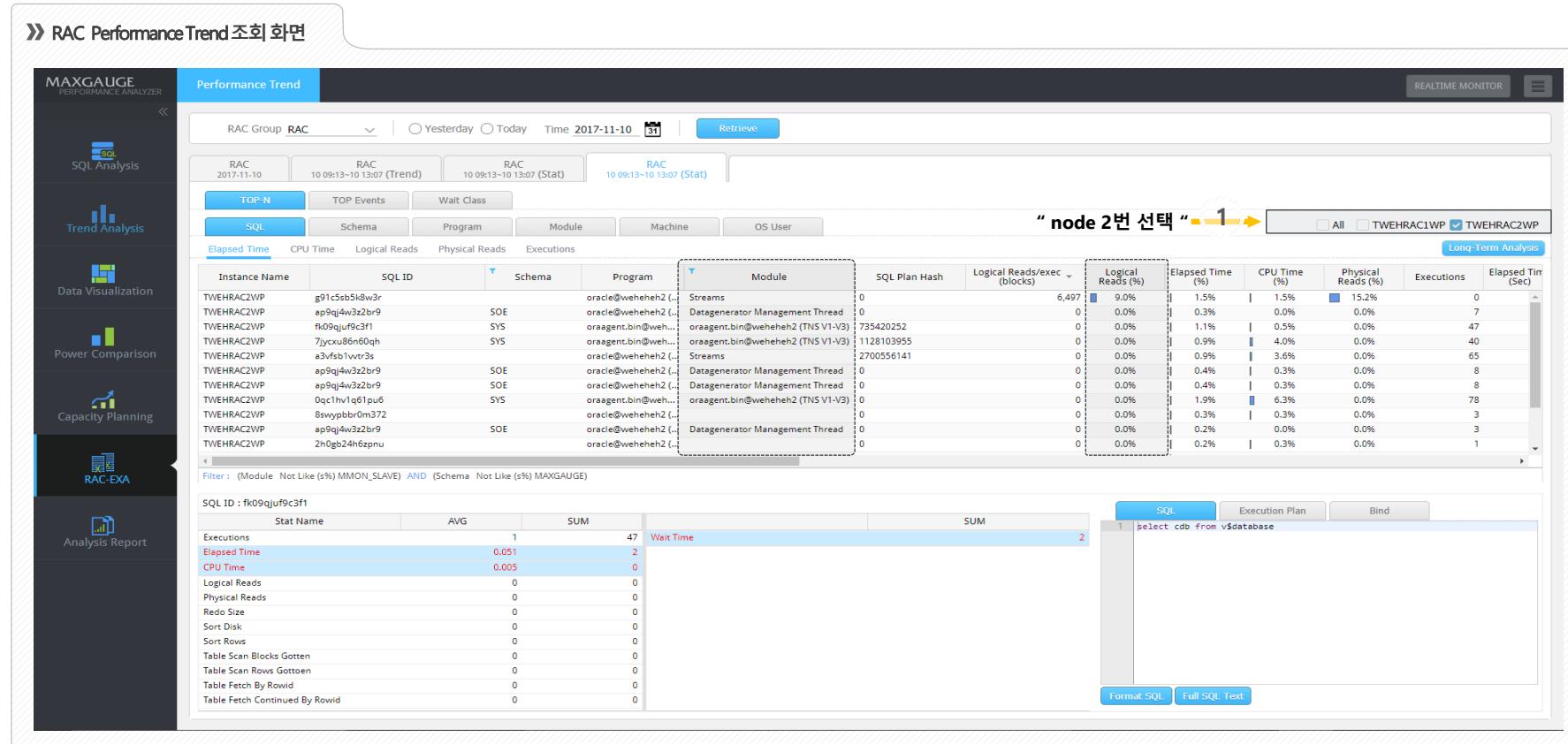
■ 성능 추이 확인

■ Trend Analysis

└ Trend Analysis 연계

└ 1 minute Analysis

└ Stat Analysis 비교



1 2번 node의 정보를 확인해 본 결과 Node 1번에서 수행되었던 신규 업무에 관련된 SQL의 수행 정보가 없습니다.

✓ 만약 신규 업무가 Node 1, Node 2에서 나누어서 실행되기를 원했던 경우라면, RAC 구성임에도 불구하고 1번 node로만 Session 접속이 이루어져

1번 Node에서만 부하가 발생 한 것입니다. 즉 RAC의 “ TNS 설정 정보 확인, 업무 구분을 통한 분산 작업 ”에 대한 정밀한 검토가 필요할 것입니다.

실전 분석 사례 Case 3.
**한 달간 가장 큰 부하의 원
인을 알고 싶어요.**

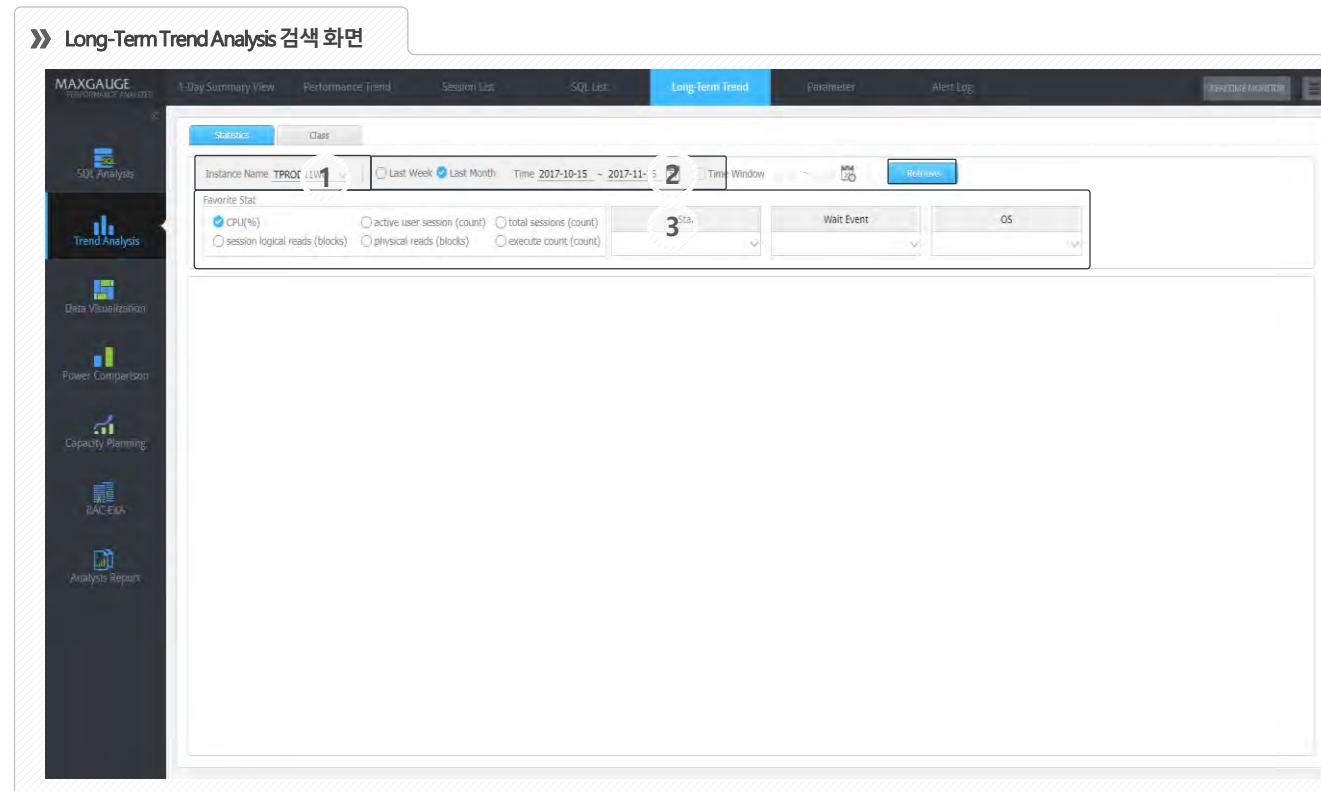
“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 성능 추이 확인
 - └ 1달 성능 Trend 확인
 - └ 부하 시점 Trend 확인
- Trend Analysis
 - └ Stat Analysis 비교



- 1 성능 비교를 원하는 인스턴스를 선택 합니다.
- 2 분석을 원하는 날짜를 선택 합니다. (최소 당일부터 로그 보관 주기 까지 성능 추이를 확인 할 수 있습니다.)
- 3 성능 추이를 확인하고자 하는 지표를 선택합니다. (성능 분석 시 자주 사용하는 지표에서부터, 사용자가 원하는 지표를 직접 선택 할 수 있습니다.)
- 4 위의 설정한 조건을 기준으로 Long-Term Trend를 실행 합니다.

✓ 시나리오

INSTANCE : TROD11WO
분석 기간 : 2017.10.15~2017.11.15

✓ 목표

한 달간 로그 분석을 통해 대상 서버의 성능 추이파악과 부하 시점을 확인

“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

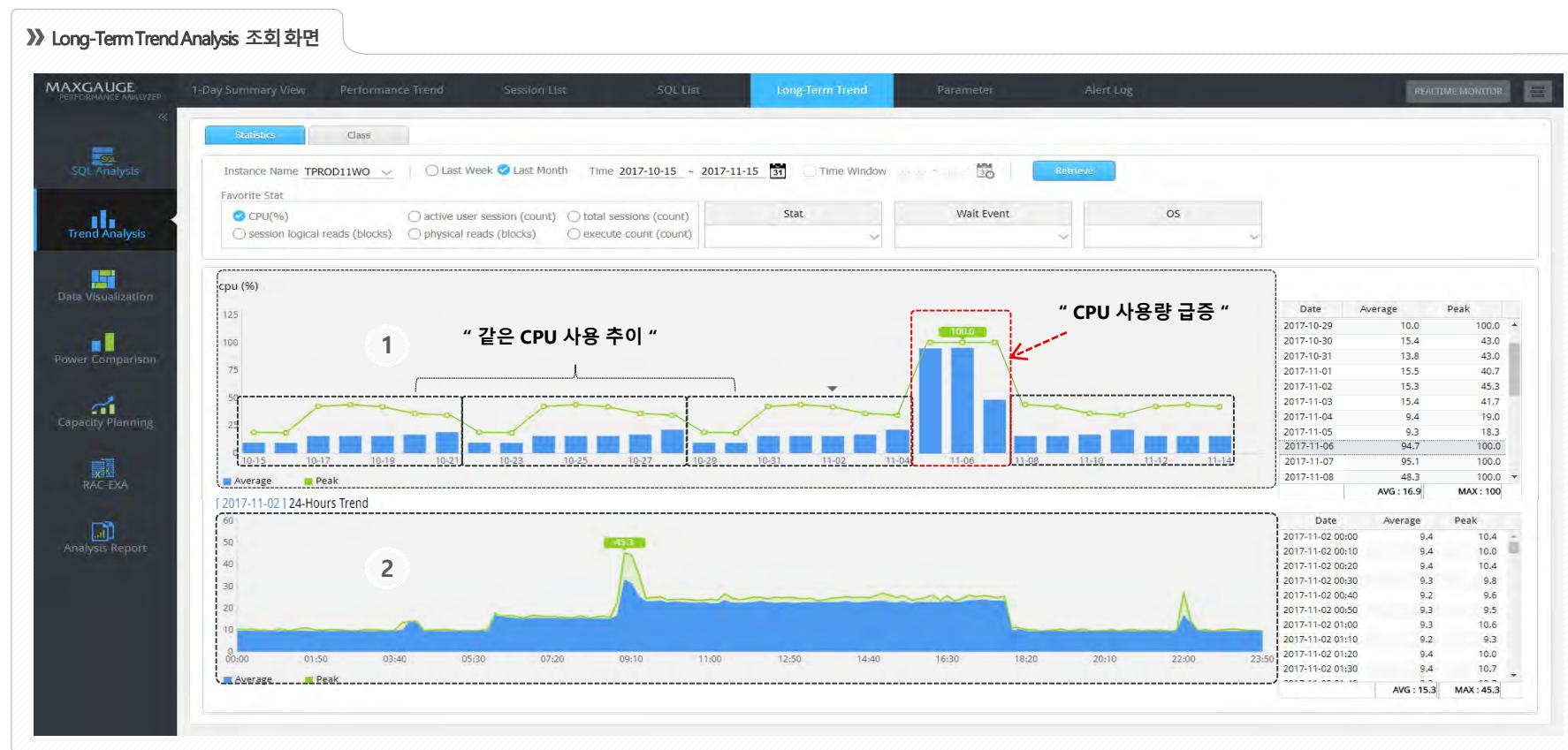
2. 데이터 분석

■ 성능 추이 확인

- └ 1달 성능 Trend 확인
- └ 부하 시점 Trend 확인

■ Trend Analysis

- └ Stat Analysis 비교



1 Long Term Analysis 결과를 분석해 보겠습니다. 화면 상단의 **성능 지표** 그래프를 통해 한 달간의 추이를 확인 할 수 있습니다. 확인 결과 일주일 동안

9~15퍼센트의 CPU 사용을 보이고 있습니다.

2 하단의 24-Hours Trend 확인 시 업무 시간에 CPU사용량 증가 후 줄어드는 일정한 형태를 보이고 있습니다. 그러나 11월5일~11월7일 CPU 사용량이 급증한 것을 확인 할 수가 있습니다. 평소와 다른 부하가 있었음을 예상할 수 있습니다.

“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

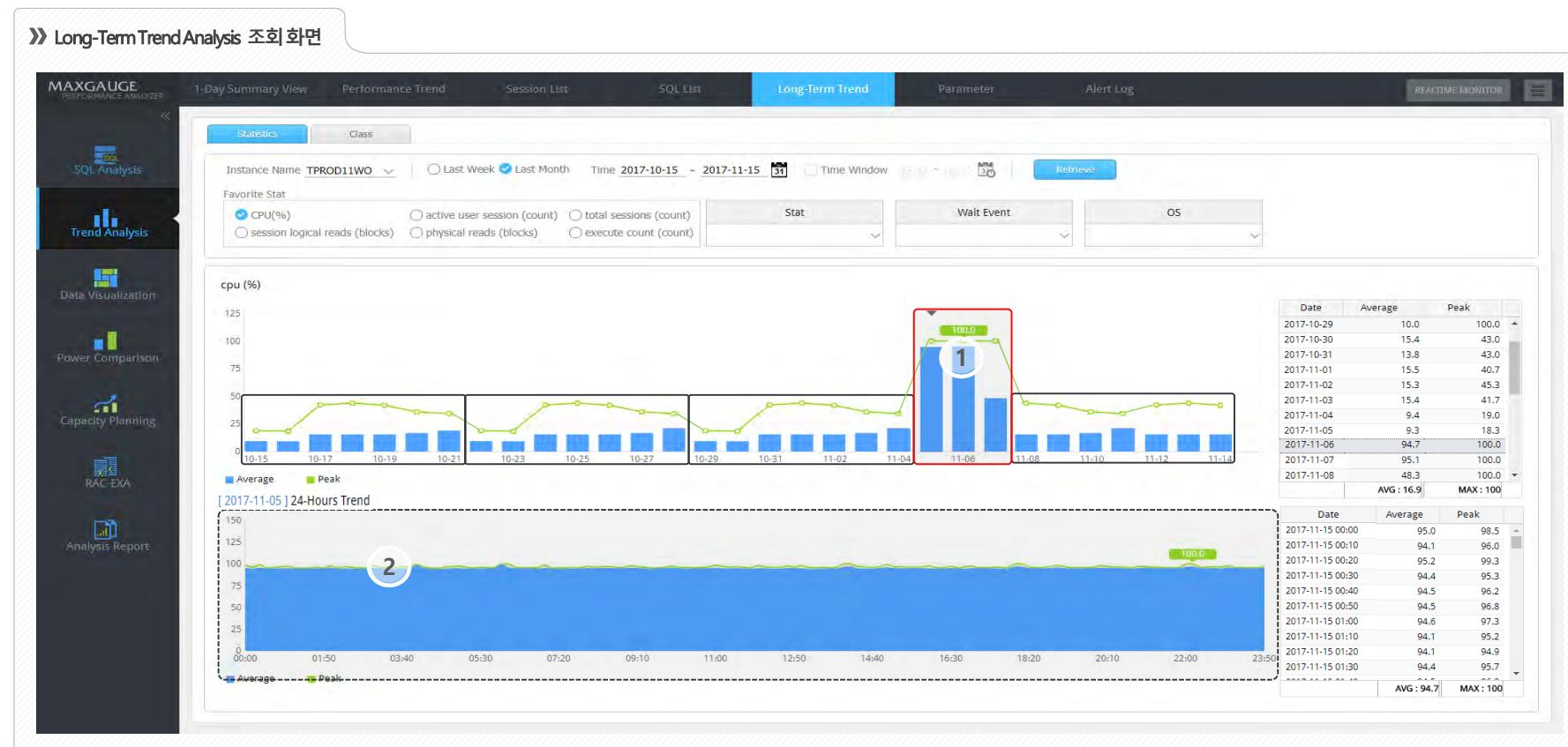
2. 데이터 분석

■ 성능 추이 확인

- └ 1달 성능 Trend 확인
- └ 부하 시점 Trend 확인

■ Trend Analysis

- └ Stat Analysis 비교



1 문제가 예상 되는 11월 5일의 막대 그래프를 선택해 보겠습니다.

2 11월 5일 00:00시 부터 CPU 사용량이 95%이상을 계속해서 유지하고 있습니다. CPU 급증 현상은 11월7일 12:00 까지 계속 되었습니다.

✓ 정확한 부하의 원인 파악을 위해 정상 시점과 부하시점의 Stat Analysis를 진행 해 보겠습니다.

“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

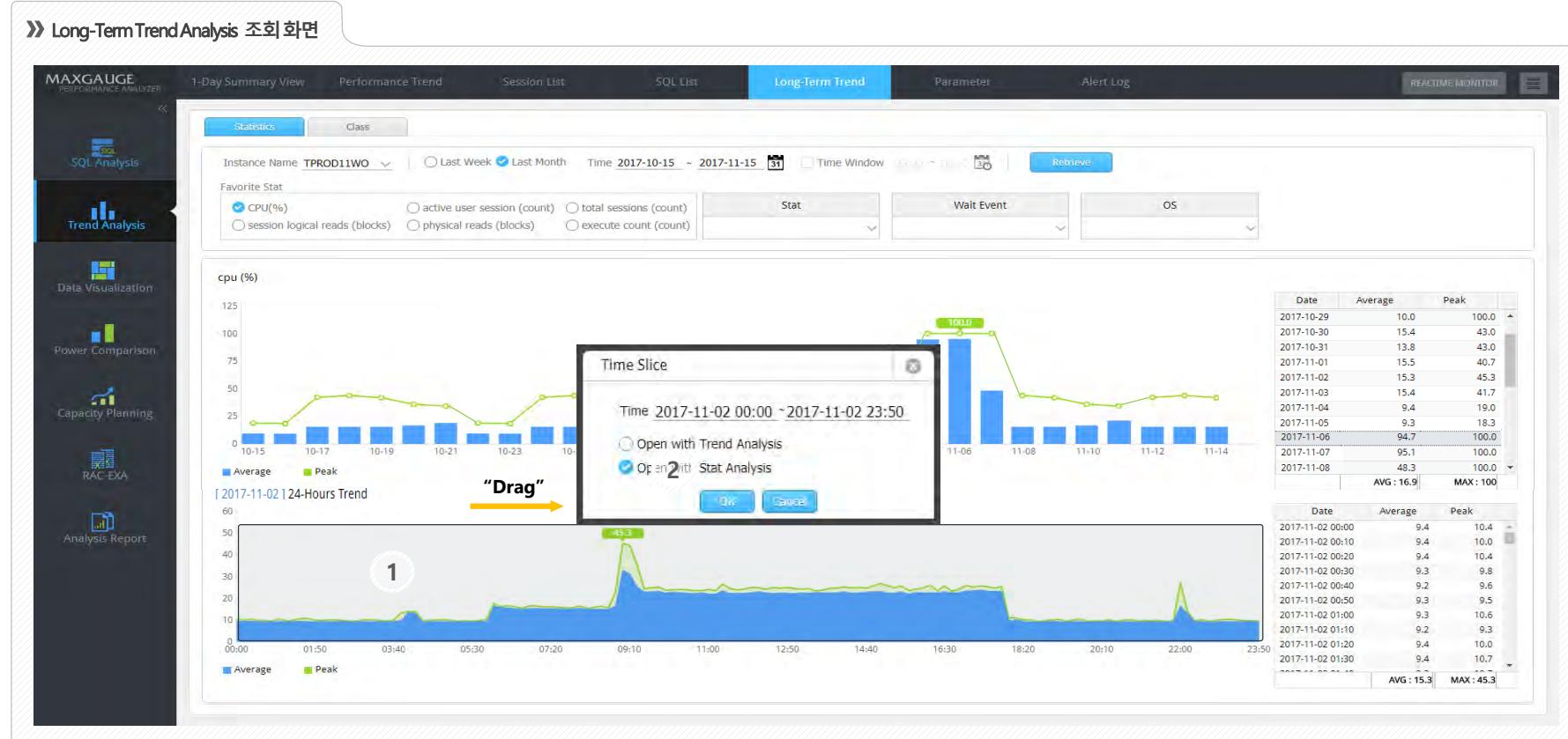
■ 성능 추이 확인

└ 1달 성능 Trend 확인

└ 부하 시점 Trend 확인

■ Trend Analysis

└ Stat Analysis 비교



1 11월 2일의 하루 동안의 Stat 정보를 확인하기 위해 24-Hour-Trend를 Drag 합니다.

2 Open with Stat Analysis를 선택합니다.

“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

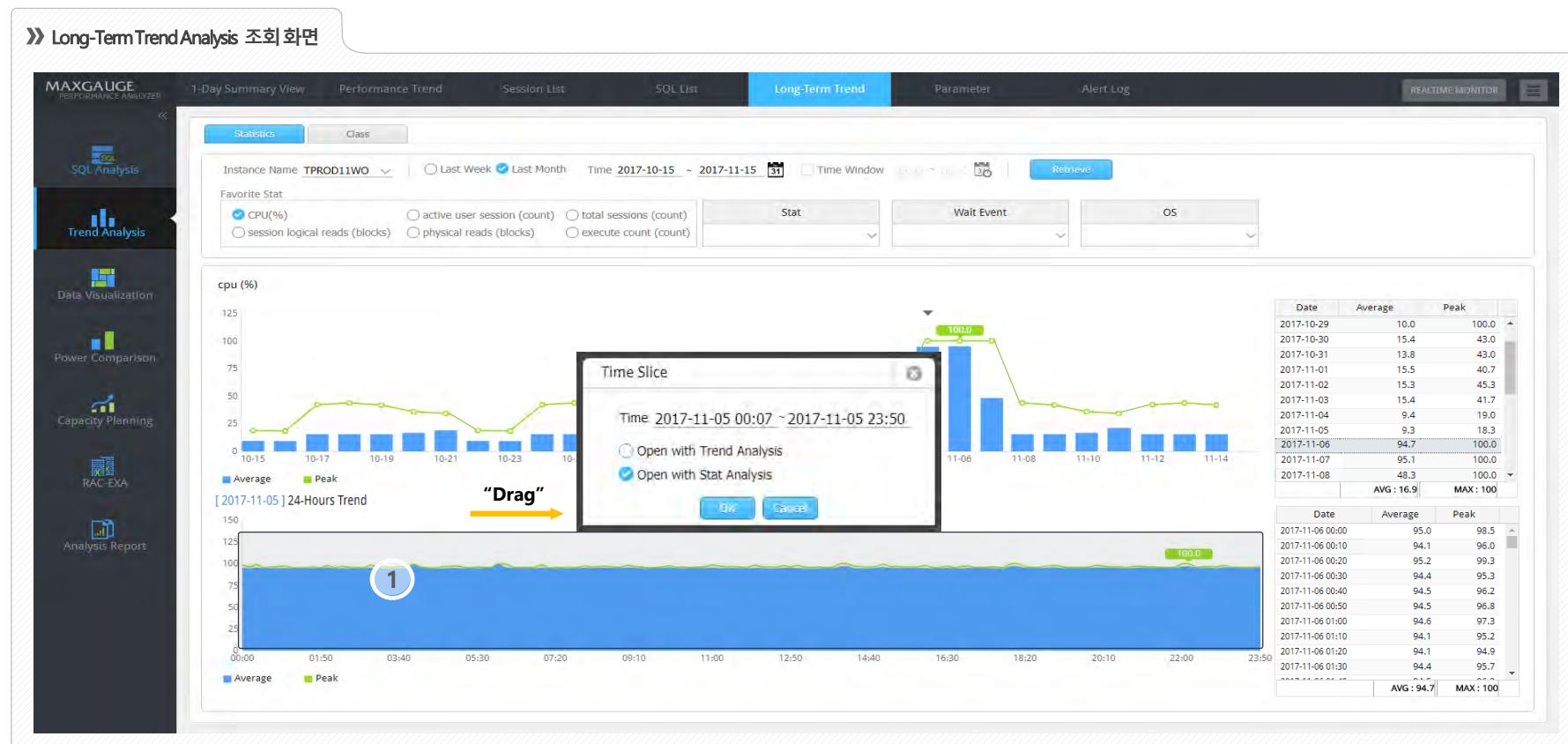
■ 성능 추이 확인

└ 1달 성능 Trend 확인

└ 부하 시점 Trend 확인

■ Trend Analysis

└ Stat Analysis 비교



① 최초 부하 시점인 11월 5일의 하루 동안의 Stat 정보를 확인하기 위해 24-Hour-Trend를 Drag 합니다.

“ 한 달간 가장 큰 부하의 원인을 알고 싶어요 ”

STEP

1. 검색 조건 설정

2. 데이터 분석

▪ 성능 추이 확인

└ 1달 성능 Trend 확인

└ 부하 시점 Trend 확인

▪ Trend Analysis

└ Stat Analysis 비교

» Long-Term Trend Analysis 조회 화면

Elapsed Time	CPU Time	Logical Reads	Physical Reads	Executions	SQL ID	Schema	Program	Module	SQL Plan Hash	Logical Reads (blocks)	CPU Time (Sec)	Executions
29qp10usqkqh0	SOE	JDBC Thin Client	Sales Rep Query	4228977748		685,161,890	5,289	4,937				
c13sma6rkr27c	SOE	JDBC Thin Client	New Order	1206466372		7,465,442	155	95,409				
a1dkahc57tu6k	DBSNMP	perl@techlinux1 (TNS...	Oracle Enterprise Ma...	1743960565		2,870,936	12	42				
0y1prvxqc2ra9	SOE	JDBC Thin Client	Browse Products	3686042051		1,422,233	101	100,835				
5vag3kqnwpv5	SYS	oracle@techlinux1 (J0...	DBMS_SCHEDULER	296924608		1,233,682	10	1				
f7rxuxzt64k87	SOE	JDBC Thin Client	New Order	0		1,122,706	88	184,215				
3fw75k1snsddx	SOE	JDBC Thin Client	New Order	494735477		1,093,476	65	102,192				
gh2g2tynpcpv1	SOE	JDBC Thin Client	Update Customer Det...	0		614,065	30	38,521				
8z3542ffmp562	SOE	JDBC Thin Client	New Order	1655552467		464,748	36	179,271				
12ajd4qc7bba	MAXGAUGE	JDBC Thin Client	JDBC Thin Client	4291714398		239,402	22	117				
7hk2m2702ua0g	SOE	JDBC Thin Client	Process Orders	1278617784		214,399	6	12,534				
g81cbrq5yamf5	SOE	JDBC Thin Client	New Order	3585448325		183,324	27	102,589				

11월2일 하루의 수행된 SQL 정보

Elapsed Time	CPU Time	Logical Reads	Physical Reads	Executions	SQL ID	Schema	Program	Module	SQL Plan Hash	Logical Reads (blocks)	CPU Time (Sec)	Executions
g81cbrq5yamf5	SOE	JDBC Thin Client	New Order	1286489376		2,772,410,474	43,173	214,276				
7ws837zynp1zv	SOE	JDBC Thin Client	New Order	2597291669		1,085,622,885	23,496	214,269				
8zz6y2yzdqjp0	SOE	JDBC Thin Client	Update Customer Det...	1005345217		908,680,972	35,183	52,132				
7t0959msvty5g	SOE	JDBC Thin Client	Browse and Update O...	335441244		435,733,199	4,580	26,727				
g81cbrq5yamf5	SOE	JDBC Thin Client	Browse and Update O...	1286489376		345,427,948	5,386	26,748				
29qp10usqkqh0	SOE	JDBC Thin Client	Sales Rep Query	3619984409		184,004,307	2,457	10,635				
74cpnuu24wmx7	SYS	oracle@weheheh1 (J0...	DBMS_SCHEDULER	1672039087		1,882,783	48	164				
7hk2m2702ua0g	SOE	JDBC Thin Client	Process Orders	2307454521		1,225,230	511	12,579				
dvb42b3hfryu	SYS	oracle@weheheh1 (J0...	DBMS_SCHEDULER	1886112120		321,204	12	6				
7hu2k3a31b6j7	SYS	oracle@weheheh1 (J0...	DBMS_SCHEDULER	0		203,132	32	464				
4y1y43113gv8f	SYS	oracle@weheheh1 (J0...	DBMS_SCHEDULER	1844266620		194,917	31	504				

11월5일 하루의 수행된 SQL 정보

✓ 11월 2일과 11월 5일 수행 된 SQL 문 비교 결과 신규로 추가된 SQL문은 없었습니다. 이는 신규로 추가된 업무가 없음을 의미합니다.

단지 11월 5일 기존 SQL들의 Execution 횟수가 평상시 보다 급증하여 Session Logical Reads, CPU가 증가한 것으로 확인됩니다.

✓ 확인 결과 CPU가 급증한 11월 5일~7일 “ 세일 이벤트 진행 ”으로 판매 업무가 집중된 것을 확인 하였습니다.

Contents

1-Day Summary View?

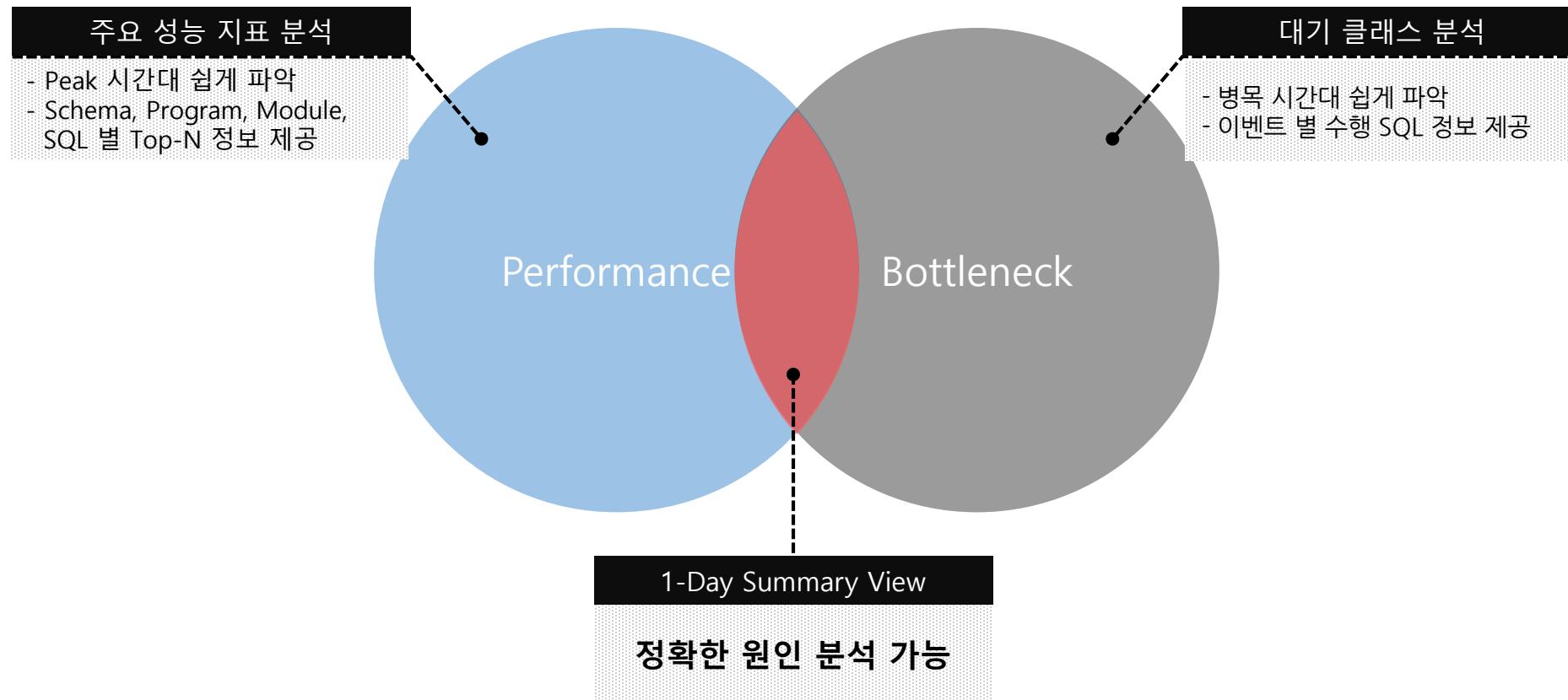
1-Day Summary View 의 활용

Case 1. 어제의 성능 저하 구간과 그 원인을 알고 싶어요.
(Performance 분석)

Case 2. Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요.
(Bottleneck 분석)

1-Day Summary View?

1-Day Summary View의 주요 기능은 무엇인가요?



1-Day Summary View은 언제 쓰나요?



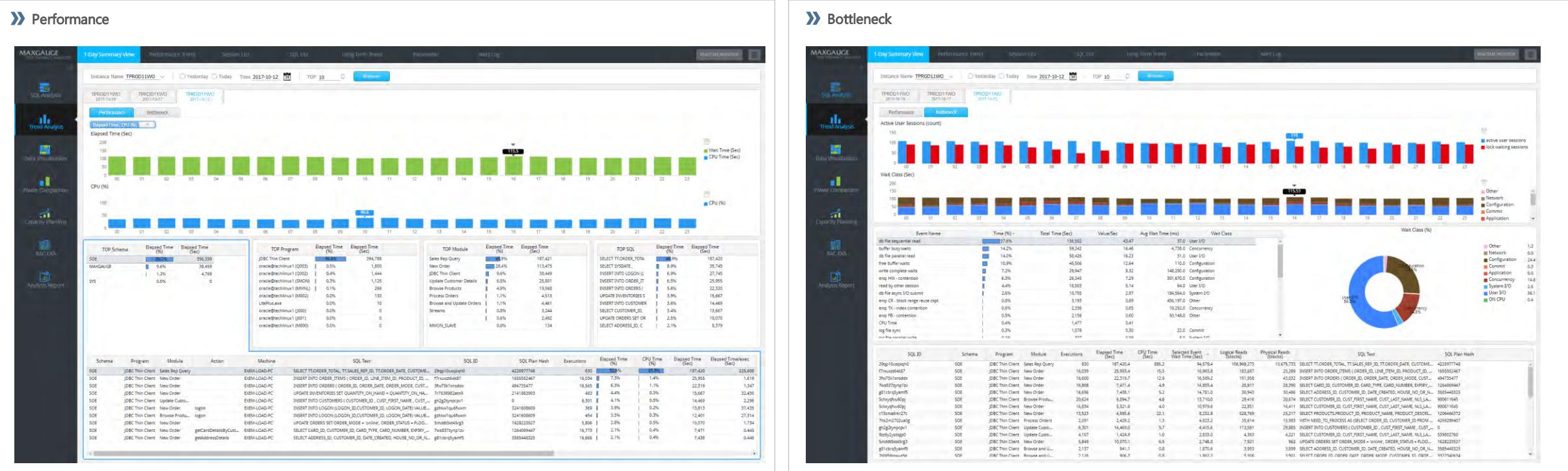
* → 1-Day Summary View

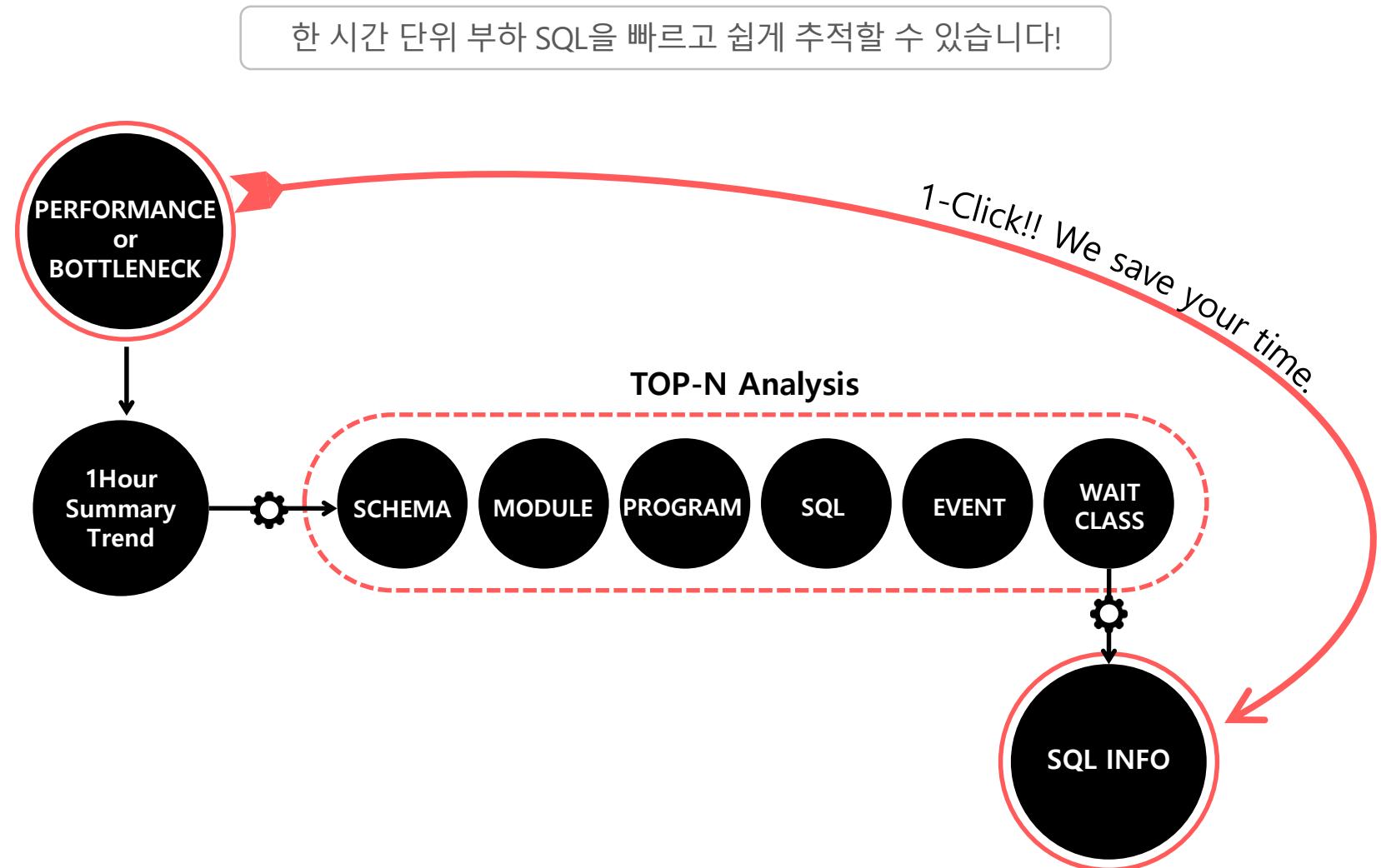
분석에 필요한 시간은 10초에 불과하며, 단 한번의 클릭으로 성능 저하 시점을 추적할 수 있습니다.
하루 동안 발생했던 성능 정보를 1시간 단위로 점검하거나, 빠르고 간편하게 장애 시점의 문제 SQL을 추적할 때 사용합니다.

1-Day Summary View?

1-Day Summary View

- 주요 성능 지표 및 대기 클래스에 대한 시간 별 평균 추이 그래프를 제공함으로써 Peak 시간대를 쉽게 확인할 수 있다.
- 시간대에 수행된 Top-N 데이터 및 대기 이벤트 별 수행 SQL을 마우스 1-클릭만으로 제공한다.
- Performance, Bottleneck의 두 가지 탭으로 구성되어 있다.
- Performance Tab**은 주요 성능 지표에 대한 시간 별 성능 추이 및 시간 대별 Top-N 정보를 제공한다.
- Bottleneck Tab**은 Wait Class에 대한 시간 별 성능 추이 및 대기 이벤트 별 수행 SQL 정보를 제공한다.
- 해당 기능은 Performance Analyzer ▶ Trend Analysis ▶ 1-Day Summary View 경로를 통해 사용할 수 있다.





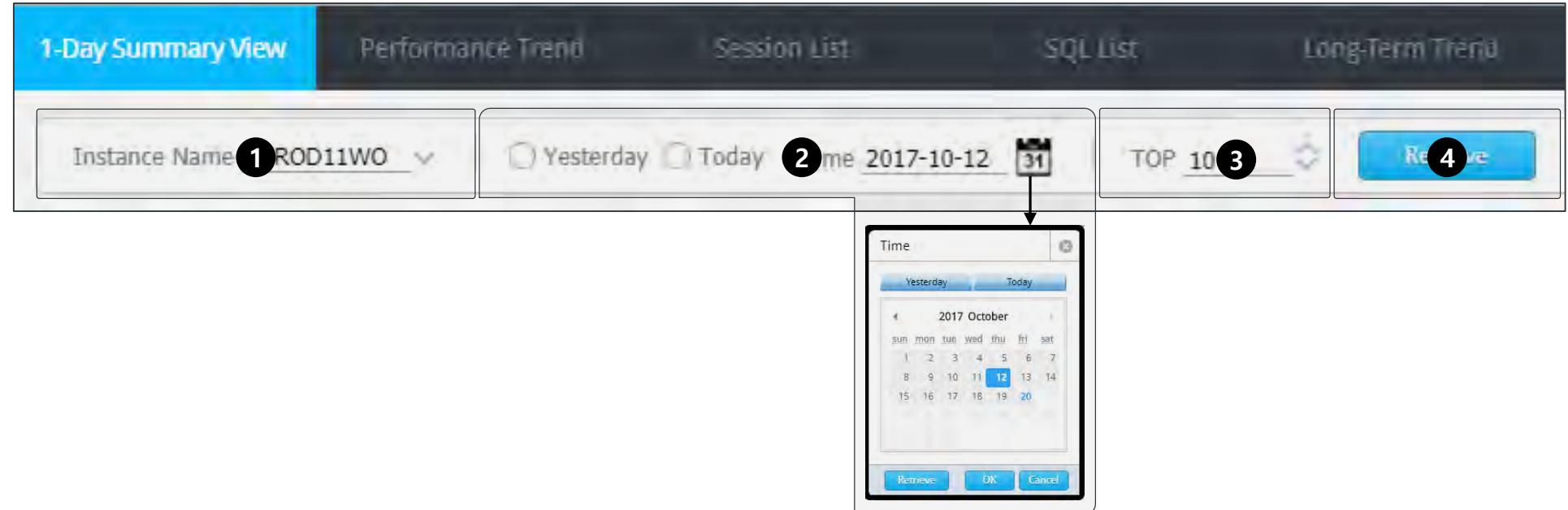
1-Day Summary View의 활용

1-Day Summary View의 사용 방법

1-Day Summary View

- 검색 조건 설정
- 데이터 분석 (Performance Tab)
- 데이터 분석 (Bottleneck Tab)

» 1-DaySummaryView검색 화면



- 1 분석을 원하는 Instance를 선택합니다.
- 2 분석 일자를 Yesterday(기본 설정 값), Today를 선택하거나, 31 모양을 눌러서 원하는 날짜를 선택합니다.
- 3 TOP-N 구간에 출력되는 SQL 수를 조절합니다.
- 4 위의 설정한 조건으로 1-Day Summary View를 실행합니다.

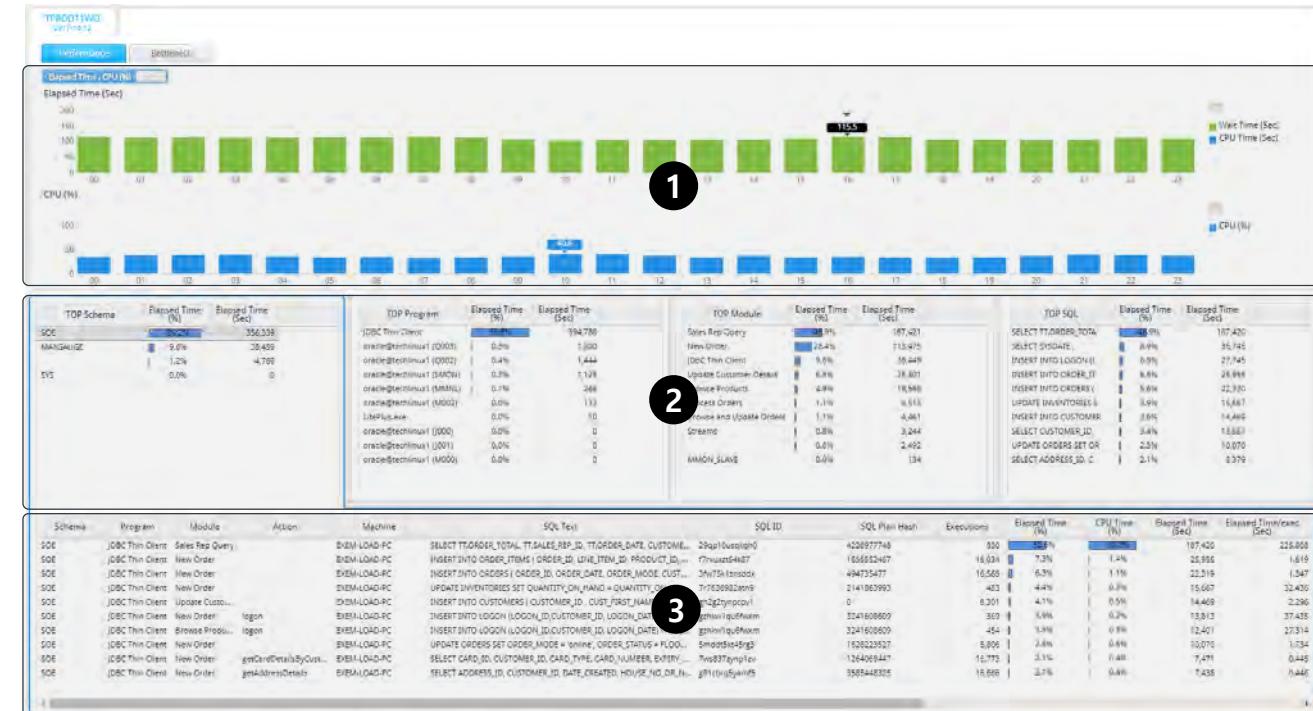
1-Day Summary View의 활용

1-Day Summary View의 사용 방법 (Performance Tab)

1-Day Summary View

- 검색 조건 설정
 - 데이터 분석 (Performance Tab)
 - 데이터 분석 (Bottleneck Tab)

» 1-Day Summary View (Performance Tab) 조회 화면



- 1 **1-Day Summary Trend** : 선택한 주요 성능 지표에 대한 시간 별 성능 추이를 제공합니다.
 - ELAPSED TIME, CPU
 - LOGICAL IO, PHYSICAL IO
 - 2 **TOP-N Area** : 1-Day Summary Trend에서 선택된 시간에 대한 Top-N(Schema, Program, Module, SQL) 정보를 제공한다.
 - 3 **SQL Info** : TOP-N Area내에서 선택한 항목과 관련된 SQL 정보를 제공한다.

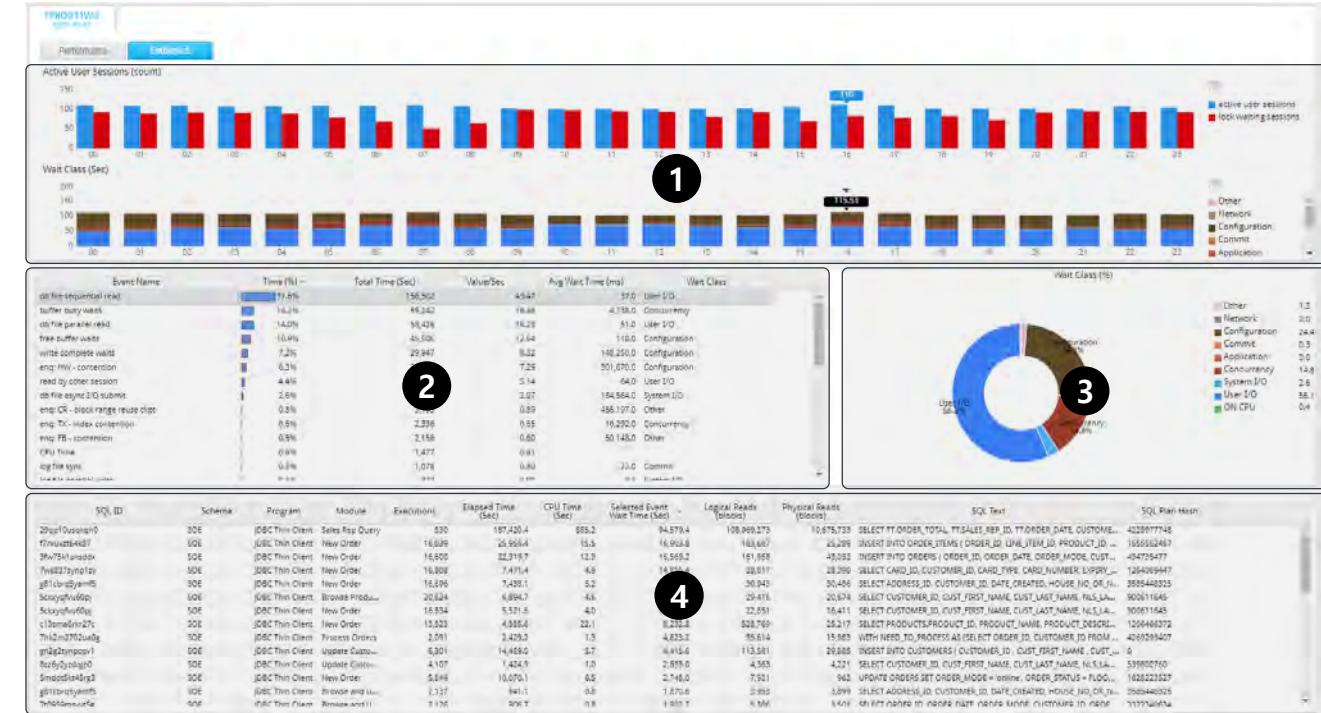
1-Day Summary View의 활용

1-Day Summary View의 사용 방법 (Bottleneck Tab)

1-Day Summary View

- 검색 조건 설정
- 데이터 분석 (Performance Tab)
- 데이터 분석 (Bottleneck Tab)

» 1-Day Summary View(Bottleneck Tab) 조회 화면



1 1-Day Summary Trend : 선택한 주요 성능 지표에 대한 시간 별 성능 추이를 제공합니다.

- Active User Session (count) : Active user session 및 Lock waiting session의 시간 별 평균 수치를 제공한다.
- Wait Class (sec) : 대기 클래스 별, 시간 별 평균 대기 시간을 제공한다.

2 Top Event Area : 1-Day Summary Trend에서 선택된 시간에 대한 Top Event 정보를 제공한다.

3 Wait Class Area : 1-Day Summary Trend에서 선택된 시간에 대한 Wait Class 별 대기 시간 비율 정보를 제공한다.

4 SQL Info : Top Event Area에서 선택한 항목과 관련된 SQL 정보를 제공한다.

실전 분석 사례 Case 1.
어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)

“어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU)
- 추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO)
- 추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적
- Logical/Physical IO 성능 저하 SQL 추적
- SQL 상세 정보 확인

» 1-Day Summary View 검색 화면

✓ 시나리오

INSTANCE : TPROD11WO
SCHEMA : MAXGAUGE
Time : 2017년 10월 24일 (Yesterday)

✓ 목표

어제의 성능 지표를 확인하여 Peak 시간대를
Top-N 정보를 확인하여 성능 저하의 원인이
되는 SQL 추출 및 성능 저하 원인 파악합니다.

- 1 비교 대상 Instance (TPROD11WO)를 선택합니다.
- 2 분석 날짜 2017-10-24(Yesterday)를 선택합니다. 캘린더 아이콘을 통해 특정 날짜 선택도 가능합니다.
- 3 출력되는 Top-N의 개수(TOP 10)를 설정합니다.
- 4 위의 설정한 조건으로 1-Day Summary View 실행합니다.

1-Day Summary View의 활용

"어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

▪ 출력 데이터 확인

- 주요 성능(Elapsed Time, CPU)

추이 및 Top-N 확인

- 주요 성능(Logical/Physical IO)

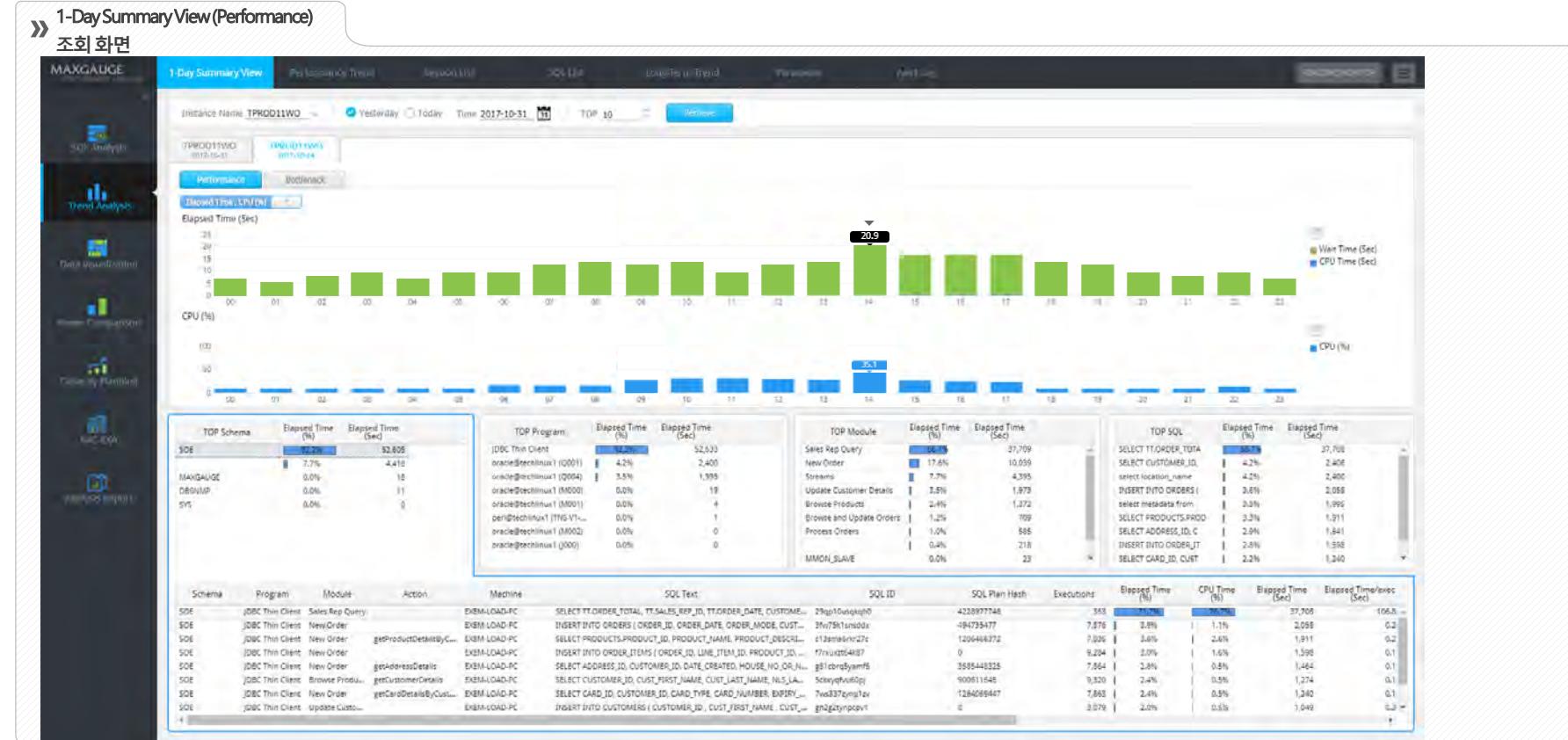
추이 및 Top-N 확인

- Elapsed Time 수치 증가의 원인 SQL 추적

- Logical/Physical IO 성능 저하

SQL 추적

- SQL 상세 정보 확인



- ✓ 상단 그래프를 통하여 Elapsed Time 과 CPU 성능에 대한 시간 별 평균 추이 그래프를 확인합니다.
- ✓ Toggle Button 을 통해 Logical IO, Physical IO 지표로 변경하여 시간 별 평균 추이 그래프를 확인합니다.
- ✓ MAX 값의 경우 그래프 상단에 별도 표시되어 각 성능에 대한 Peak 시간대를 확인할 수 있습니다.
- ✓ 시간 대 별 Top-N 정보를 그래프 하단에서 확인할 수 있습니다.
- ✓ 가장 하단 표를 통해 Top-N 데이터 별 수행 SQL 정보를 확인할 수 있습니다.

1-Day Summary View의 활용

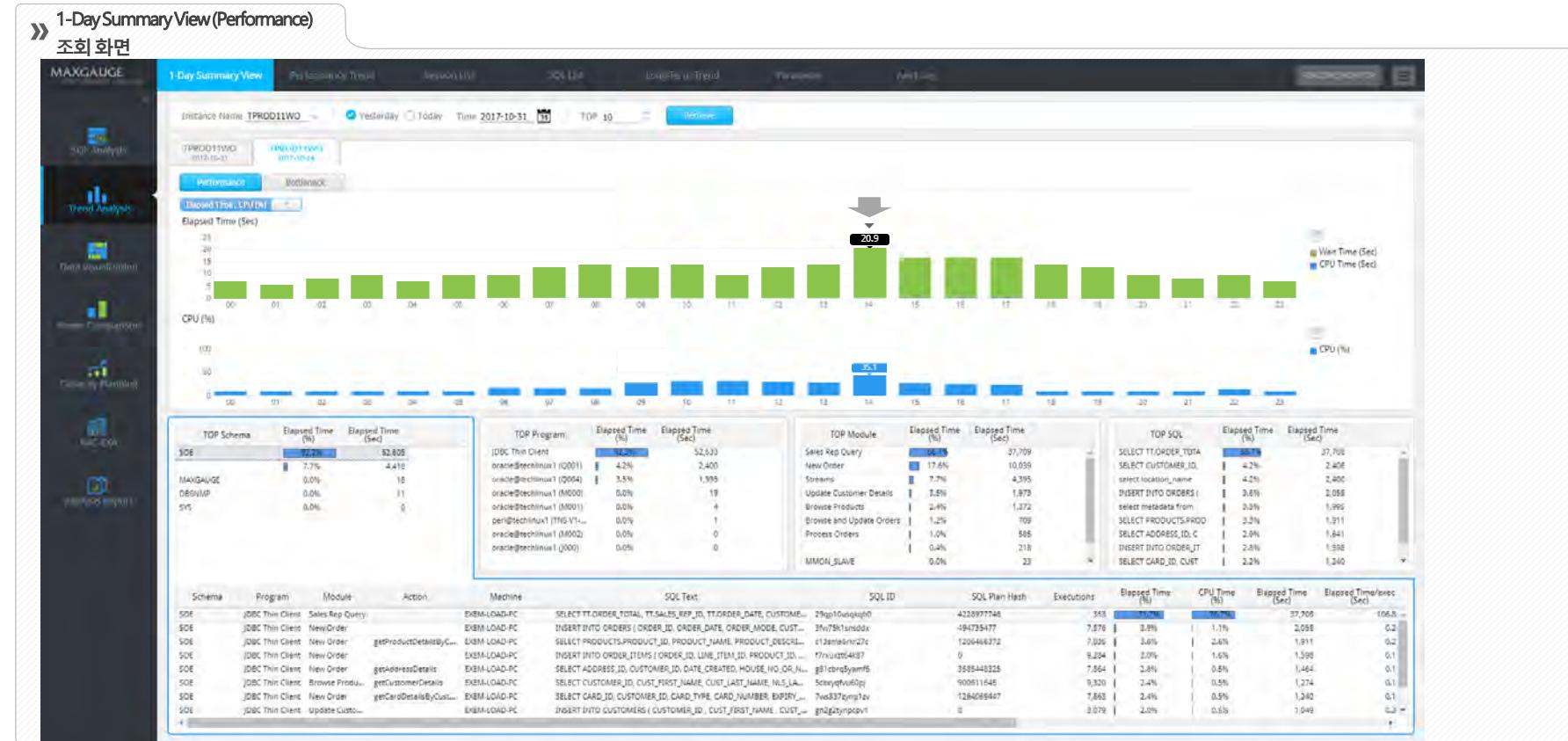
"어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU) 추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO) 추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적
- Logical/Physical IO 성능 저하 SQL 추적
- SQL 상세 정보 확인



- ✓ Elapsed Time 과 CPU 성능 지표의 시간 별 성능 추이를 확인합니다.
- ✓ Elapsed Time 지표 중 초록색은 Wait Time을, 파란색은 CPU Time 을 나타냅니다.
- ✓ Elapsed Time 지표 확인 시 09 ~ 17 에 높은 수치를 기록하고 있음을 알 수 있습니다.
- ✓ Wait Time 20.8(Sec), CPU Time 0.1(Sec) 로 Elapsed Time 지표에 대한 Peak 시간대는 14:00~14:59 임을 확인할 수 있습니다.

1-Day Summary View의 활용

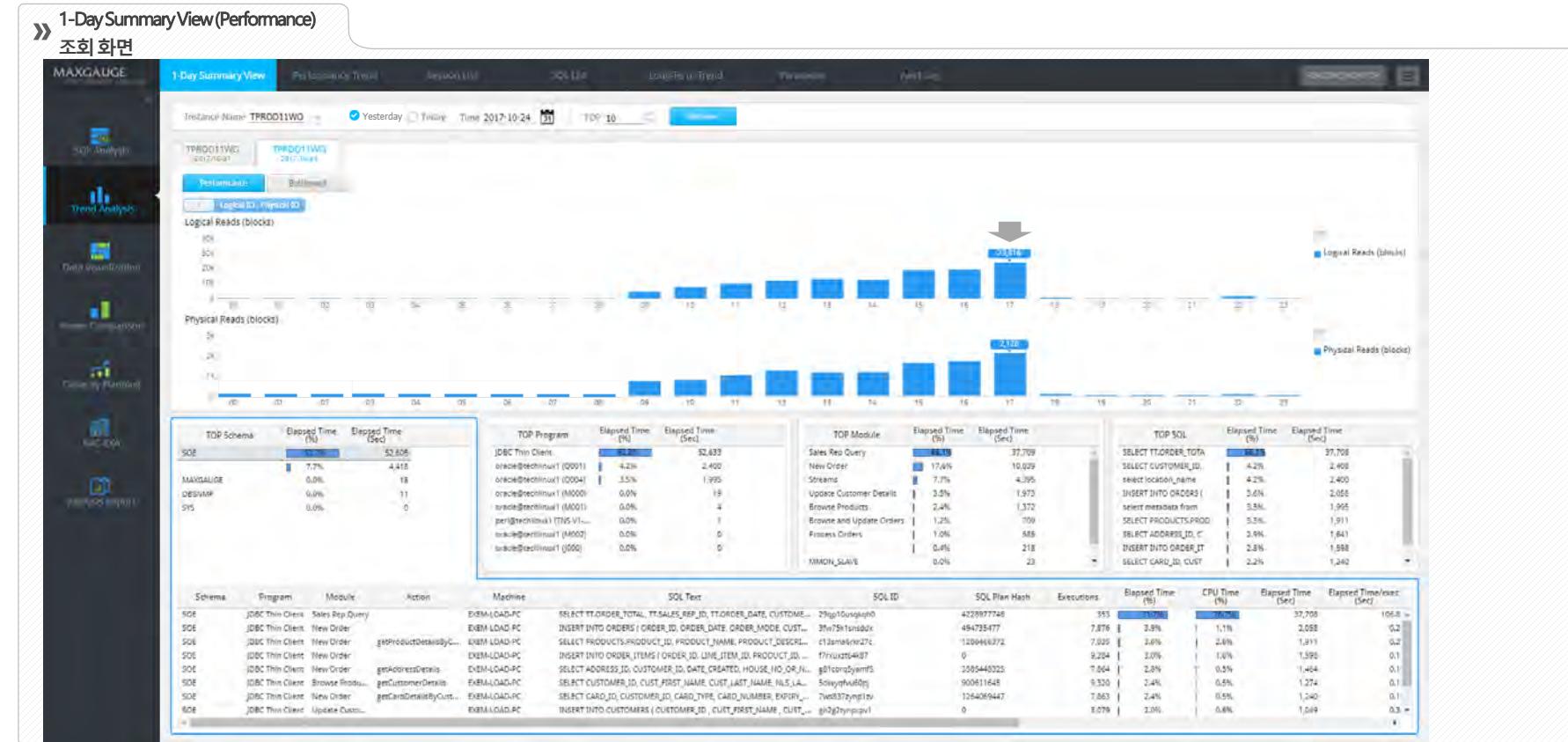
"어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU)
- 추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO)
- 추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적
- Logical/Physical IO 성능 저하 SQL 추적
- SQL 상세 정보 확인



- ✓ Toggle Button 을 클릭하여 Elapsed Time, CPU 성능 지표에서 Logical IO, Physical IO 성능 지표로 변경 조회 가능합니다.
- ✓ Logical IO 와 Physical IO 의 성능 지표에 대한 시간 별 성능 추이를 확인합니다.
- ✓ Logical/Physical IO 모두 오전 9시부터 오후 6시 이전까지 계속해서 수치가 증가하는 것을 확인할 수 있습니다.
- ✓ Logical IO 에 대해 23,816(blocks) 로 Peak 시간대는 17:00~17:59 임을 확인할 수 있습니다.
- ✓ Physical IO 에 대해 2,128(blocks) 로 Logical IO 와 동일하게 17:00~17:59 이 Peak 시간대 임을 확인할 수 있습니다.

1-Day Summary View의 활용

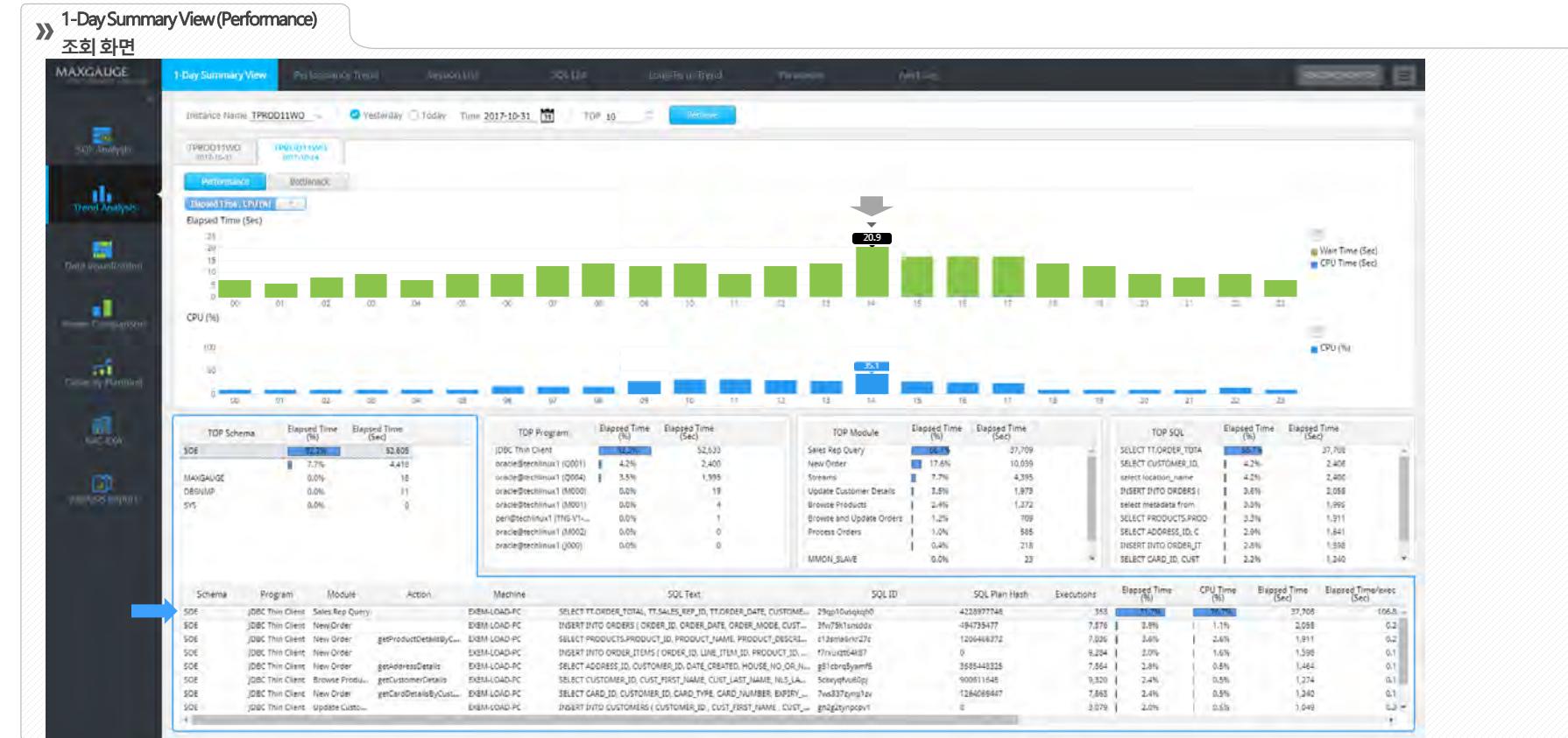
"어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU)
추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO)
추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적
- Logical/Physical IO 성능 저하
SQL 추적
- SQL 상세 정보 확인



- ✓ Elapsed Time에 대한 Peak 시간대의 Top Schema, Program, Module, SQL 은 각각 SOE, JDBC Thin Client, Sales Rep Query, SELECT TT.ORDER_TOTA~ 임을 확인 할 수 있습니다.
- ✓ Top-N 정보를 통해 하나의 SQL 이 Elapsed Time 의 절반 이상의 비율인 66.1% 차지하는 것을 확인할 수 있습니다.
- ✓ 이 SQL 은 Top-N에 해당하는 Schema, Program, Module 에 해당하는 SOE, JDBC Thin Client, Sale Rep Query이며, "SQL ID : 29qp10usqkqh0" 입니다.

1-Day Summary View의 활용

"어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)"

STEP

1. 검색 조건 설정

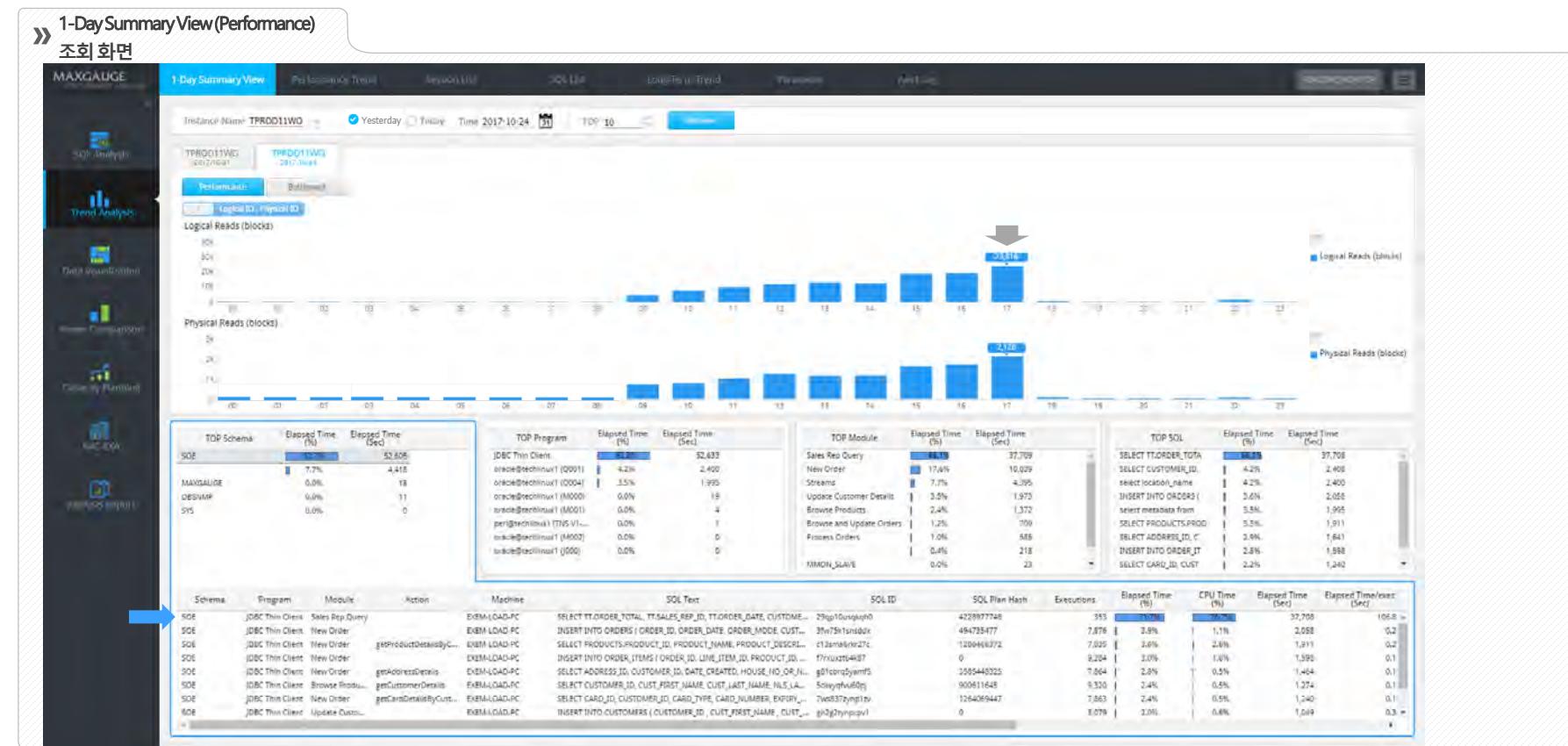
2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU)
- 추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO)
- 추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적

Logical/Physical IO 성능 저하

SQL 추적

SQL 상세 정보 확인



- ✓ Logical IO 와 Physical IO 에 대한 Peak 시간대의 Top Schema, Program, Module, SQL 은 각각 SOE, JDBC Thin Client, Sales Rep Query, SELECT TT.ORDER_TOTA~ 임을 확인할 수 있습니다.
- ✓ Top-N 정보를 통해 하나의 SQL 이 Logical IO 와 Physical IO 에 대해 각각 97.3(%), 94.5% 로 대다수를 차지하는 것을 확인할 수 있습니다.
- ✓ 이 SQL 은 "SQL ID : 29qp10usqkqh0" 로 Elapsed Time 성능에 대해 Top SQL 로 조회되었던 SQL 과 동일한 SQL 임을 확인할 수 있습니다.

“어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)”

1-Day Summary View의 활용

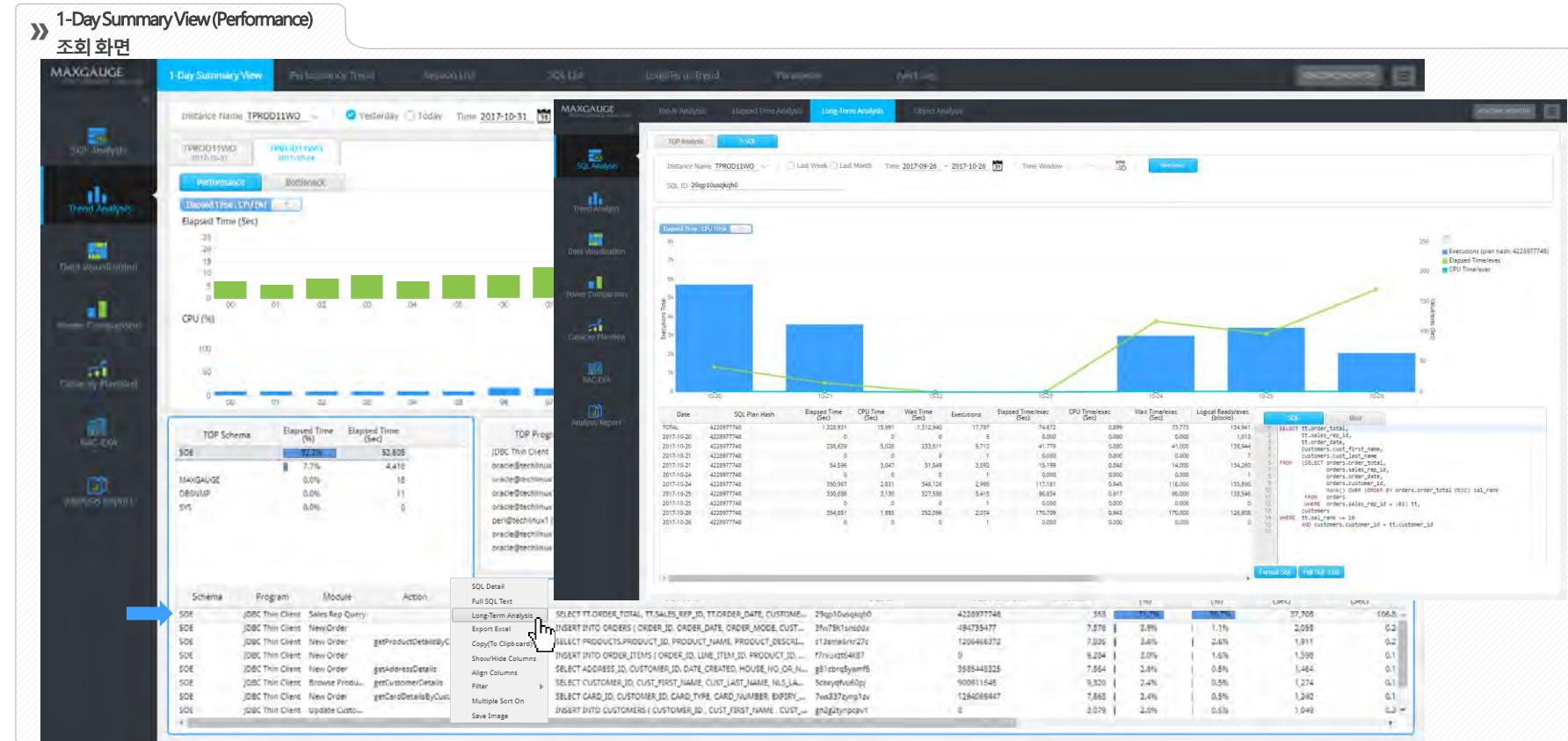
STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
 - 주요 성능(Elapsed Time, CPU)
추이 및 Top-N 확인
 - 주요 성능(Logical/Physical IO)
추이 및 Top-N 확인
 - Elapsed Time 수치 증가의 원
인 SQL 추적
 - Logical/Physical IO 성능 저하
SQL 추적
 - **SQL 상세 정보 확인**

SQL 상세 정보 확인



- ✓ 해당 SQL 을 그리드에서 선택 후 우클릭을 통해 Long-Term Analysis 를 선택하여 “SQL ID : 29qp10usqkqh0” 의 지난 일주일 동안의 일별 성능 추이를 확인할 수 있습니다.
 - ✓ Plan 변경된 이력 없이 모두 동일한 Plan Hash Value 를 가지는 것을 확인할 수 있으며, SQL Text 와 Bind 등 상세정보를 확인할 수 있습니다.
 - ✓ 이 SQL 은 10/24(Yesterday) 이 아닌 다른 날짜에도 해당 SQL 의 평균 Elapsed Time 및 IO 수치가 높은 것을 확인할 수 있습니다.

1-Day Summary View의 활용

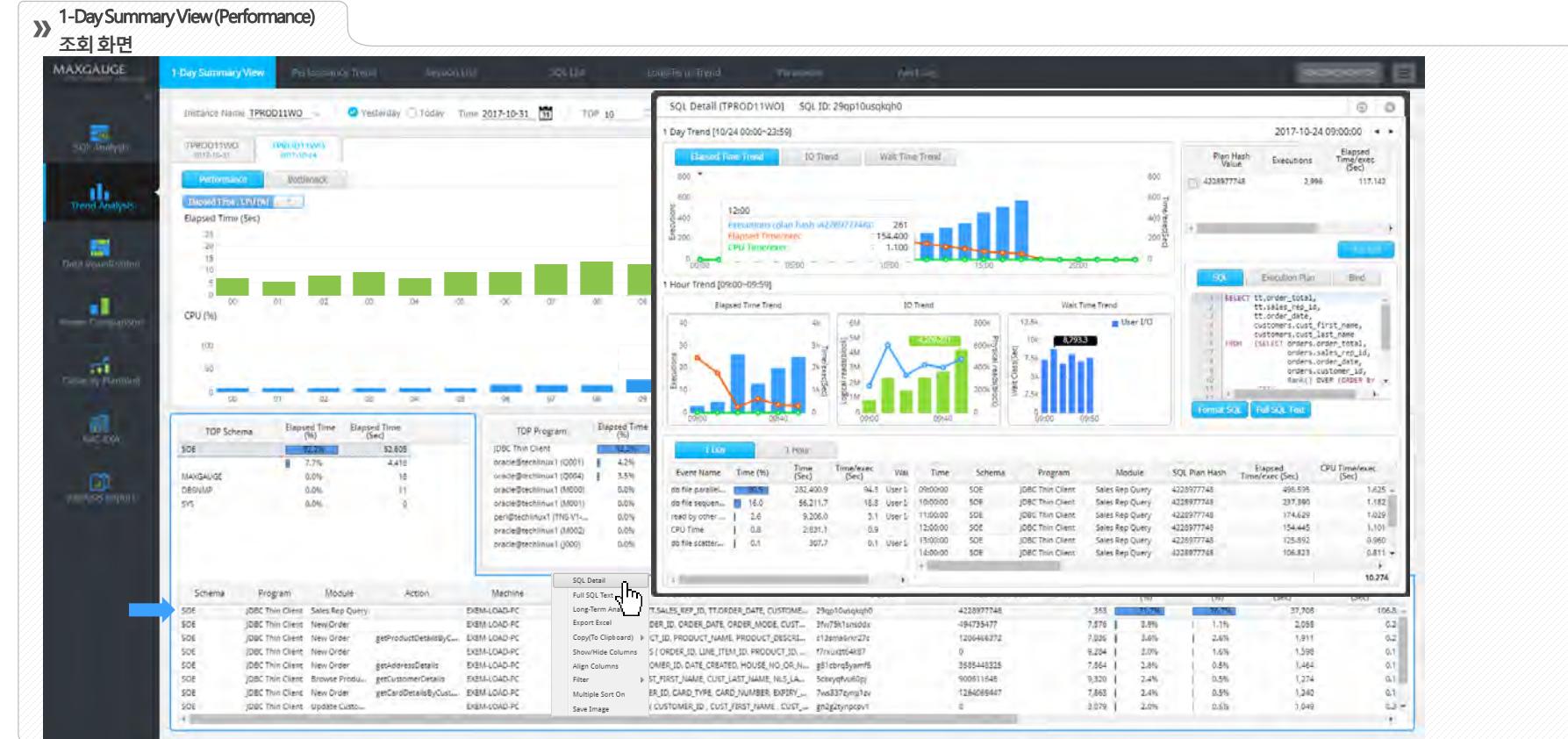
“어제의 성능 저하 구간과 그 원인을 알고 싶어요. (Performance 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- 주요 성능(Elapsed Time, CPU) 추이 및 Top-N 확인
- 주요 성능(Logical/Physical IO) 추이 및 Top-N 확인
- Elapsed Time 수치 증가의 원인 SQL 추적
- Logical/Physical IO 성능 저하 SQL 추적
- SQL 상세 정보 확인



- ✓ “SQL ID : 29qp10usqkqh0” 인 SQL 이 다른 SQL 에 비해 평균 Elapsed Time 이 길고 IO 량이 많은 것을 확인할 수 있습니다.
- ✓ 해당 SQL 을 그리드에서 선택 후 우클릭을 통해 SQL Detail 을 선택하여 SQL 상세 정보를 확인할 수 있습니다.
- ✓ 선택한 SQL 에 대한 시간 별, 10분 별 Elapsed Time, IO, Wait Time 추이 및 10분 단위의 상세 성능 이력 정보를 확인할 수 있습니다.
- ✓ 9:00 부터 18:00 이전까지 계속해서 수행횟수가 많아지면서 Elapsed Time 과 IO 수치가 계속해서 증가한 것을 확인할 수 있습니다.
- ✓ SQL 수행 시 db file parallel read 대기 이벤트를 포함하여 User I/O 가 많이 발생하는 것도 확인할 수 있습니다.

실전 분석 사례 Case 2.
Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

» 1-Day Summary View(Bottleneck)
조회 화면

1-Day Summary View

Performance Trend Session List SQL List Long-Term Trend

Instance Name **1** TPROD11WO | Yesterday Today **2** 2017-10-12 **3** TOP 10 **4** Retr

✓ 시나리오

INSTANCE : TPROD11WO
SCHEMA : MAXGAUGE
Time : 2017년 10월 12일

✓ 목표

Peak 시간대에 발생하는 대기 이벤트를 확인하고 해당 이벤트를 일으키는 대상 SQL 파악

- 1 비교 대상 Instance (TPROD11WO)를 선택합니다.
- 2 분석 날짜 2017-10-12를 선택합니다. 캘린더 아이콘을 통해 특정 날짜 선택도 가능합니다.
- 3 출력되는 Top-N의 개수(TOP 10)를 설정합니다.
- 4 위의 설정한 조건으로 1-Day Summary View 실행합니다.

1-Day Summary View의 활용

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"

STEP

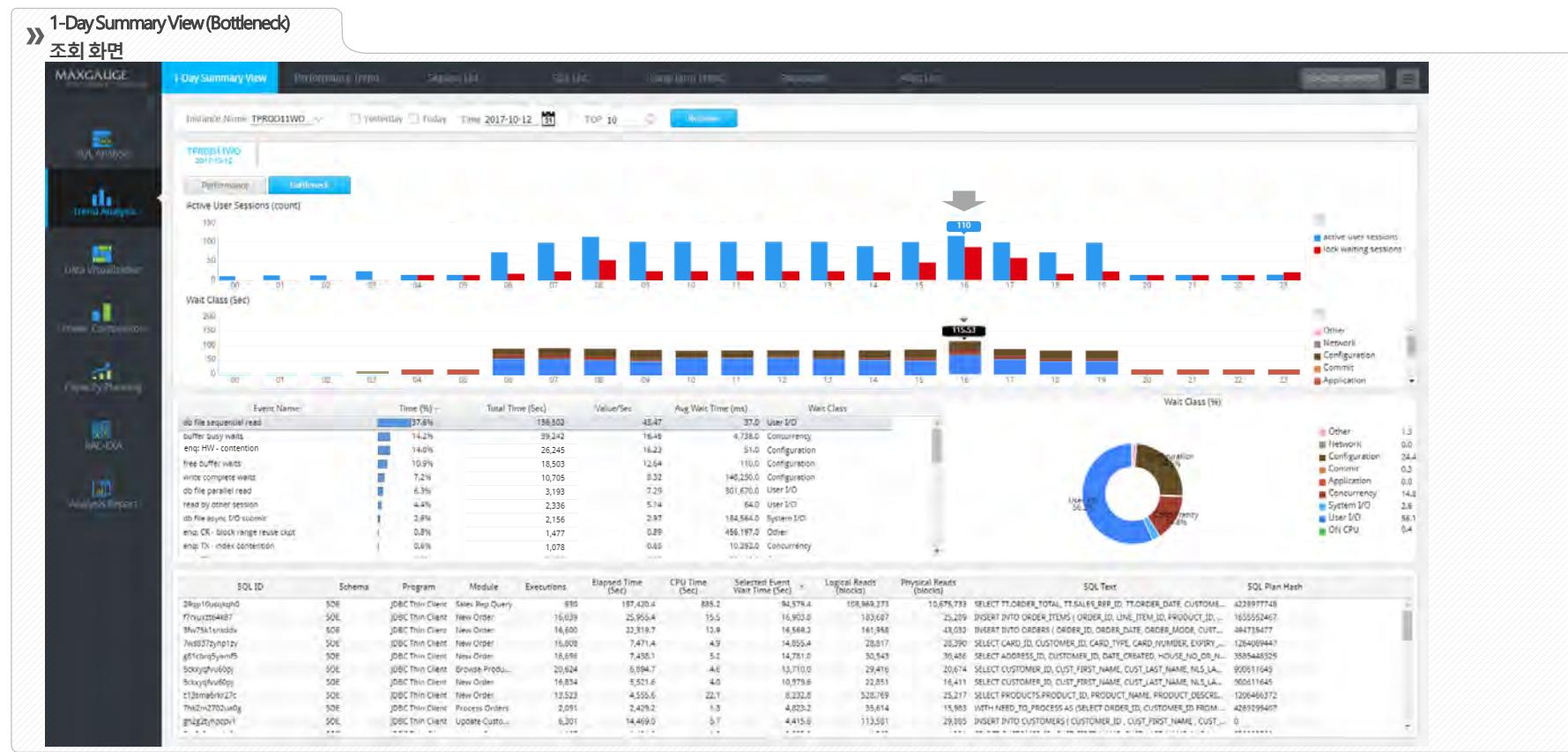
1. 검색 조건 설정

2. 데이터 분석

▪ 출력 데이터 확인

▪ Wait Class 추이 및 비율 확인

▪ Top Event 확인

▪ 분석 대상 SQL 추출
(buffer busy waits)▪ SQL 상세 정보 확인
(buffer busy waits)▪ 분석 대상 SQL 추출
(enq: HW – contention)▪ SQL 상세 정보 확인
(enq: HW – contention)

- ✓ 상단 그래프를 통하여 Active Session 과 Wait Class 성능 지표의 시간 별 추이를 확인합니다.
- ✓ MAX 값의 경우 그래프 상단에 별도 표시되어 각 성능에 대한 Peak 시간대를 찾을 수 있습니다.
- ✓ 지표를 클릭하여 해당 시간대의 Top Event 정보와 Wait Class 별 대기 시간 비율을 확인할 수 있습니다.
- ✓ Top Event 목록에서 이벤트 선택 시 해당 이벤트를 대기한 SQL 목록을 하단 SQL Info 영역에서 확인할 수 있습니다.

1-Day Summary View의 활용

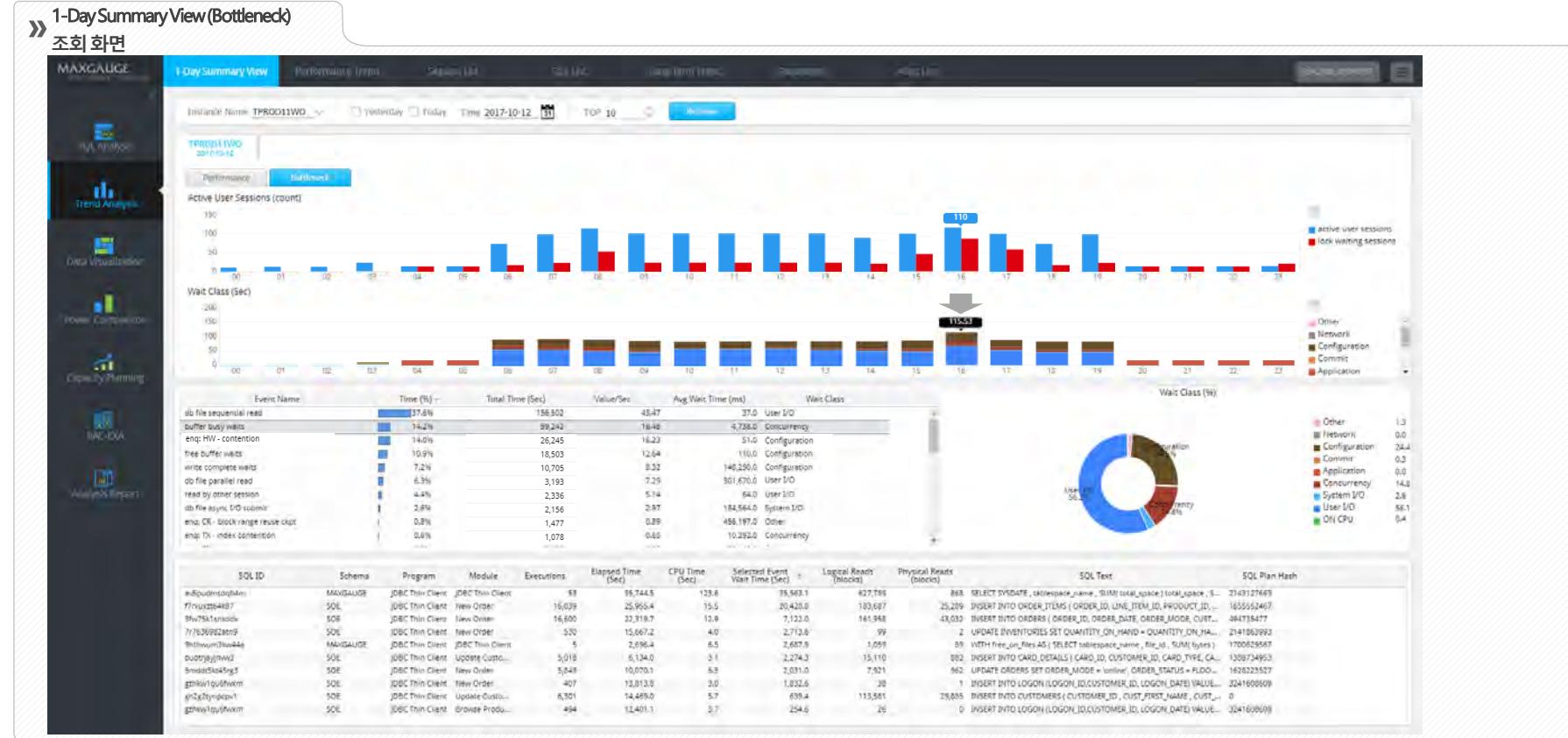
STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"



- ✓ Active User Session 은 Lock Waiting Sessions 을 제외한 수치의 시간 별 평균을 표시합니다.
- ✓ Wait Class 비율로 볼 때 User I/O, Configuration, Concurrency 항목에 해당하는 Event 의 발생 비율이 높으며 56.2% 이 User I/O 로 가장 많은 비율을 차지하는 것을 확인할 수 있습니다.
- ✓ 16:00~16:59 시간대에 Active User Sessions 은 110개, Lock Waiting Sessions 은 82 개로 락 세션 수가 급격히 늘어났으며 Wait Class 도 115.53 (Sec) 로 최대값으로 Peak 시간대임을 확인할 수 있습니다.

1-Day Summary View의 활용

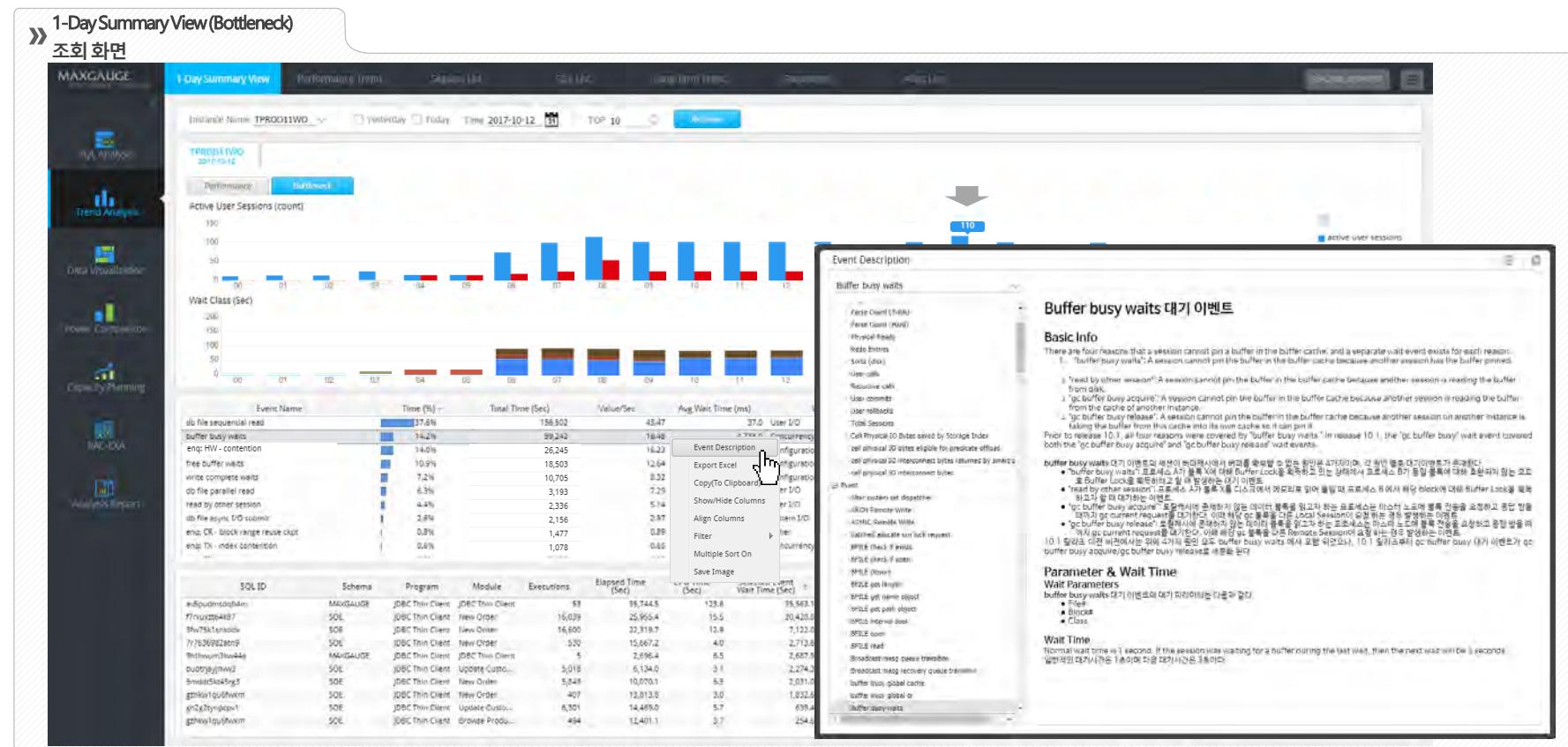
"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)



- ✓ Peak 시간대 또는 분석을 원하는 시간의 지표를 클릭하여 해당 시간대에 발생한 Top Event 를 확인할 수 있습니다.
- ✓ Peak 시간대의 Top Event 는 db file sequential read, buffer busy waits, enq: HW - contention 등이 있습니다.
- ✓ Top Event에서 우클릭 연계 기능(Event Description)으로 해당 대기 이벤트에 대한 설명을 확인할 수 있습니다.

1-Day Summary View의 활용

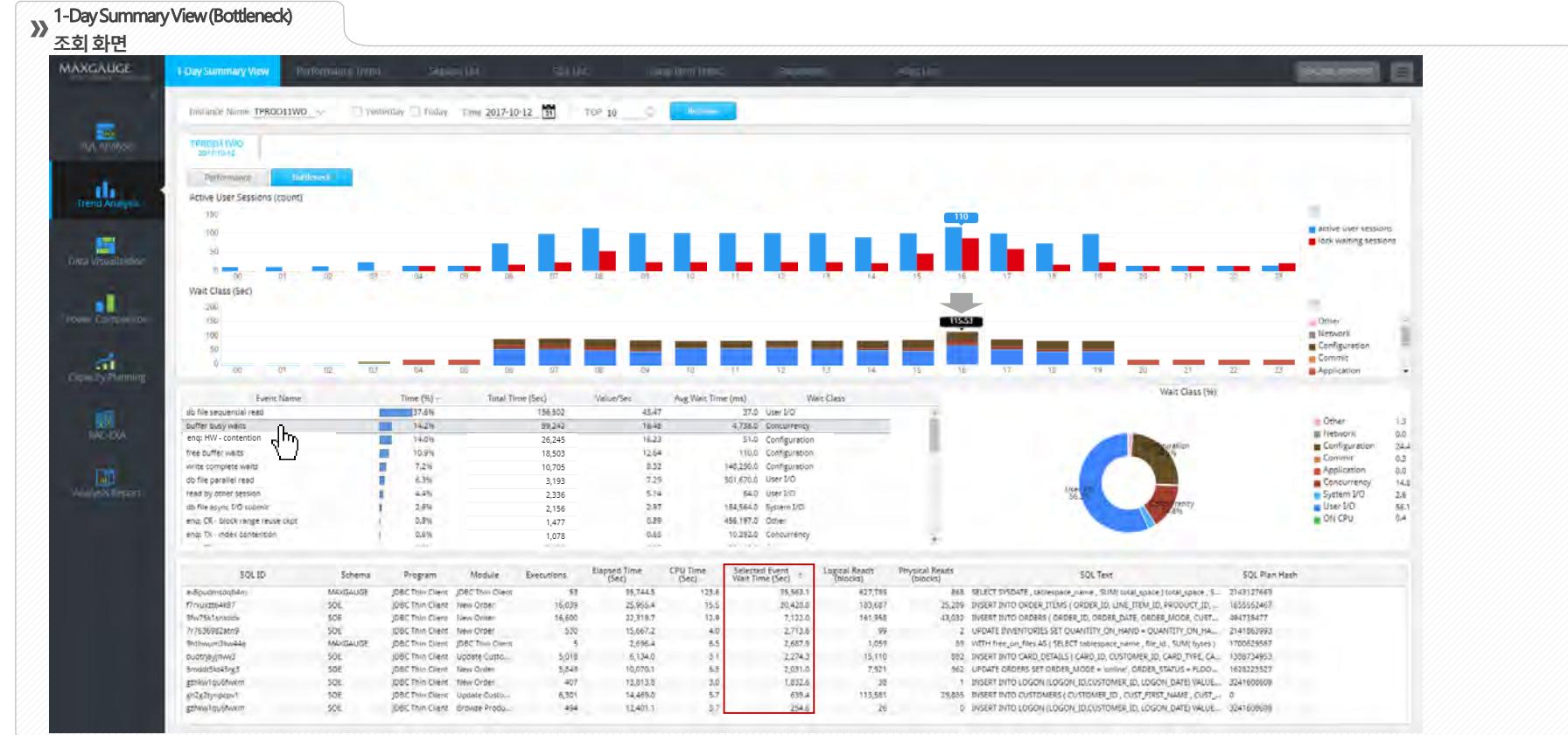
STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"



- ✓ Top Event 중 하나를 선택하여 해당 이벤트를 대기한 SQL 목록을 확인할 수 있습니다.
- ✓ buffer busy waits 이벤트를 선택하여 이 이벤트를 대기한 SQL의 SQL ID, Executions, Elapsed Time, CPU Time, Selected Event Wait Time (Sec), Logical Reads, Physical Reads, SQL Text, SQL Plan Hash 정보를 확인할 수 있습니다.
- ✓ Selected Event Wait Time (Sec) 수치가 높은 "SQL ID : av5pudmsdqh4m, f7rxuxzt64k87"에 해당하는 2개의 SQL이 특히 buffer busy wait 이벤트를 대기한 시간이 긴 것을 확인할 수 있습니다.

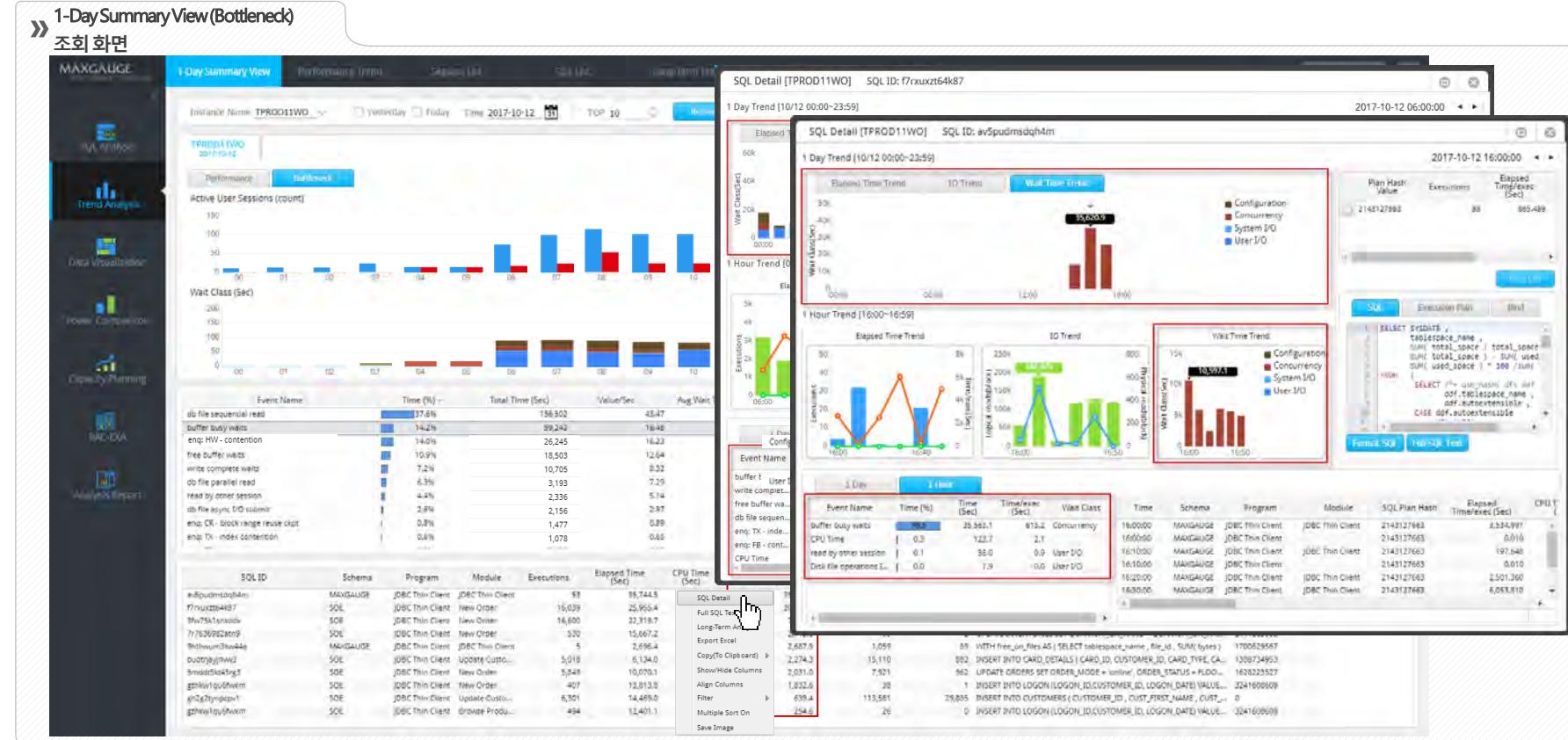
1-Day Summary View의 활용

STEP

- 검색 조건 설정
- 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)**
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"



- ✓ SQL Detail 기능 연계를 통해 "SQL ID : av5pudmsdqh4m, f7rxuxzt64k87" 의 상세 Wait Time 정보를 확인할 수 있습니다.
- ✓ 시간 별, 10분 별 Wait Time 추이를 확인할 수 있으며, 해당 SQL 이 대기한 이벤트 내역에 대해 시간 단위, 10분 단위로 Summary 된 정보를 확인 할 수 있습니다.
- ✓ Wait Time 정보 외에도 주요 성능 정보를 포함하여 SQL Text, Plan, Bind 등의 정보를 표시하므로 SQL 분석을 통하여 해당 SQL 의 문제를 파악할 수 있습니다.

1-Day Summary View의 활용

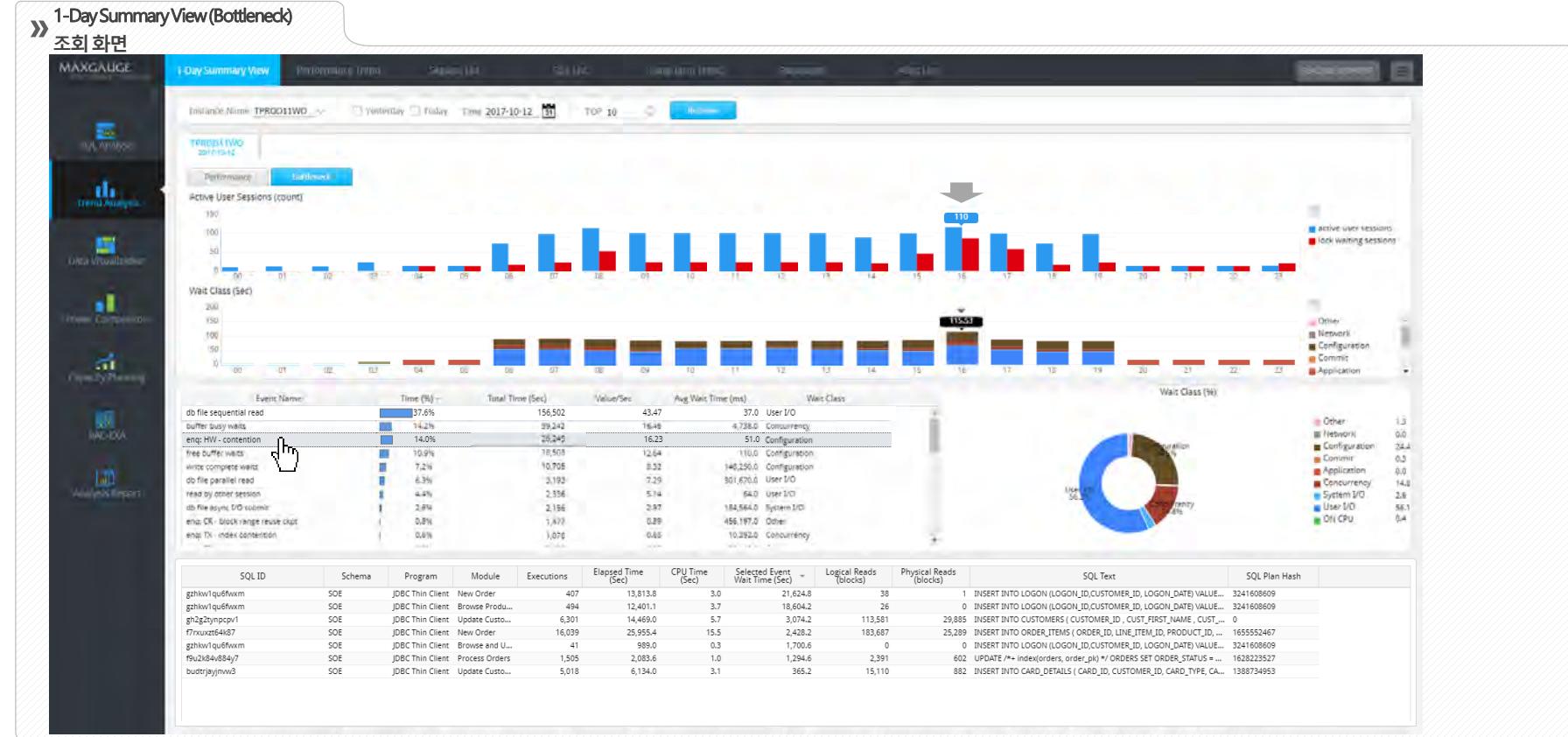
STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"



- ✓ Peak 시간대인 16:00~16:59 에 Lock Waiting Sessions 수가 급증한 것을 확인할 수 있습니다.
- ✓ Lock Waiting Sessions 가 급증한 시간대에 enq: HW – contention, enq: CR – block range reuse ckpt, enq: TX – index contention 등의 이벤트를 확인할 수 있습니다.
- ✓ 그 중 가장 높은 비율의 enq: HW – contention 이벤트를 클릭하였을 때, Selected Event Wait Time (Sec) 수치가 높은 "SQL ID : gzhkw1qu6fwxm"에 해당하는 SQL 이 특히 대기 시간이 긴 것을 확인할 수 있습니다.

1-Day Summary View의 활용

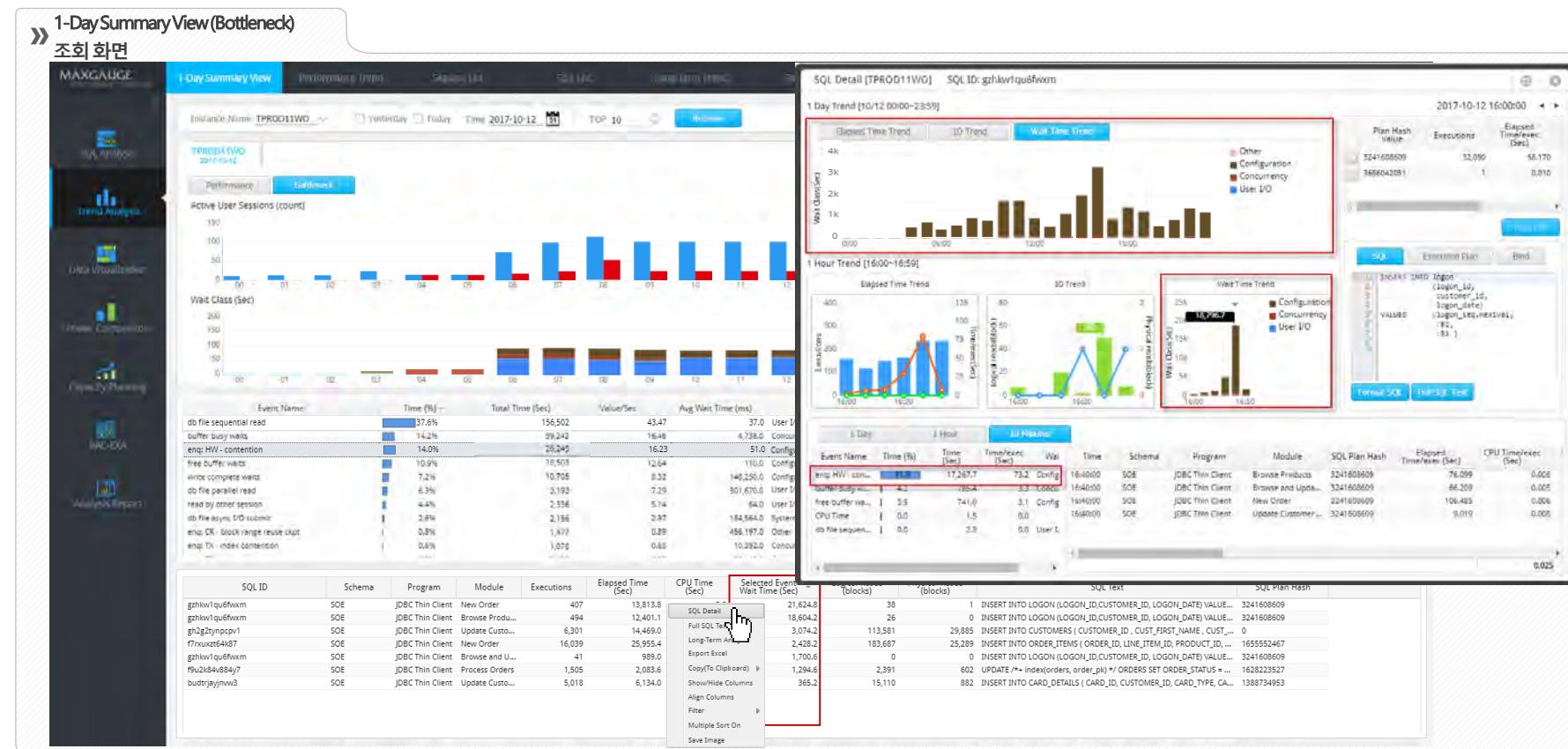
"Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)



- ✓ SQL Detail 기능 연계를 통해 "SQL ID : gzhkw1qu6fwxm" 의 상세 Wait Time 정보를 확인할 수 있습니다.
- ✓ Wait Time 추이를 통해 Configuration Class 비중이 높다는 것을 확인할 수 있으며, Peak 시간대에서도 16:40 가장 많이 대기한 것을 알 수 있습니다.
- ✓ Wait Time 정보 외에도 주요 성능 정보를 포함하여 SQL Text, Plan, Bind 등의 정보를 표시하므로 SQL 분석을 통하여 해당 SQL의 문제를 파악할 수 있습니다.

1-Day Summary View의 활용

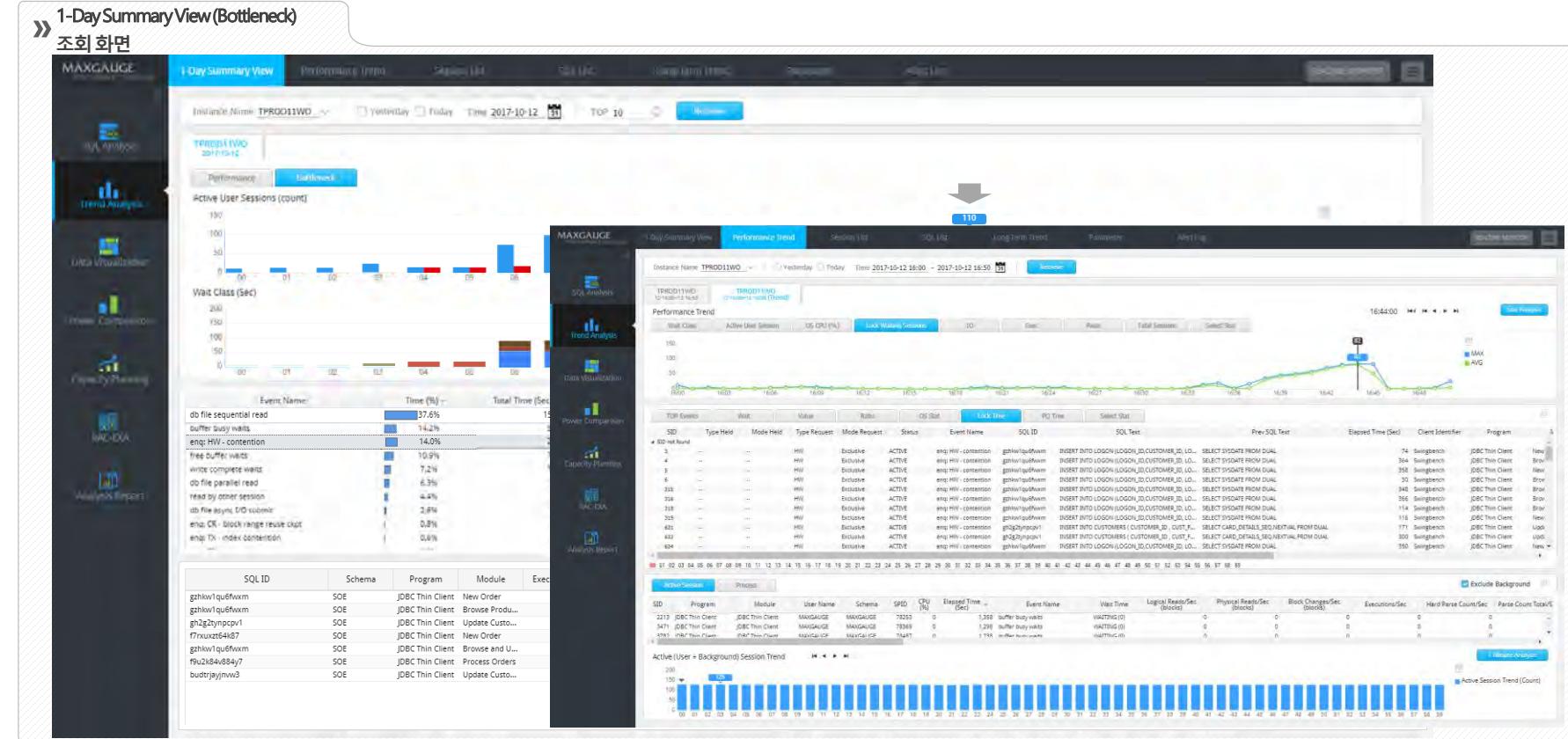
STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Wait Class 추이 및 비율 확인
- Top Event 확인
- 분석 대상 SQL 추출
(buffer busy waits)
- SQL 상세 정보 확인
(buffer busy waits)
- 분석 대상 SQL 추출
(enq: HW – contention)
- SQL 상세 정보 확인
(enq: HW – contention)

“Peak 시간 대에 발생한 대기 이벤트 및 원인을 확인하고 싶어요. (Bottleneck 분석)”



- ✓ Performance Trend에서 10/12 16:00~16:59 구간을 지정하여 상세 분석을 하면 Lock Waiting Sessions 수치가 올라가는 그래프를 확인할 수 있습니다.
- ✓ 그래프 하단의 Lock Tree 랩을 통해 해당 시간대에 enq: HW – contention 이벤트를 대기하는 세션 정보를 트리구조로 확인할 수 있습니다.
- ✓ Lock Tree에서 Holder Session 또는 Blocking Session 기능을 활용해서 락 발생의 원인을 찾을 수 있습니다.

MAXGAUGE



Practical Guide

Parameter View [Performance Analyzer]

Contents

Parameter ?

Parameter View 의 활용

Case 1. 배치업무가 느려진 이유를 알고 싶어요

Parameter View?

Parameter View의 활용방법은 무엇인가요?



파라미터 변경 이력을 통해
성능 장애를 분석할 때

Parameter View?

Parameter View

- 오라클에서 제공하는 모든 Parameter 정보를 제공하는 기능이며 Performance Analyzer ▶ Tend Analysis ▶ Parameter 경로를 통해 사용할 수 있다.
- Pre-Check 탭은 모니터링 DB에 설정된 현재 Parameter에 대한 카테고리 별 현황을 보여준다.
 - Basic Initialization / Diagnostics & Statistics / Optimizer / SGA & PGA
- Search Result 탭은 Parameter 명을 %Search Word%로 검색하여 결과값을 보여주고 공백입력 시 모든 Parameter 현황을 제공한다.
- Modification History 탭은 Parameter의 변경 이력을 제공한다.

» Pre-Check

Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description
▲ Basic Initialization							
	db_name	ORA12102	FALSE	FALSE	FALSE	FALSE	database name specified in CREATE DATABASE
	nls_language	KOREAN	FALSE	FALSE	TRUE	TRUE	NLS language name
	instance_name	ORA12102	TRUE	FALSE	FALSE	FALSE	instance name supported by the instance
	instance_number	0	TRUE	FALSE	FALSE	FALSE	instance number
	ldap_directory_sysauth	no	TRUE	FALSE	FALSE	FALSE	GID usage parameter
	cluster_database	FALSE	TRUE	FALSE	FALSE	FALSE	If TRUE startup in cluster database mode
	compatible	12.1.0.2.0	FALSE	FALSE	FALSE	FALSE	Database will be completely compatible with th...
	control_files	/ora_data/or...	FALSE	FALSE	FALSE	FALSE	control-file-names list
	cpu_count	24	TRUE	FALSE	IMMEDIATE	FALSE	number of CPUs for this instance
	shared_servers	1	TRUE	FALSE	IMMEDIATE	FALSE	number of shared servers to start up
	db_create_file_dest		TRUE	FALSE	IMMEDIATE	TRUE	default database location
	db_domain		FALSE	FALSE	FALSE	FALSE	directory part of global database name stored ...
	sessions	1536	TRUE	FALSE	IMMEDIATE	FALSE	user and system sessions
	nls_territory	KOREA	FALSE	FALSE	FALSE	TRUE	NLS territory name
	db_recovery_file_dest		FALSE	FALSE	IMMEDIATE	FALSE	default database recovery file location
	db_recovery_file_dest_size	1048576000	FALSE	FALSE	IMMEDIATE	FALSE	database recovery files size limit
	db_unique_name	ORA12102	TRUE	FALSE	FALSE	FALSE	Database Unique Name
	remote_login_passwordfile	EXCLUSIVE	FALSE	FALSE	FALSE	FALSE	password file usage parameter
	remote_listener		TRUE	FALSE	IMMEDIATE	FALSE	remote listener
	processes	1000	FALSE	FALSE	FALSE	FALSE	user processes
	open_cursors	300	FALSE	IMMEDIATE	FALSE	FALSE	max # cursors per session
▶ Diagnostics & Statistics							
▶ Optimizer							
▶ SGA & PGA							

» Search Result

Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description
Parameter							
	_aq_pt_processes	15	TRUE	FALSE	IMMEDIATE	FALSE	Partition background processes
	_ges_server_processes	1	TRUE	FALSE	FALSE	FALSE	number of background global enqueue server ...
	_high_priority_processes	LMS*	TRUE	FALSE	FALSE	FALSE	High Priority Process Name Mask
	_highest_priority_processes	VKTM	TRUE	FALSE	FALSE	FALSE	Highest Priority Process Name Mask
	_ksr_unit_test_processes	0	TRUE	FALSE	FALSE	FALSE	number of ks unit test processes
	_shrd_que_tm_processes	1	TRUE	FALSE	IMMEDIATE	FALSE	number of sharded queue Time Managers to st...
	_trace_processes	ALL	TRUE	FALSE	FALSE	FALSE	enable KST tracing in process
	aq_tm_processes	1	TRUE	FALSE	IMMEDIATE	FALSE	number of AQ Time Managers to start
	db_writer_processes	3	TRUE	FALSE	FALSE	FALSE	number of background database writer proces...
	gcs_server_processes	0	TRUE	FALSE	FALSE	FALSE	number of background gcs server processes to ...
	global_txn_processes	1	TRUE	FALSE	IMMEDIATE	FALSE	number of background global transaction proc...
	job_queue_processes	48	FALSE	FALSE	IMMEDIATE	FALSE	maximum number of job queue slave processes
	log_archive_max_processes	4	TRUE	FALSE	IMMEDIATE	FALSE	maximum number of active ARCH processes
	processes	1000	FALSE	FALSE	IMMEDIATE	FALSE	user processes

» Modification History

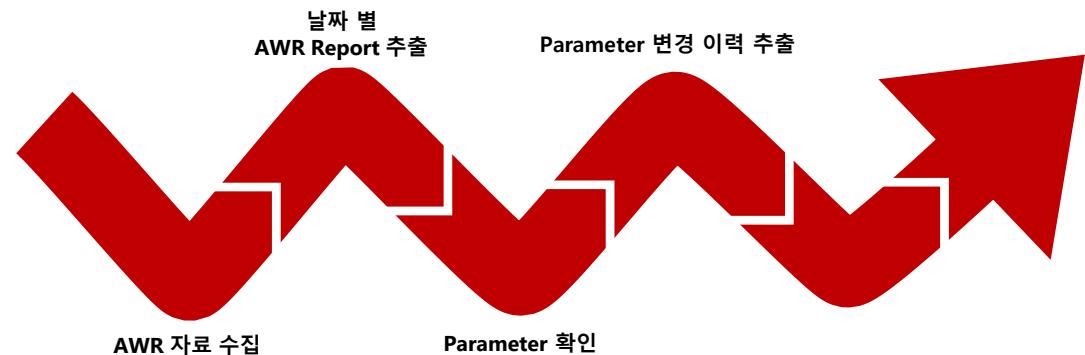
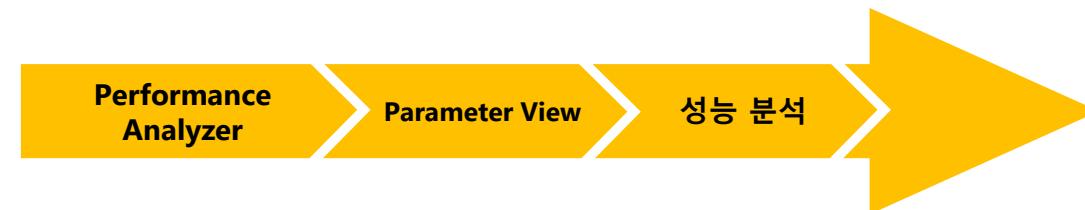
Name	Date	Current Value	Prev Value
__db_cache_size	2017-11-27	16441671680	16039016496
	2017-11-22	16039016496	134217728
	2017-10-29	134217728	16441671680
__java_pool_size	2017-11-27	67108864	402653184
	2017-11-22	402653184	67108864
__streams_pool_size	2017-11-27	67108864	134217728
	2017-10-29	134217728	67108864
__fastpin_enable	2017-11-27	TRUE	FALSE
	2017-10-29	FALSE	TRUE
__load_without_compile	2017-11-22	221945089	227499265
	2017-10-29	227499265	none
db_cache_size	2017-11-22	0	134217728
	2017-10-29	134217728	0
java_pool_size	2017-11-22	0	67108864
	2017-10-29	67108864	0
large_pool_size	2017-11-22	0	469762048
	2017-10-29	469762048	0

Parameter 성능 분석을 더욱 쉽게 할 수 있습니다!

성능
분석 방법

MaxGauge

Guide



성능개선

Parameter View 의 활용

Parameter View 의 사용 방법

Parameter View

- 검색 조건 설정
- 데이터 분석

» ParameterView 검색화면

MAXGAUGE

SQL Analysis

Trend Analysis

Data Visualization

Power Comparison

Capacity Planning

RAC-DMA

Analysis Report

Instance No: ① 11WPL

Yesterday ② Today 2017-11-29 ③ Parameter %Search Word%

Pre-Check Search Result Modification History

Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description

- ① 대상 Instance를 선택합니다.
- ② 일단위로 시간을 선택합니다.
- ③ Parameter 이름을 like 조건과 동일한 %Search Word% 를 통해 검색할 수 있습니다.

Parameter View의 활용

Parameter View 의 사용 방법

Parameter View

- 검색 조건 설정
- 데이터 분석

» ParameterView 조회화면

The screenshot shows the MAXGAUGE Parameter View interface. At the top, there are several tabs: 1-Day Summary View, Performance Trend, Session Log, SQL LK, Long Term Trend, Parameter (highlighted in blue), and Alert Log. Below the tabs, there are filters for Instance Name (PROD11WP), Date (Yesterday), Time (2017-11-29), and a search bar for 'Parameter %Search word%'. The main area is titled 'PROD11WP 20171129' and contains three tabs: Pre-Check (circled in red), Search (circled in green), and Modification History (circled in blue). The 'Pre-Check' tab displays a table of Oracle parameters categorized into 'Basic Initialization' and 'Diagnostic & Statistics'. The 'Search' tab and 'Modification History' tab are also visible.

Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description
Basic Initialization	db_name	ORA11204	FALSE	FALSE	FALSE	FALSE	database name specified in CREATE DATABASE
	nls_language	AMERICAN	TRUE	FALSE	FALSE	TRUE	NLS language name
	instance_name	ORA11204	TRUE	FALSE	FALSE	FALSE	instance name supported by the instance
	instance_number	0	TRUE	FALSE	FALSE	FALSE	Instance number
	ldap_directory_sysauth	no	TRUE	FALSE	FALSE	FALSE	OID usage parameter
	cluster_database	FALSE	TRUE	FALSE	FALSE	FALSE	if TRUE startup in cluster database mode
	compatible	11.2.0.4.0	FALSE	FALSE	FALSE	FALSE	Database will be completely compatible with the...
	control_files	/ora_data/or...	FALSE	FALSE	FALSE	FALSE	control file names list
	cpu_count	24	TRUE	FALSE	IMMEDIATE	FALSE	number of CPUs for this instance
	shared_servers	1	TRUE	FALSE	IMMEDIATE	FALSE	number of shared servers to start up
	db_create_file_dest		TRUE	FALSE	IMMEDIATE	TRUE	default database location
	db_domain		FALSE	FALSE	FALSE	FALSE	directory part of global database name stored ...
	sessions	7536	FALSE	FALSE	FALSE	FALSE	user and system sessions
	nls_territory	AMERICA	TRUE	FALSE	FALSE	TRUE	NLS territory name
	db_recovery_file_dest		FALSE	FALSE	IMMEDIATE	FALSE	default database recovery file location
db_recovery_file_dest_size	10737418240	FALSE	FALSE	IMMEDIATE	FALSE	database recovery files size limit	
db_unique_name	ORA11204	TRUE	FALSE	FALSE	FALSE	Database Unique Name	
remote_login_passwordfile	EXCLUSIVE	FALSE	FALSE	FALSE	FALSE	password file usage parameter	
remote_listener		TRUE	FALSE	IMMEDIATE	FALSE	remote listener	
processes	5000	FALSE	FALSE	FALSE	FALSE	user processes	
open_cursors	300	FALSE	FALSE	IMMEDIATE	FALSE	max # cursors per session	
Diagnostic & Statistics	timed_statistics	TRUE	TRUE	FALSE	IMMEDIATE	TRUE	maintain internal timing statistics
	timed_os_statistics	0	TRUE	FALSE	IMMEDIATE	TRUE	Internal os statistic gathering interval in seconds
	db_block_checking	FALSE	TRUE	FALSE	IMMEDIATE	FALSE	header checking and data and index block chec...

- ① 검색 결과는 Pre-Check, Search Result, Modification History 으로 분류됩니다.
- ② Pre-Check 탭은 Oracle Parameter 중 성능 관점의 주요 Parameter 를 4가지로 분류하여 제공합니다.
 - Basic Initialization
 - Diagnostic & Statistics
 - Optimizer
 - SGA & PGA

Parameter View 의 사용 방법

Parameter View

- 검색 조건 설정
- 데이터 분석

» ParameterView 조회 화면

The screenshot shows the MAXGAUGE Parameter View interface. At the top, there are tabs for '1-Day Summary View', 'Performance Trend', 'Session View', 'SQL Filter', 'Long Term Trend', 'Parameter' (which is selected), 'Monitoring', and 'REALTIME MONITOR'. Below the tabs, there's a search bar with the placeholder 'Parameter process' and a magnifying glass icon. A circled '1' is over the search bar. Below the search bar, there are three tabs: 'Pre-Check' (disabled), 'Search' (selected, circled with a '2'), and 'Modification History'. The main area displays a table of parameter settings. The columns are: Name, Value, Default, Modified, SVS Modified, Session Modified, and Description. The table includes rows for various Oracle parameters like '_high_priority_processes', '_jvm_unit_test_processes', '_trace_processes', etc. A circled '3' is over the table.

Name	Value	Default	Modified	SVS Modified	Session Modified	Description
_high_priority_processes	LMS* VKTM	TRUE	FALSE	FALSE	FALSE	High Priority Process Name Mask
_jvm_unit_test_processes	0	TRUE	FALSE	FALSE	FALSE	number of jvm unit test processes
_trace_processes	ALL	TRUE	FALSE	FALSE	FALSE	enable KST tracing in process
aq_tm_processes	1	TRUE	FALSE	IMMEDIATE	FALSE	number of AQ Time Managers to start
db_writer_processes	3	TRUE	FALSE	FALSE	FALSE	number of background database writer proces...
gcs_server_processes	0	TRUE	FALSE	FALSE	FALSE	number of background gcs server processes to ...
global_txn_processes	1	TRUE	FALSE	IMMEDIATE	FALSE	number of background global transaction proc...
job_queue_processes	1000	FALSE	FALSE	IMMEDIATE	FALSE	maximum number of job queue slave processes
log_archive_max_processes	4	TRUE	FALSE	IMMEDIATE	FALSE	maximum number of active ARCH processes
processes	5000	FALSE	FALSE	FALSE	FALSE	user processes

- ① Parameter 이름을 like 조건인 %Search Word%로 검색합니다. (공백 입력 시 모든 Parameter 출력)
- ② 검색 결과는 Pre-Check, Search Result, Modification History으로 분류됩니다.
- ③ Search Result 탭은 검색 조건에 따른 Parameter 정보를 제공합니다.

Parameter View 의 사용 방법

Parameter View

- 검색 조건 설정
- 데이터 분석

» ParameterView 조회화면

The screenshot shows the MAXGAUGE Parameter View interface. At the top, there are tabs for 1-Day Summary View, Performance Trend, Session List, SQL List, Long-Term Trend, Parameter (which is selected), and Alert Log. Below the tabs, there are dropdowns for Instance Name (PROD11WP) and Date (Yesterday, Today, 2017-11-29). A search bar contains 'Parameter processes'. A 'Modification History' tab is highlighted with a circled '1'. The main area displays a table of parameter modifications:

Name	Date	Current Value	Prev Value
__db_cache_size	2017-11-27	16441671680	16039010496
	2017-11-22	16039010496	134217728
	2017-10-29	134217728	16441671680
__java_pool_size	2017-11-27	67108864	402653104
	2017-11-22	402653104	67108864
__streams_pool_size	2017-11-27	67108864	134217728
	2017-10-29	134217728	67108864
fastpin_enable	2017-11-22	221945089	227499265
load_without_compile	2017-11-22	0	NONE
	2017-10-29	NONE	none
db_cache_size	2017-11-22	0	134217728
	2017-10-29	134217728	0
java_pool_size	2017-11-22	0	67108864
	2017-10-29	67108864	0
large_pool_size	2017-11-22	0	469762048
	2017-10-29	469762048	0
resource_manager_plan	2017-11-29	SCHEDULER(0x3200);DEFAULT...	SCHEDULER(0x3200);DEFAULT...

① 검색결과는 Pre-Check, Search Result, Modification History 으로 분류됩니다.

② Modification History 탭은 Parameter 변경이력을 제공합니다.

실전 분석 사례 Case 1.
**배치 업무가 느려진 이유를
알고 싶어요.**

“배치 업무가 느려진 이유를 알고 싶어요!”

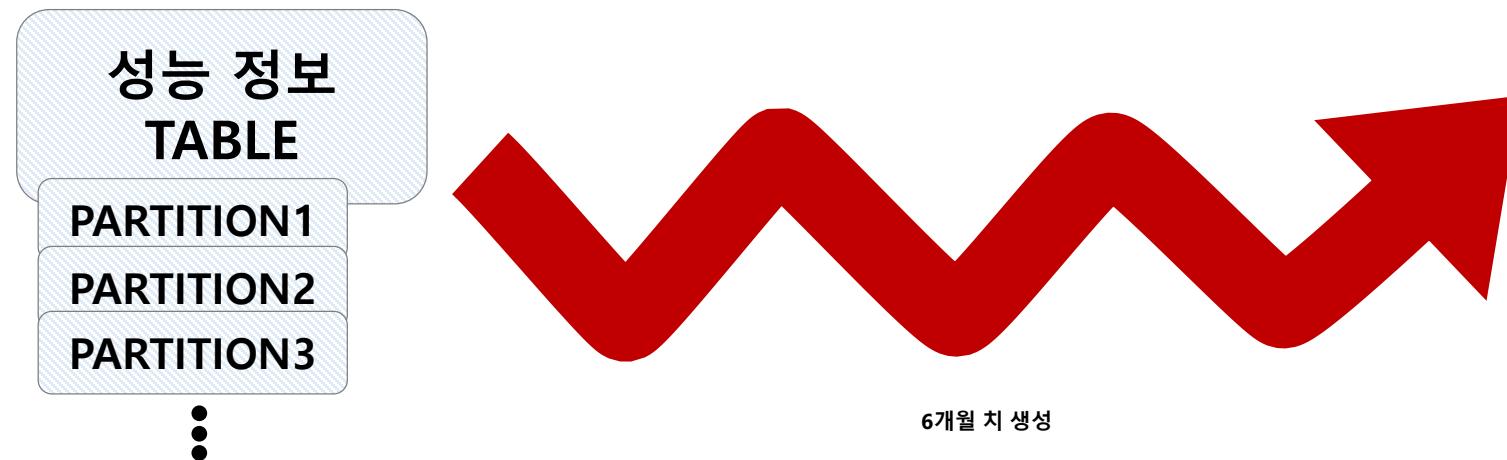
STEP

» MaxGauge 수집서버 운용 중 문제발생

1. 문제발생

2. 문제 분석

- Parameter 확인
 - Parameter 변경이력 확인
3. 문제 해결



- ✓ 하루에 한번 수행되는 MaxGauge 파티션 생성 작업을 정기 PM 작업 시 6개월 치 생성으로 대체하여 수행하는 배치 업무
- ✓ 6개월 치의 파티션을 미리 생성하는 배치 작업 수행 시 오랜 시간 대기하는 현상 발생
- ✓ 이를 분석하고자 MaxGauge 의 Performance Trend, Session Detail, Parameter View 를 활용

“배치 업무가 느려진 이유를 알고 싶어요!”

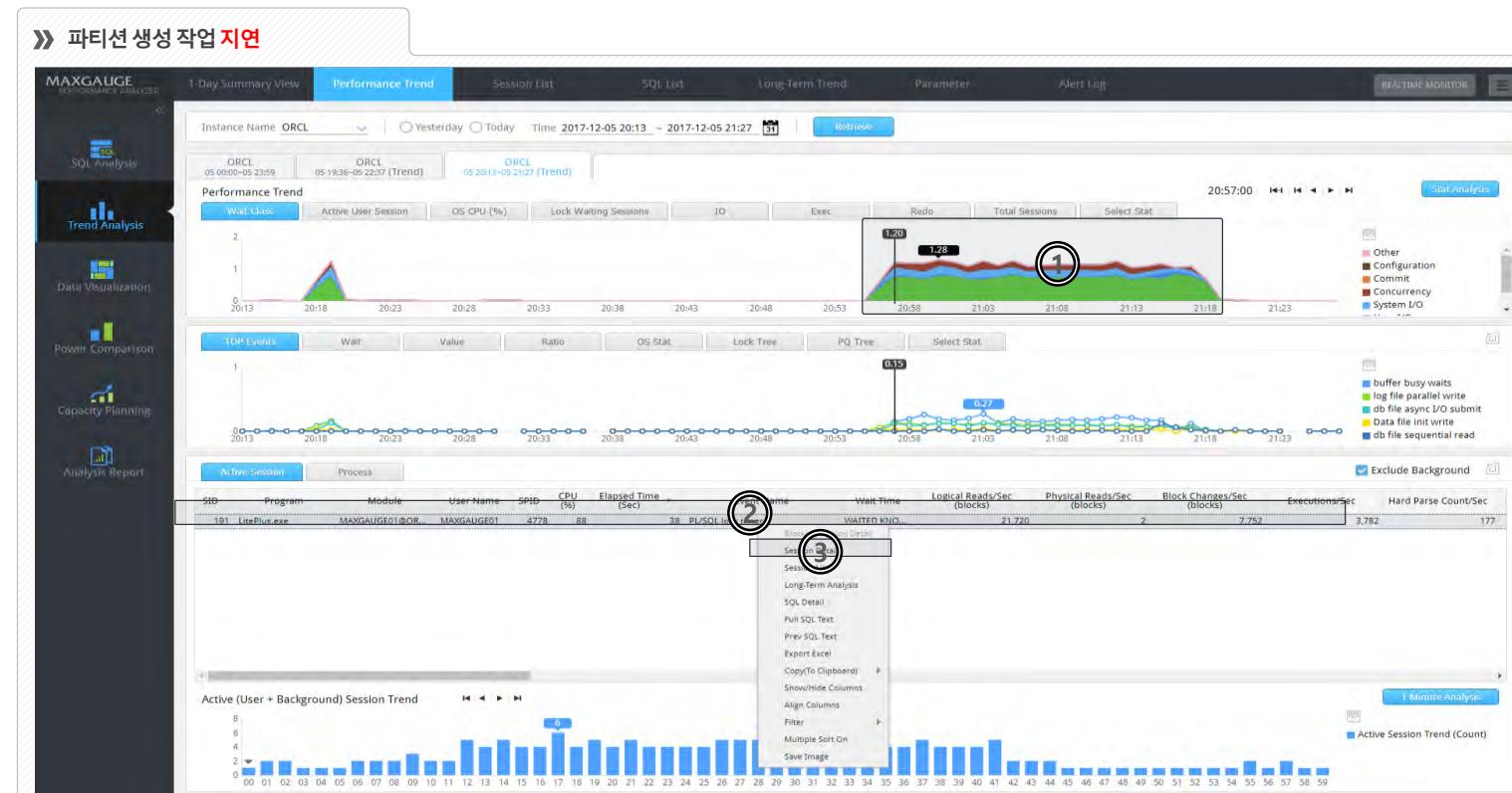
STEP

1. 문제발생

2. 문제 분석

- Parameter 확인
- Parameter 변경이력 확인

3. 문제 해결



- ① Trend Analysis -> Performance Trend에서 세션 20:56:00 ~ 21:02:00 시간대 7분 이상 파티션 생성 작업이 지연됨
- ② 20:57:00 시점에 대한 세션 리스트를 확인
- ③ 해당 세션에 대한 상세정보를 확인하기 위해 마우스 오른쪽 버튼 -> "Session Detail" 클릭

“배치 업무가 느려진 이유를 알고 싶어요!”

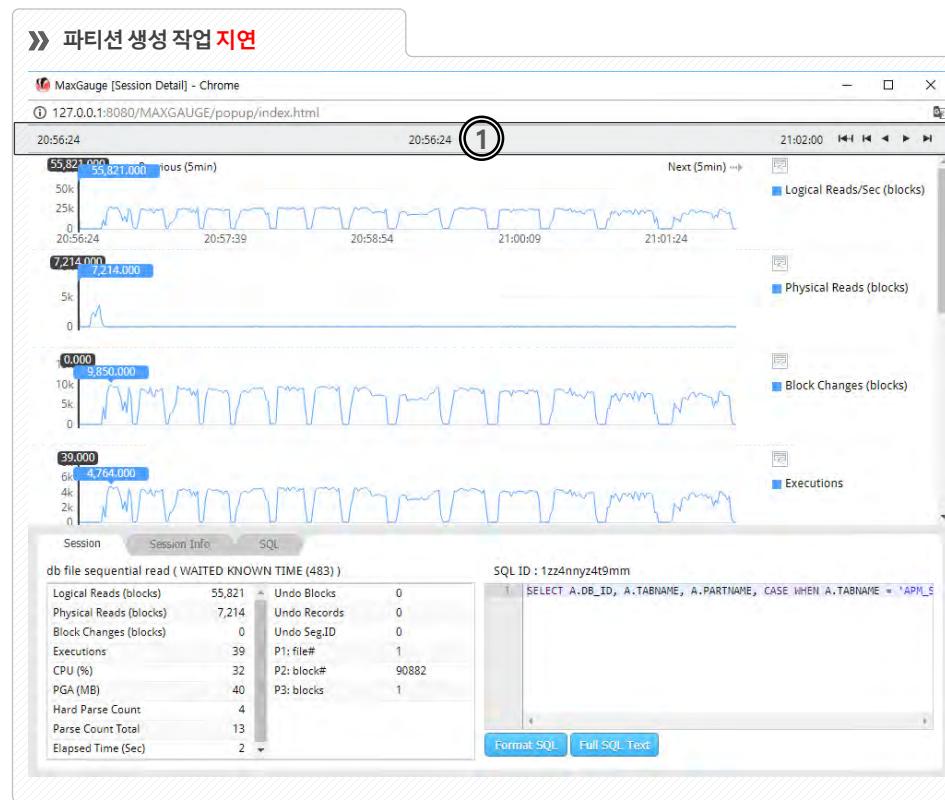
STEP

1. 문제발생

2. 문제 분석

- Parameter 확인
- Parameter 변경이력 확인

3. 문제 해결



① 20:56:24 ~ 21:02:00 약 7분 이상 수행하며 배치작업 지연 발생

STEP

1. 문제발생

2. 문제 분석

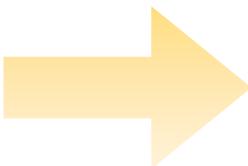
- Parameter 확인
- Parameter 변경이력 확인

3. 문제 해결

배치 작업 외 다른 업무 수행 안 함

파티션 생성은 오라클 내부 동작

사내 테스트 환경 발생 안 함



파라미터
전수 조사

“배치 업무가 느려진 이유를 알고 싶어요!”

STEP

1. 문제 발생

2. 문제 분석

- Parameter 확인
 - Parameter 변경이력 확인
3. 문제 해결

» ParameterView 검색 화면

Instance Name: PROD11WP Yesterday Today Time: 01:11:21 Parameter: deferred_segment_creation Review

Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description

① 일별 Parameter 값을 확인하기 위한 검색조건 추가 “`deferred_segment_creation`”

✓ 시나리오

INSTANCE : ORCL
PARAMETER : `deferred_segment_creation`
일 단위 : 2017년 11월 30일

✓ 목표

Parameter 현재 값 확인 후 변경이력 파악

“배치 업무가 느려진 이유를 알고 싶어요!”

STEP

1. 문제 발생

2. 문제 분석

▪ Parameter 확인

▪ Parameter 변경이력 확인

3. 문제 해결

» ParameterView 조회 화면

The screenshot shows the MAXGAUGE Performance Analyzer interface with the ParameterView module selected. The main window displays a table of parameters for the instance PROD11WP from November 29, 2017. The 'Search Result' tab is highlighted with a red circle and labeled with circled number ①. The table has columns: Name, Value, Default, Modified, SYS Modified, Session Modified, and Description. One row is shown for the parameter 'deferred_segment_creation' with the value 'TRUE'.

Name	Value	Default	Modified	SYS Modified	Session Modified	Description
deferred_segment_creation	TRUE	TRUE	FALSE	IMMEDIATE	TRUE	defer segment creation to first insert

① 검색 조건에 부합하는 결과값을 보기 위해 Search Result 선택

② “deferred_segment_creation” Parameter 현재 값을 확인

“배치 업무가 느려진 이유를 알고 싶어요!”

STEP

1. 문제 발생

2. 문제 분석

- Parameter 확인
- Parameter 변경이력 확인

3. 문제 해결

» ParameterView 조회 화면

The screenshot shows the MAXGAUGE ParameterView interface. The main title is "ParameterView 조회 화면". The top navigation bar includes tabs for "1-Day Summary View", "Performance Trend", "Session IBA", "SQL View", "Long-term Trend", "Parameter", and "Alert Log". The "Parameter" tab is selected. Below the tabs, there's a search bar with "Instance Name: ORCL", date range "Yesterday ~ Today", time "2017-12-05", and a parameter filter "Parameter: deferred_segment_creation". A "Refresh" button is also present. The main content area is titled "Historical Value" and displays a table of parameter changes over time. The table has columns: Name, Date, Current Value, and Prev Value. One row for "deferred_segment_creation" is highlighted, showing it was set to TRUE on 2017-12-04 and then changed to FALSE on 2017-12-05. Other parameters listed include _cursor_db_buffers_pinned, _db_block_buffers, _db_file_exec_read_count, _db_flash_cache_keep_limit, _db_writer_coalesce_area_size, _fastpin_enable, _load_without_compile, _small_table_threshold, _db_file_multiblock_read_size, and resource_manager_plan.

Name	Date	Current Value	Prev Value
_cursor_db_buffers_pinned	2017-12-04	155	155
_db_block_buffers	2017-12-04	23877	23560
_db_file_exec_read_count	2017-12-04	92	79
_db_flash_cache_keep_limit	2017-12-04	202747728	240000000
_db_writer_coalesce_area_size	2017-12-04	1966000	1965000
_fastpin_enable	2017-12-04	20282237	18889105
_load_without_compile	2017-12-04	none	none
_small_table_threshold	2017-12-04	671	671
_db_file_multiblock_read_size	2017-12-04	92	92
deferred_segment_creation	2017-12-04	TRUE	FALSE
resource_manager_plan	2017-12-04	SCHEDULER(0.10000000000000002)	SCHEDULER(0.10000000000000002)
shared_pool_reserved_size	2017-12-04	16198970	15400000

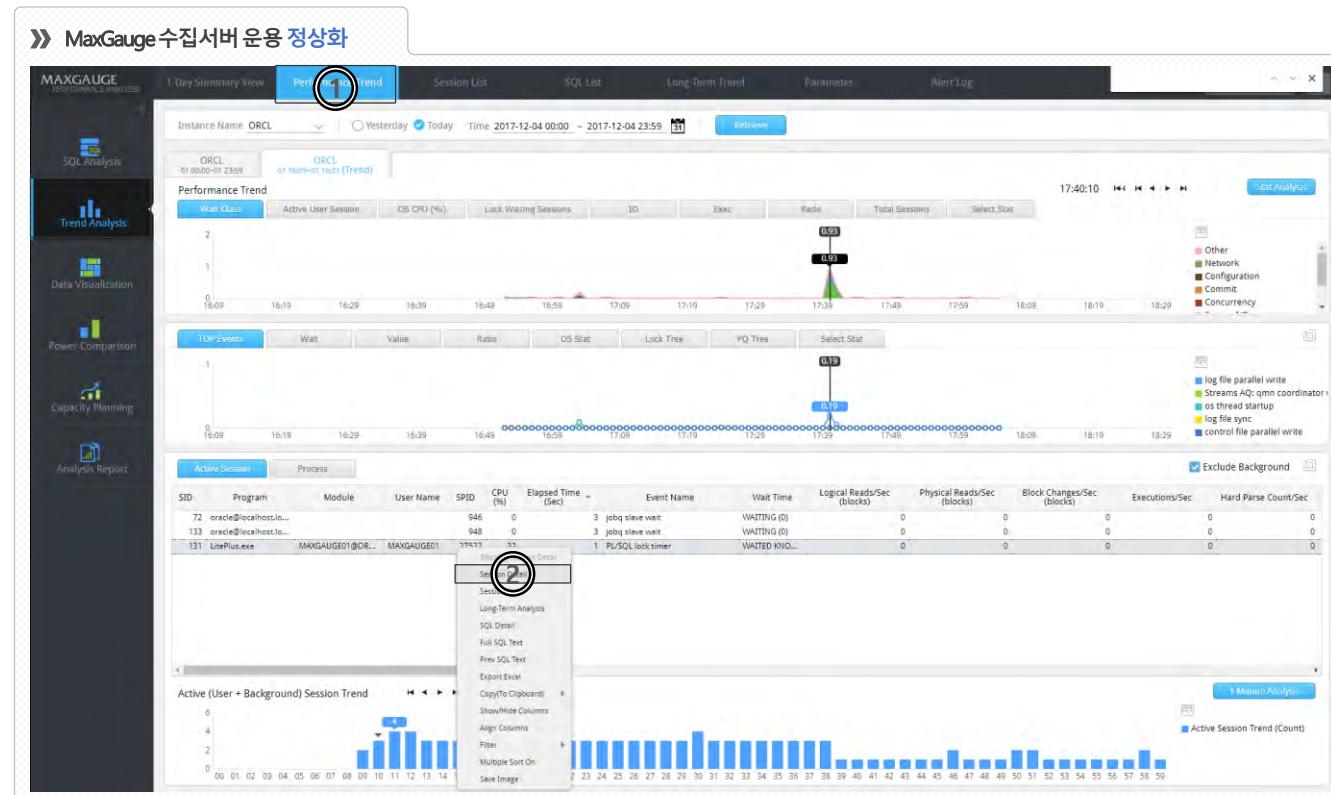
- ✓ deferred_segment_creation 파라미터가 **TRUE -> FALSE** 로 변경된 이력을 확인
- ✓ 즉, 테이블(또는 파티션) 생성 시 데이터가 존재하지 않아도 SEGMENT 를 바로 생성

Parameter View 의 활용

"배치 업무가 느려진 이유를 알고 싶어요!"

STEP

1. 문제 발생
2. 문제 분석
 - Parameter 확인
 - Parameter 변경이력 확인
3. 문제 해결

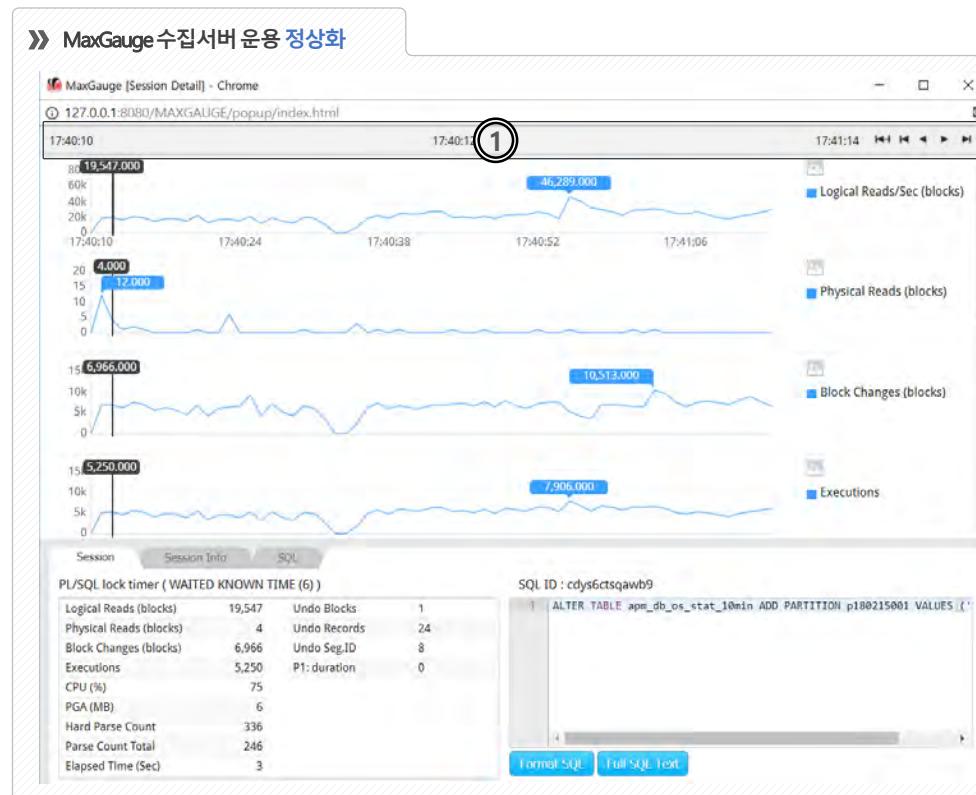


- ① Trend Analysis -> Performance Trend에서 세션의 추이를 확인
- ② 해당 세션에 대한 상세정보를 확인하기 위해 "Session Detail" 클릭

“배치 업무가 느려진 이유를 알고 싶어요!”

STEP

1. 문제 발생
2. 문제 분석
 - Parameter 확인
 - Parameter 변경이력 확인
3. 문제 해결



①

17:40:10 ~ 17:40:14 약 1분 수행하며 빠르게 완료됨

- 데이터가 들어와야만 SEGMENT가 만들어 지도록 파라미터 deferred_segment_creation=TRUE 로 변경
- 즉, **MaxGauge Parameter Modification History** 를 통해 해당 이슈를 해결

MAXGAUGE



Practical Guide

Data Visualization [Performance Analyzer]

Contents

Data Visualization?

DataPath View 의 활용

Case 1. 어떤 노드가 업무 비중도가 높은가요?

Case 2. 자꾸 GC 이벤트가 발생해요.

업무 분산이 잘 돼 있는지 확인할 수 있는 방법이 있을까요?

Case 3. 특정 SQL이 각 노드에서 수행되는 비중을 쉽게 알 수 있을까요?

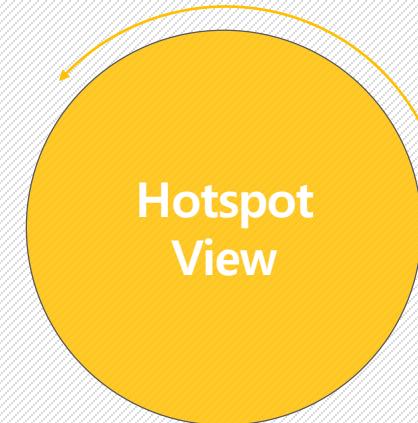
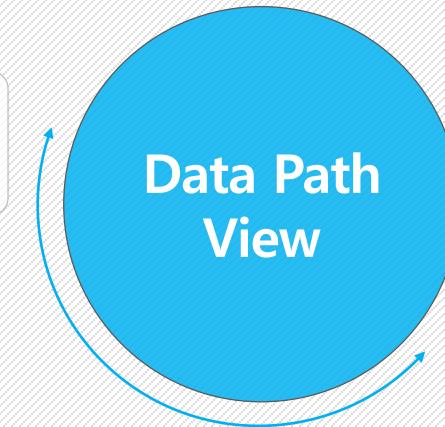
Part 3. Hotspot View 의 활용

Case 1. 출장 기간 동안 리소스에 부하가 언제 발생 했는지
확인하고 싶어요.

Data Visualization?

Data Visualization은 언제 쓰나요?

Instance별, 여러 일자의 업무 비중도 나,
Instance, Schema, Program, Module, SQL간의
연계상태가 궁금할 때



» 다수의 Instance 또는 여러 일자에 대한
CPU(%) hotspot 구간이 궁금할 때

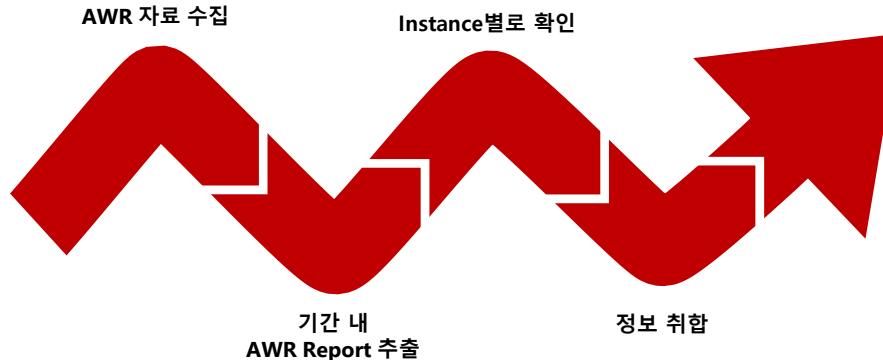
Instance별 업무 비중도를 더욱 쉽게 구분 할 수 있습니다!

MaxGauge

Performance Analyzer → Data Visualization → Instance별 업무 비중도

Instance별
업무 비중도
확인 요청

AWR



Instance별
업무 비중도
확인

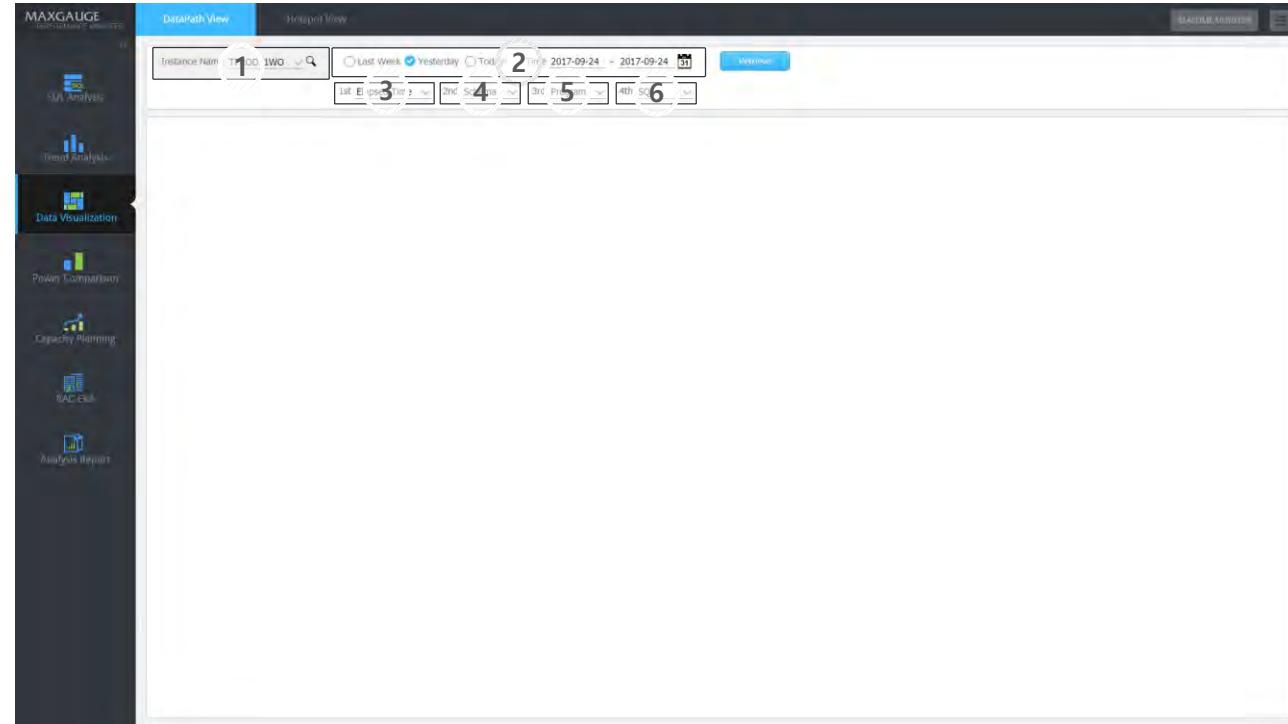
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 화면



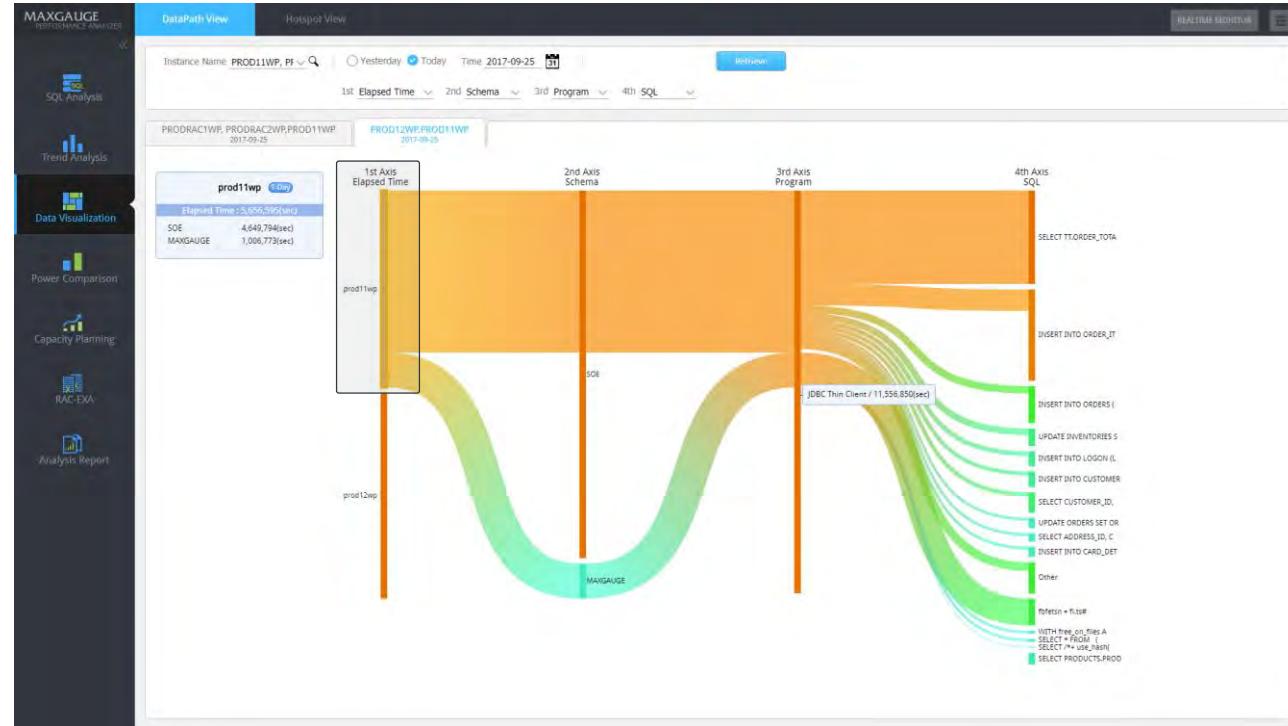
- 1 비교 대상 Instance를 선택합니다.(여러 개의 노드를 선택할 수 있습니다)
- 2 비교 하고자 하는 날짜를 선택 합니다. (다중 인스턴스의 경우 기간 검색이 불가능합니다.)
- 3 1차 축에 해당하는 Elapsed Time, CPU time, Logical Reads, Physical Reads 중에서 선택 할 수 있습니다.
- 4 2차 축에 해당하는 Schema, Program, Module 중에서 선택 할 수 있습니다.
- 5 3차 축에 해당하는 항목을 선택 할 수 있습니다. (2차 축에서 Schema를 선택한 경우 Program, Module. 2차 축에서 Program, Module을 선택한 경우 SQL)
- 6 4차 축에 해당하는 SQL을 선택 할 수 있습니다.(2차 축에서 Schema를 선택한 경우에만 4차 까지 구성 됩니다.)

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 결과화면 (1차 축 내의 특정 instance 선택 시의 화면)



- ✓ 1차 축인 Elapsed Time에서 원하는 instance 를 선택합니다.
- ✓ 선택 된 Instance 를 기준으로 Instance -> Schema -> Program -> SQL로의 Top-Down 방식의 분석이 가능합니다.

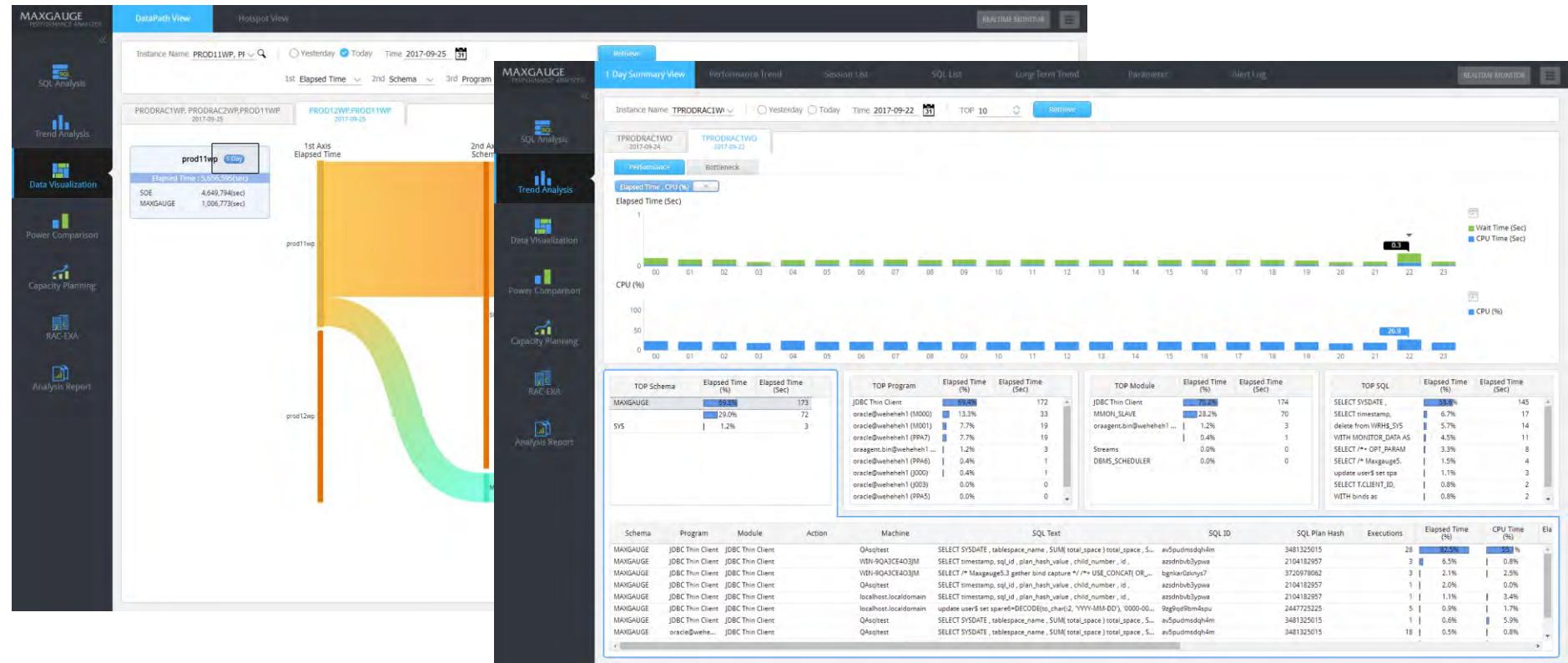
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 결과화면 (1차 축 내의 특정 instance 선택 시의 화면)



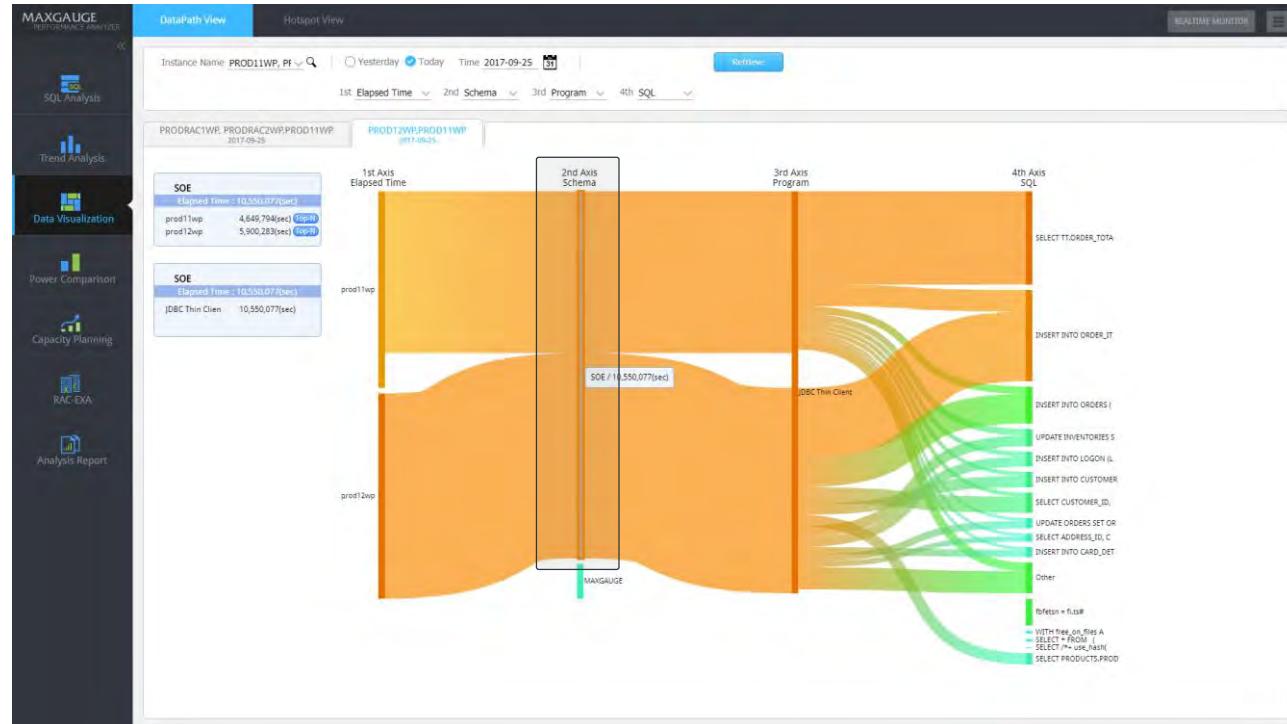
✓ 좌측 상단의 1-Day 버튼을 클릭하면 해당 Instance에 대한 1-Day Summary View로 연계 됩니다.

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 결과화면 (2차 축 내의 특정 Schema선택 시의 화면)



- ✓ 2차 축인 Schema에서 원하는 Schema를 선택합니다.
- ✓ 선택된 Schema를 기준으로 Instance <- Schema -> Program -> SQL로의 양방향의 분석이 가능합니다.

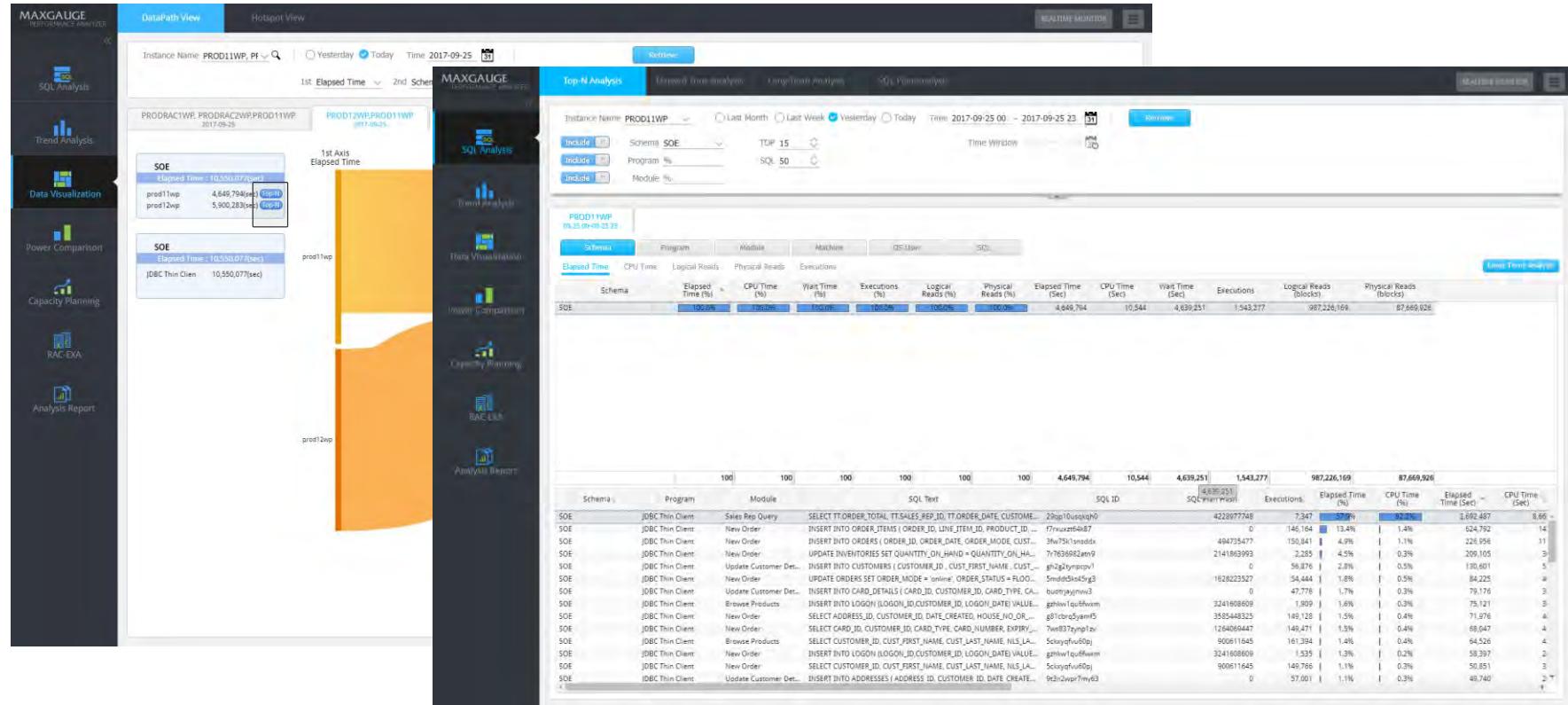
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
 - 데이터 분석
 - 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
 - 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
 - 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPath view 검색 결과 화면 (2차 축 내의 특정 Schema선택 시의 화면)



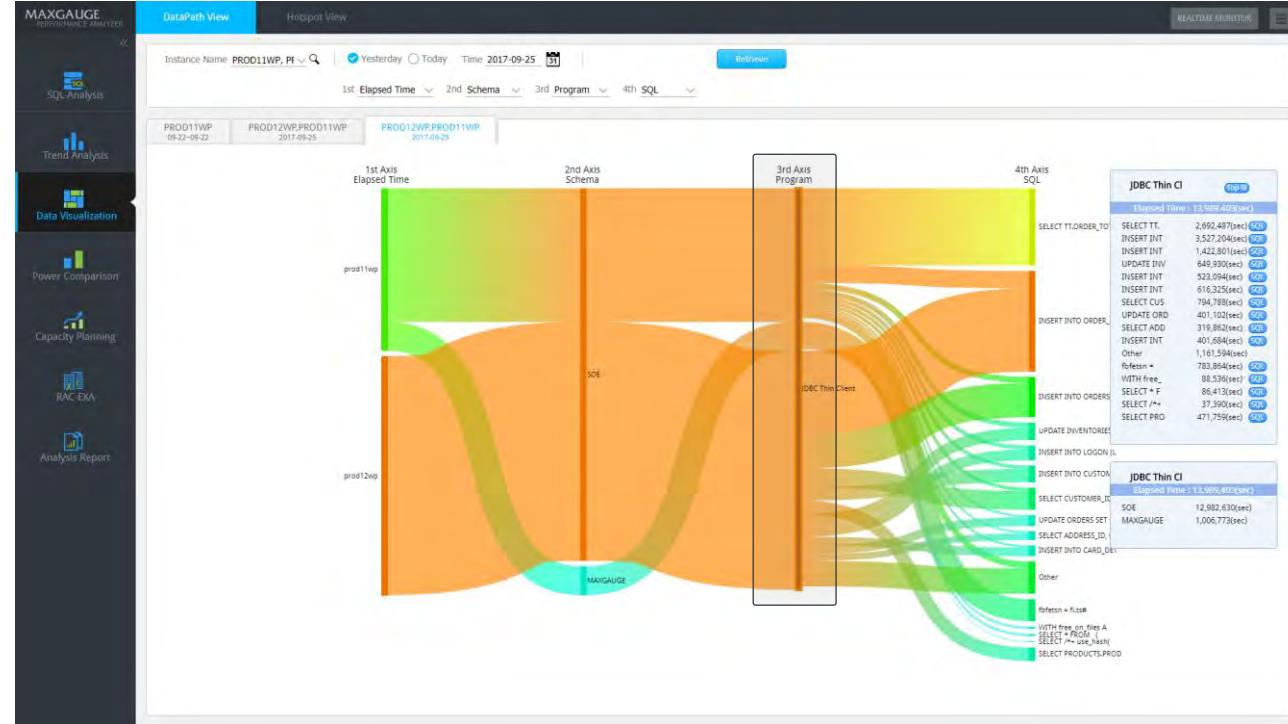
- ✓ 좌측상단 Top-N 버튼을 클릭하면 해당 Schema에 대한 Top-N Analysis로 연계 됩니다.

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 결과화면 (2차 축 내의 특정 Schema선택 시의 화면)



- ✓ 3차 축인 Program에서 원하는 Program을 선택합니다.
- ✓ 선택 된 Schema 를 기준으로 Instance <- Schema <- Program -> SQL로의 양방향의 분석이 가능합니다.

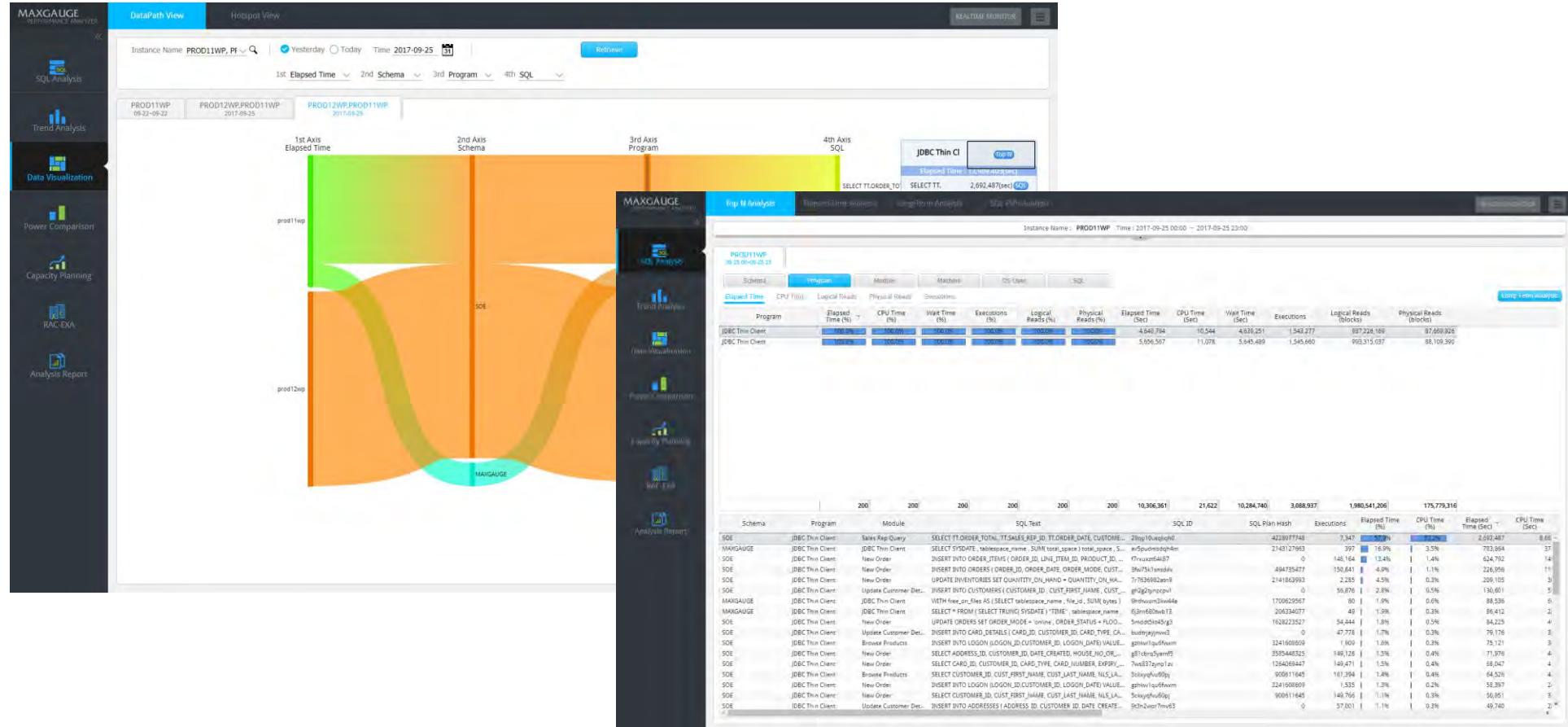
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPathView 검색 결과화면 (2차 축 내의 특정 Schema선택 시의 화면)



✓ 우측상단 Top-N 버튼을 클릭하면 해당 Schema에 대한 Top-N Analysis로 연계 됩니다.

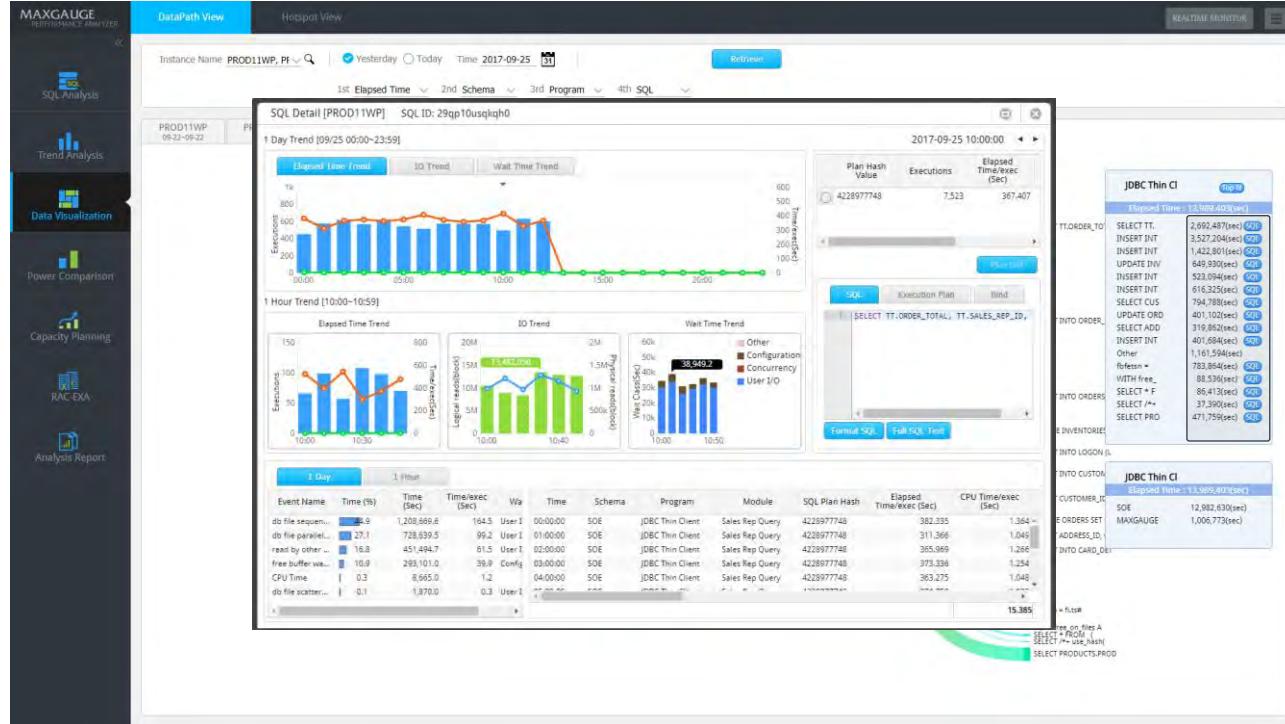
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계

» DataPath view 검색 결과화면 (3차 축 내의 특정 Program선택 시의 화면)



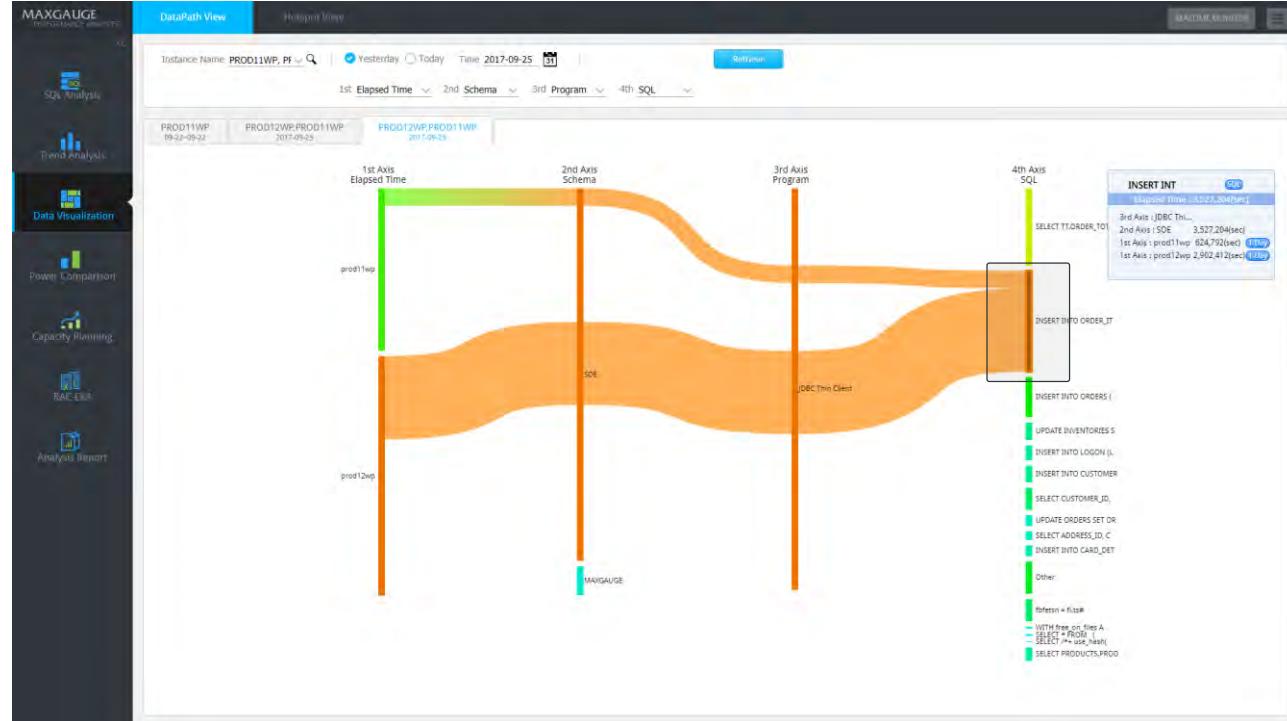
✓ 우측 SQL 버튼을 클릭하면 해당 SQL에 대한 SQL Detail로 연계 됩니다.

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계
 - └ 1-Day Summary View 연계

» DataPathView 검색 결과화면 (4차 축 내의 특정 SQL선택 시의 화면)



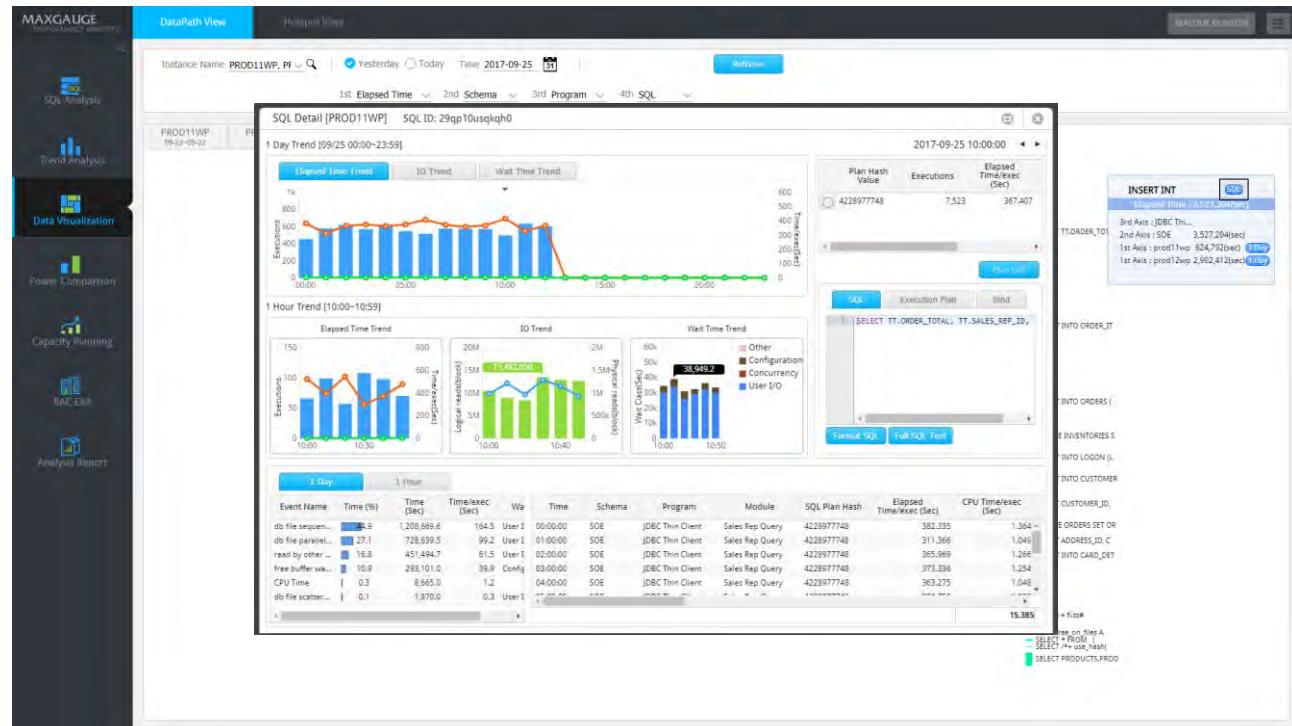
- ✓ 4차 축인 SQL에서 원하는 SQL을 선택합니다.
- ✓ 선택 된 Schema 를 기준으로 Instance <- Schema <- Program <- SQL로의 bottom-Up 분석이 가능합니다.

DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
 - 데이터 분석
 - 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
 - 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
 - 4차 축 상세 정보 확인
 - └ SQL Detail 연계
 - └ 1-Day Summary View 연계



- ✓ 우측 SQL 버튼을 클릭하면 해당 SQL에 대한 SQL Detail로 연계 됩니다.

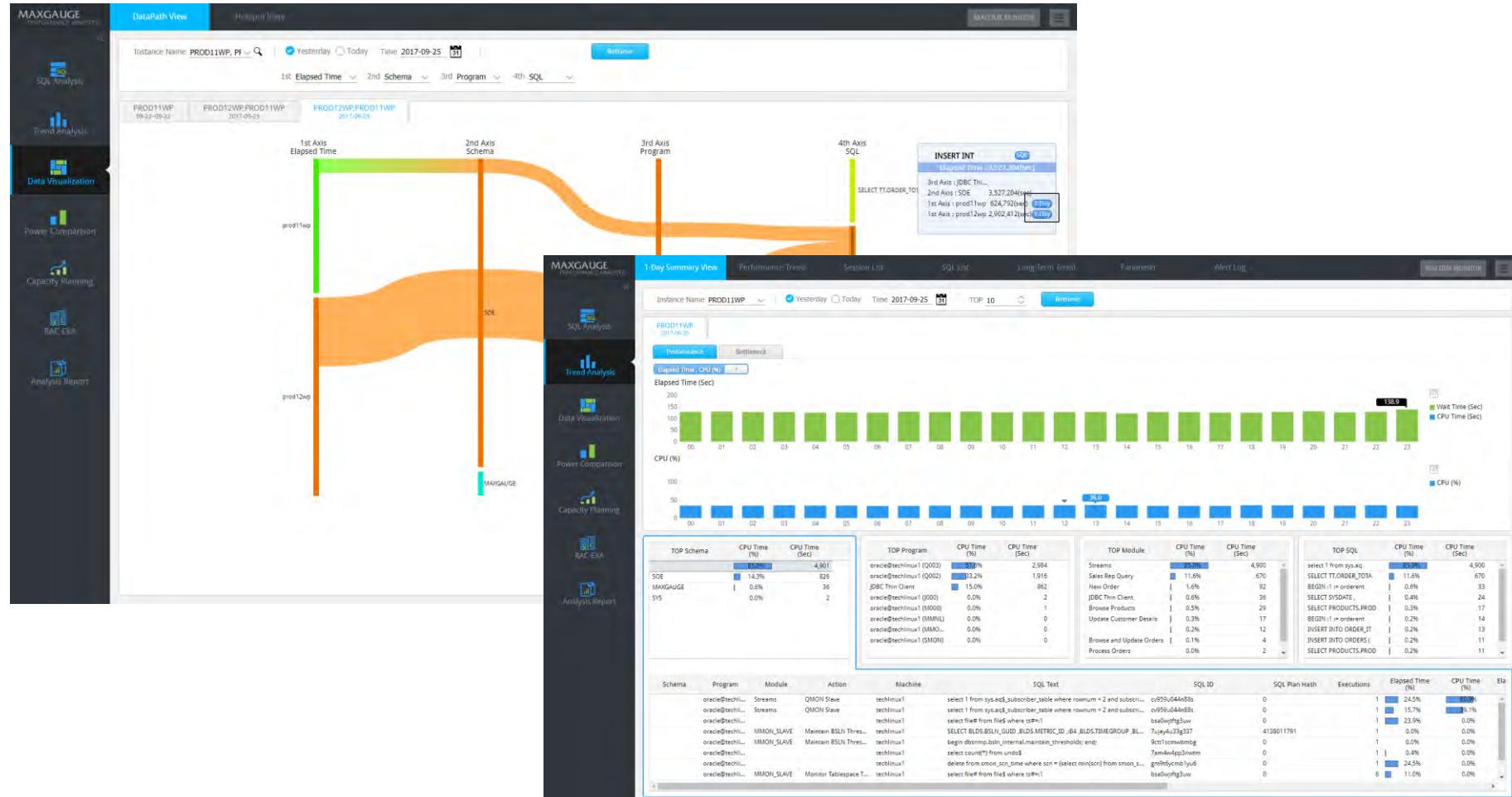
DataPath View의 활용

DataPath View의 사용 방법

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계
- 2차 축 상세 정보 확인
 - └ Top-N Analysis 연계
- 3차 축 상세 정보 확인
 - └ Top-N Analysis 연계
 - └ SQL Detail 연계
- 4차 축 상세 정보 확인
 - └ SQL Detail 연계
 - └ 1-Day Summary View 연계

» DataPathView 검색 결과화면 (4차 축 내의 특정 SQL선택 시의 화면)



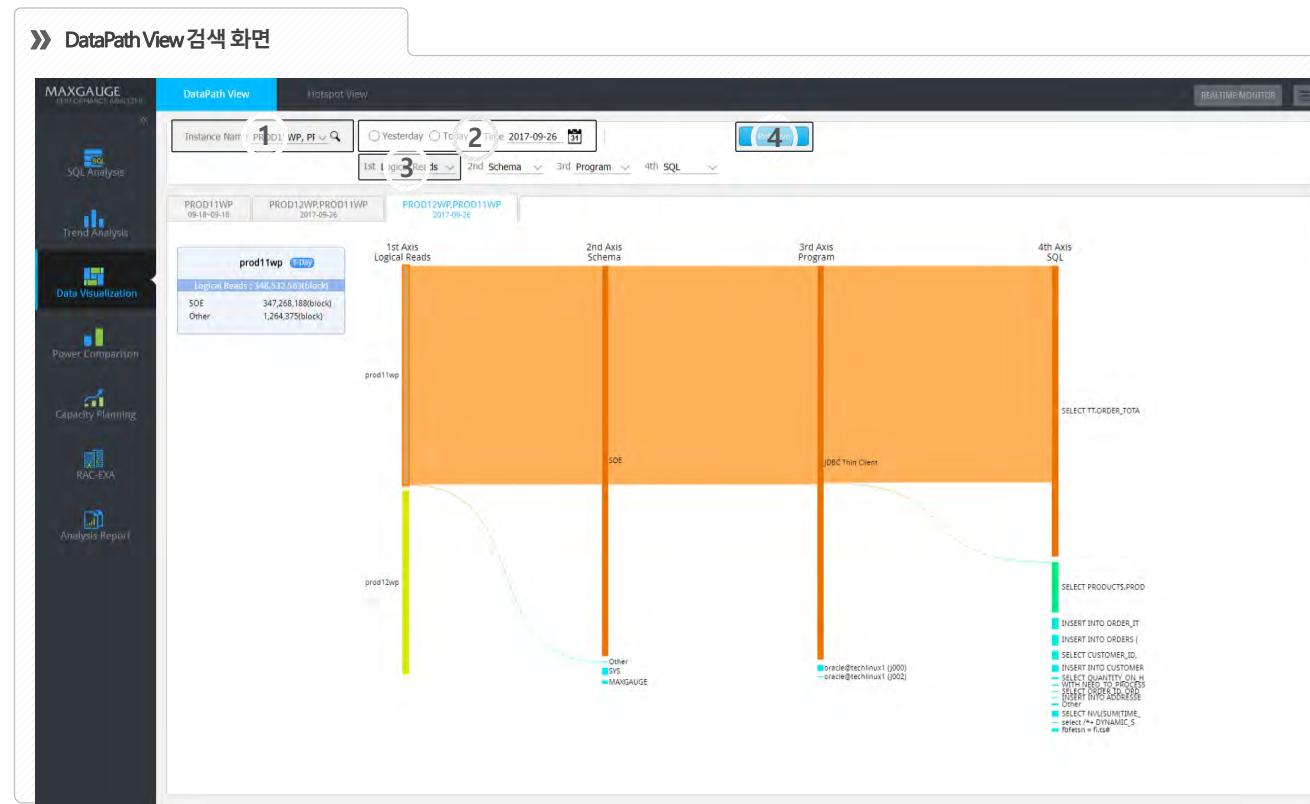
✓ 우측 1-Day 버튼을 클릭하면 해당 Instance에 대한 1-Day Summary View로 연계 됩니다.

실전 분석 사례 Case 1.
**어떤 노드가 업무비중도가
높은가요?**

“어떤 노드가 업무비중도가 높은가요?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - ↳ 1-Day Summary View 연계



- 1 비교 대상 Instance (PROD11WP, PROD12WP)를 선택합니다.
- 2 분석 날짜 (2017.09.26)는 하루만 선택 가능합니다.
- 3 업무 비중 도를 비교하기 위해 1차 축은 'Logical Reads'를 선택 합니다.
- 4 위의 설정한 조건으로 DataPath View를 실행합니다.

✓ 시나리오

INSTANCE : PROD11WP, PROD12WP
비교대상 일자 : 2017년 9월 26일

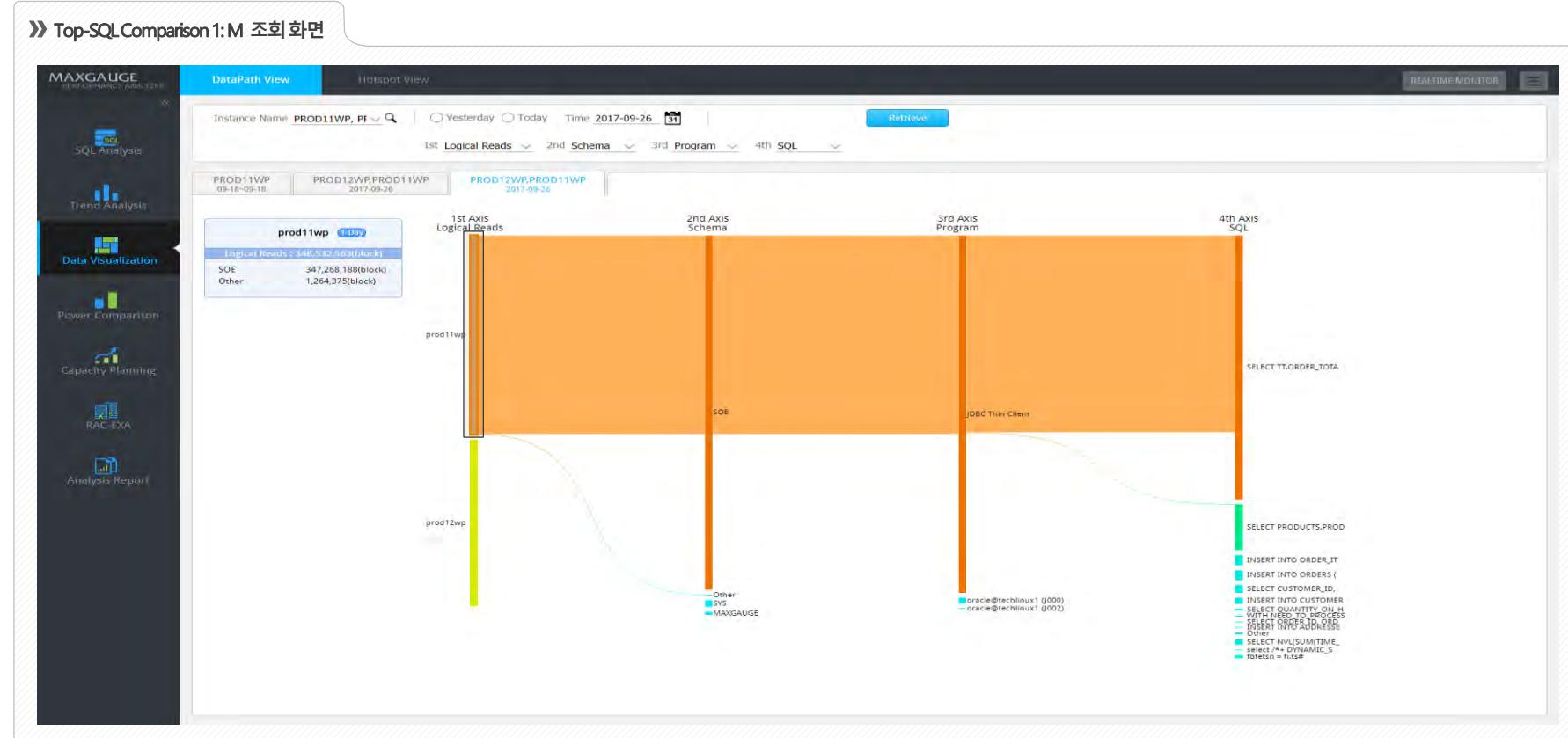
✓ 목표

업무 비중도가 높은 노드를 손쉽게 확인 하기

“어떤 노드가 업무비중도가 높은가요?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - └ 1-Day Summary View 연계

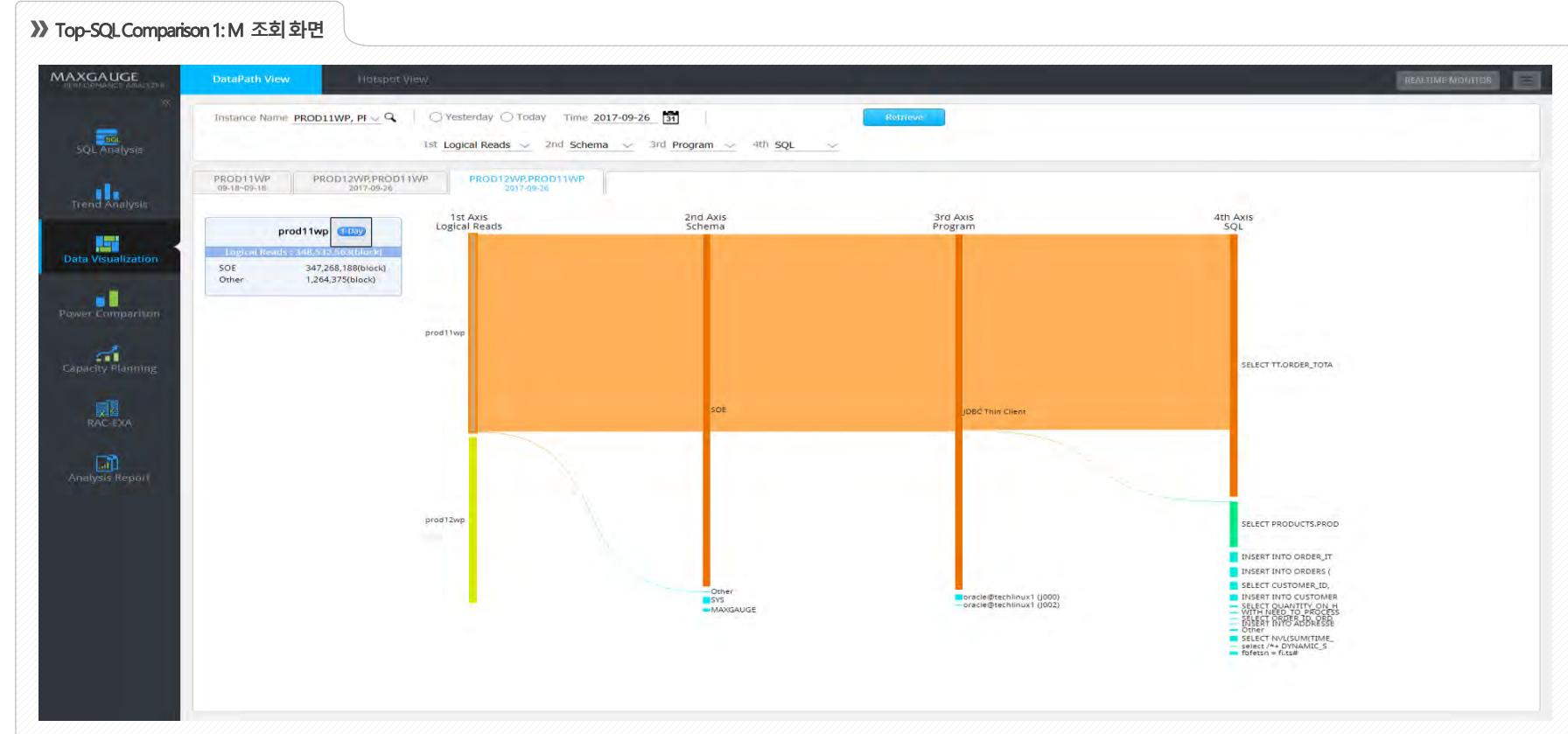


- ✓ 1차 축은 Logical Reads 를 길이로 표현 한 것입니다.
- ✓ 이 축이 길다는 것은 그 만큼 업무 비중도가 높다는 것을 뜻합니다.
- ✓ 위의 그림에선 2개의 노드가 비슷하게 Load Balancing 이 된 것으로 보입니다.

“어떤 노드가 업무비중도가 높은가요?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 1차 축 상세 정보 확인
 - ↳ 1-Day Summary View 연계



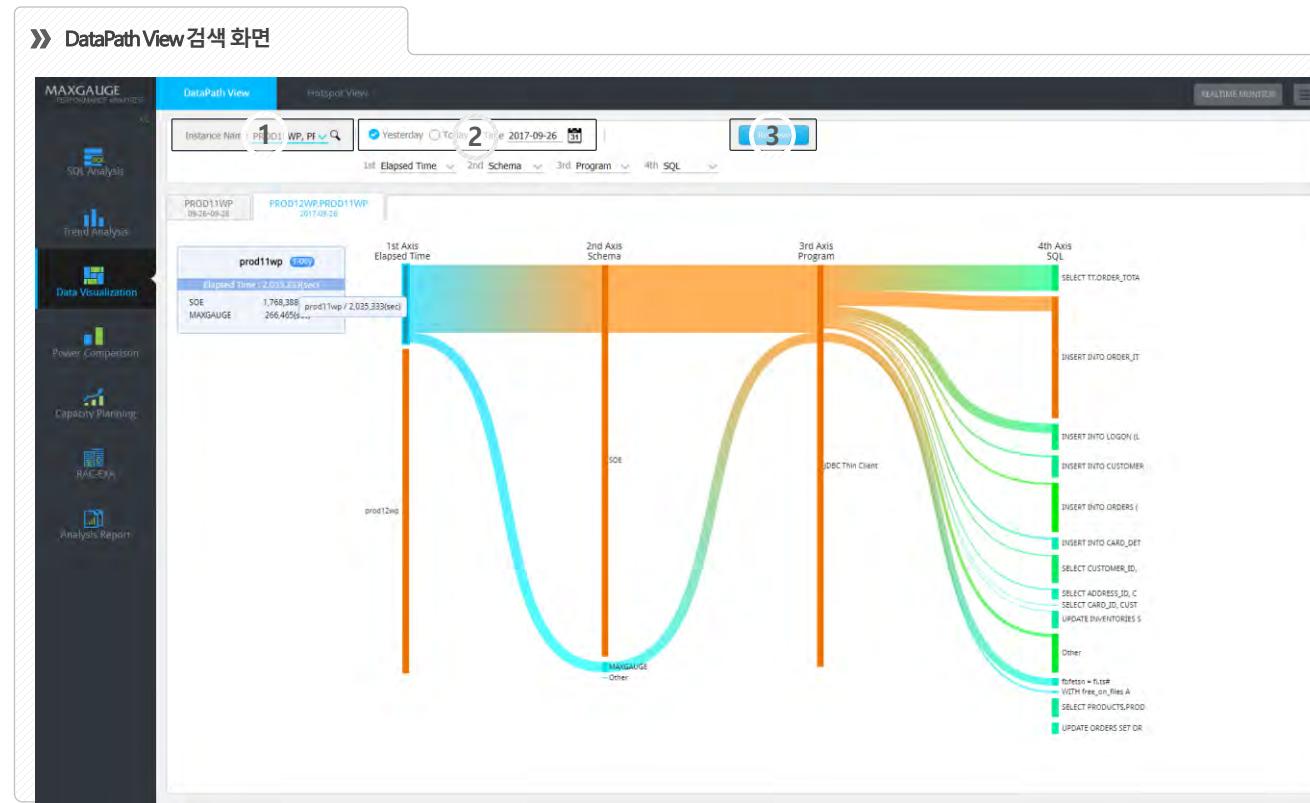
- ✓ 1차 축으로 부터 우측을 보면 순서대로 Schema, Program, SQL등을 확인 할 수 있습니다.
- ✓ 이 날짜의 상세한 내용을 확인 하려면 우측 상단의 1-Day를 확인 합니다.

실전 분석 사례 Case 2, 3.
자꾸 GC 이벤트가 발생해요. 업무 분산이
잘 돼 있는지 확인할 수 있는 방법이 있을까요? /
특정 SQL이 각 노드에서 수행되는 비중을 쉽게 알
수 있을까요?

“업무 분산은 잘 돼 있나요? / 노드 별 SQL 수행 비중은 어떻게 되죠?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 4차 축 상세 정보 확인
 - ↳ SQL Detail 연계



- 1 비교 대상 Instance (PROD11WP, PROD12WP)를 선택합니다.
- 2 분석 날짜 (2017.09.26)는 하루만 선택 가능합니다.
- 3 위의 설정한 조건으로 DataPath View를 실행합니다.

✓ 시나리오

INSTANCE : PROD11WP, PROD12WP
 비교대상 일자 : 2017년 9월 26일
 현재 상황 : gc 이벤트가 빈번하게 발생.
 노드 별 SQL 수행 비중을 알고 싶음.

✓ 목표

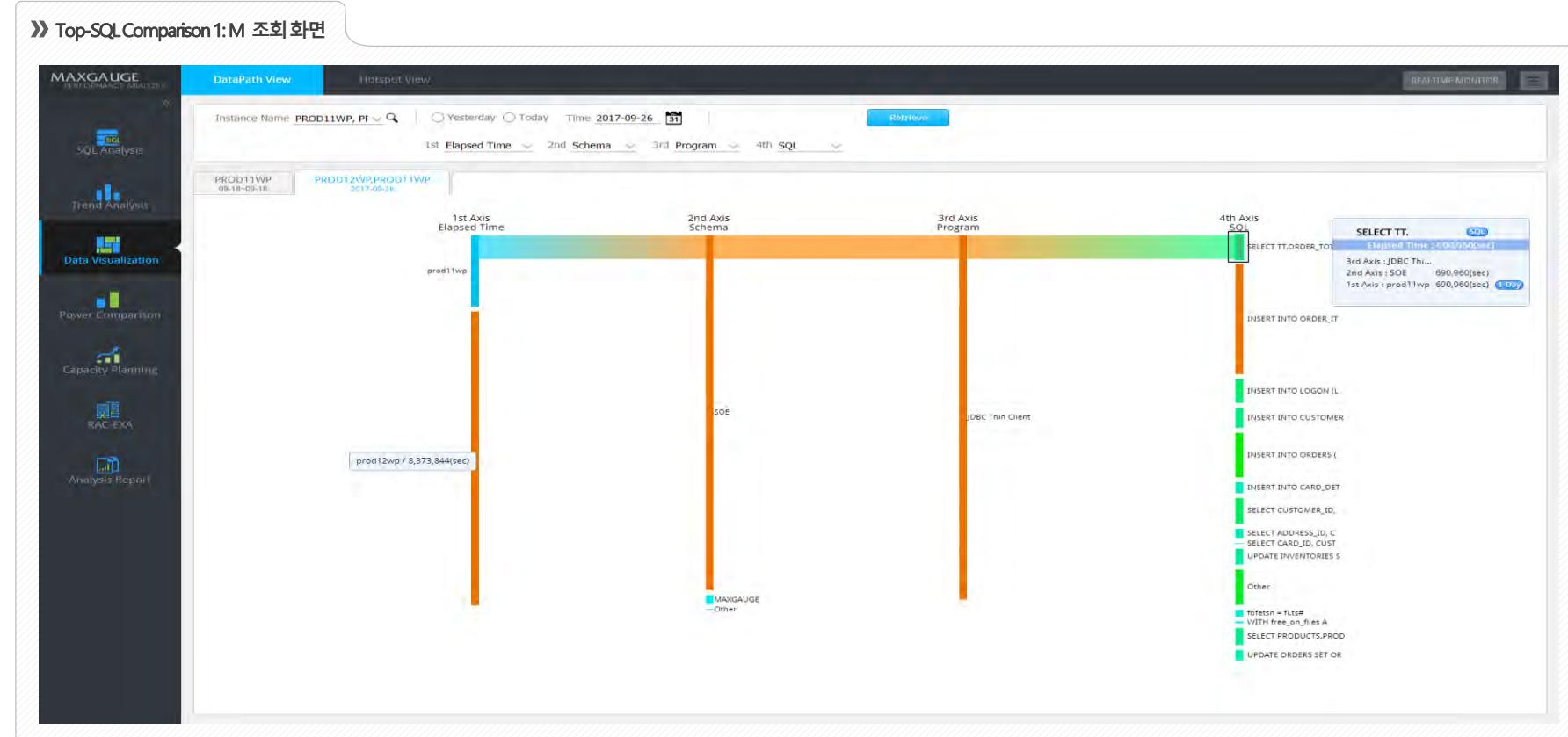
다른 노드에서 실행되면서 gc 이벤트를 발생시키는 SQL을 확인./
 노드 별, SQL의 수행 비중을 쉽게 확인.

“업무 분산은 잘 돼 있나요? / 노드 별 SQL 수행 비중은 어떻게 되죠?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 4차 축 상세 정보 확인

└ SQL Detail 연계



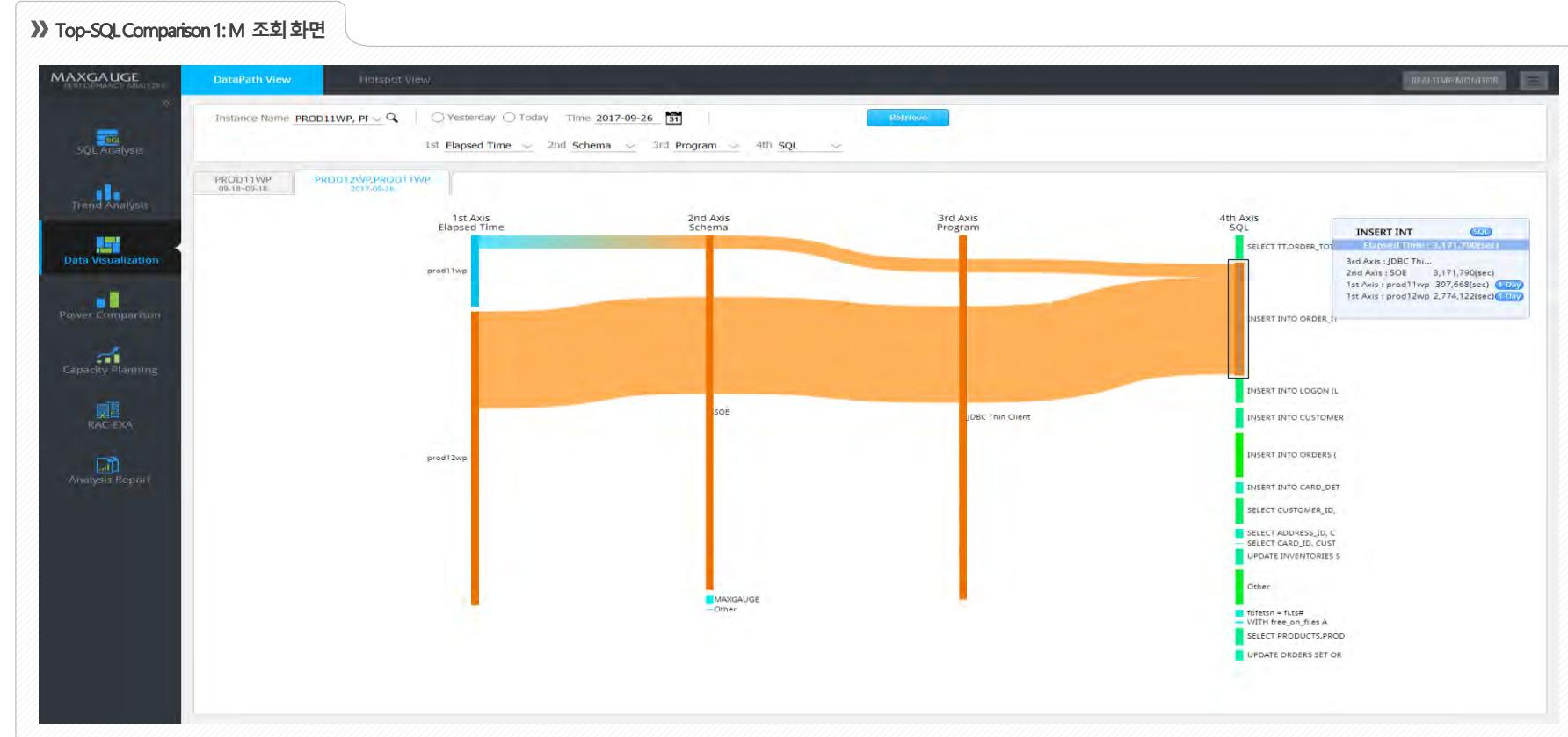
- ✓ 4차 축의 SQL 들 중에서 선택했을 때 위 그림처럼 축 전체가 하나의 노드만 바라봅니다.
- ✓ 이것은 해당 SQL은 하나의 노드에서만 실행되는 것을 뜻합니다.
- ✓ 이와 같은 경우엔 gc 이벤트가 발생 하지 않습니다.

“업무 분산은 잘 돼 있나요? / 노드 별 SQL 수행 비중은 어떻게 되죠?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 4차 축 상세 정보 확인

└ SQL Detail 연계



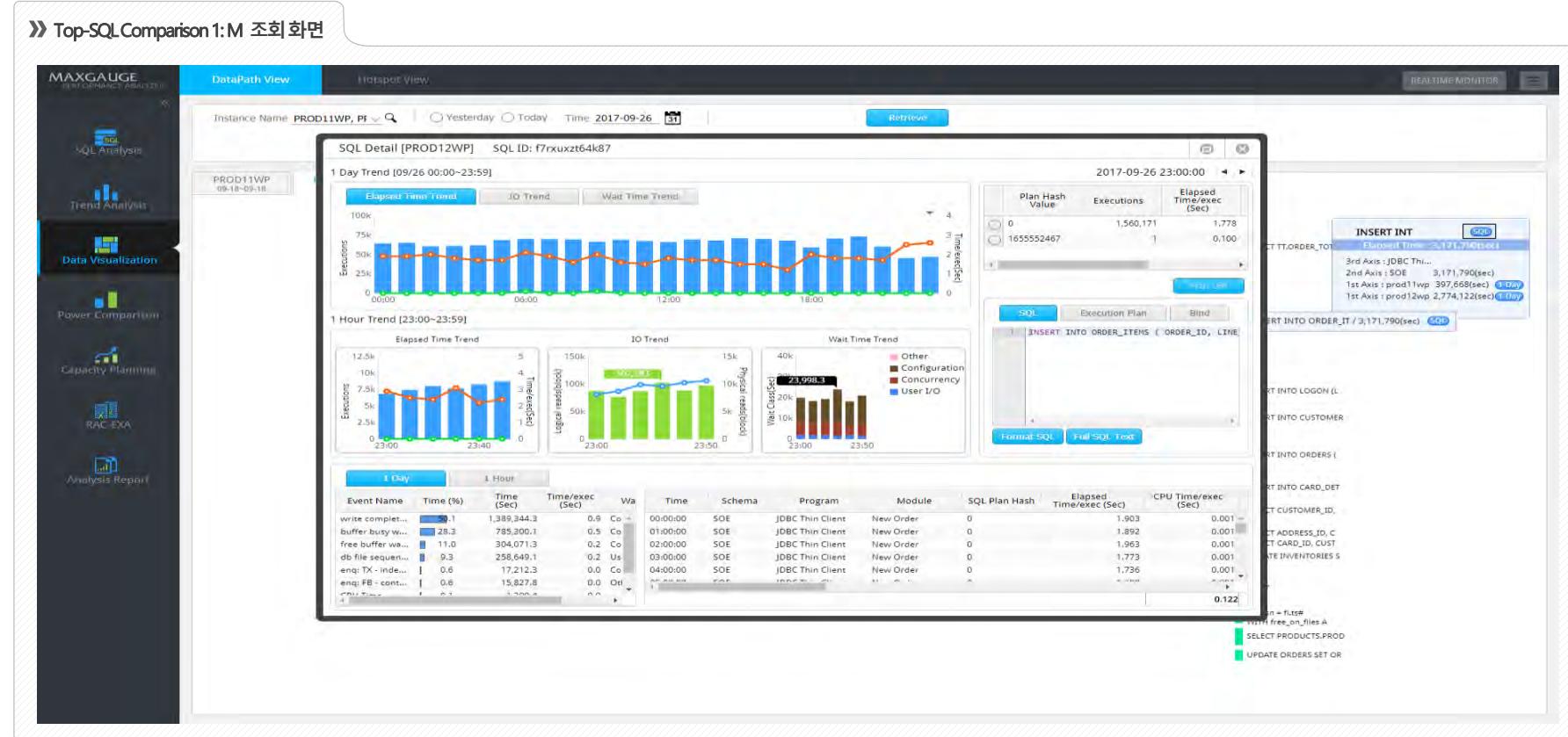
- ✓ 4차 축의 다음 SQL을 선택 했을 때 갈라져 보인다면, 이는 여러 노드에서 실행 되는 SQL을 뜻합니다.
- ✓ 갈라진 갈래가 두꺼운 쪽이 좀 더 비중 있게 실행 되는 노드를 뜻합니다.
- ✓ 이는 gc 이벤트를 발생 시킬 여지가 있으며 업무분산을 다시 확인 해봐야 합니다.

“업무 분산은 잘 돼 있나요? / 노드 별 SQL 수행 비중은 어떻게 되죠?”

DataPath View

- 검색 조건 설정
- 데이터 분석
- 4차 축 상세 정보 확인

└ SQL Detail 연계



✓ 해당 SQL의 상세한 내용을 확인 하려면 우측 상단의 SQL 버튼을 선택 합니다.

✓ 해당 SQL의 SQL detail과 연계 됩니다.

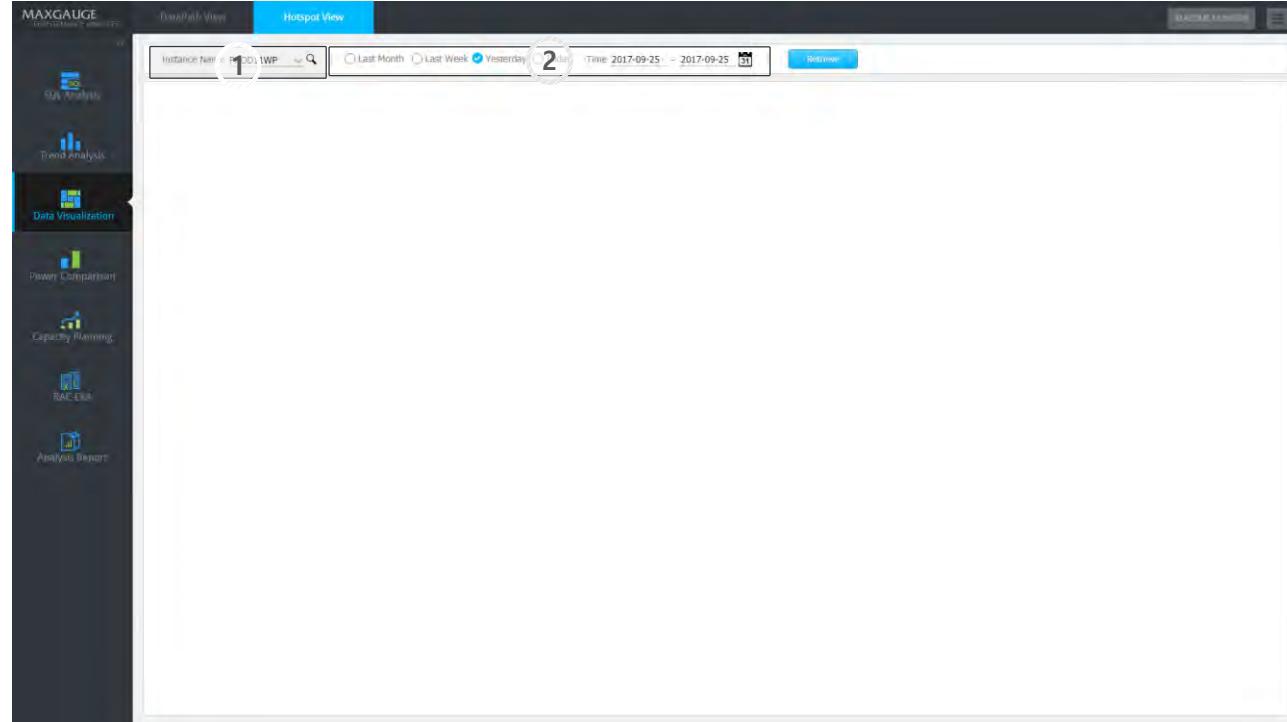
Hotspot View의 활용

Hotspot View의 사용 방법

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계
- Multi instance
 - └ 1-Day Summary View 연계

» Hotspotview 검색화면



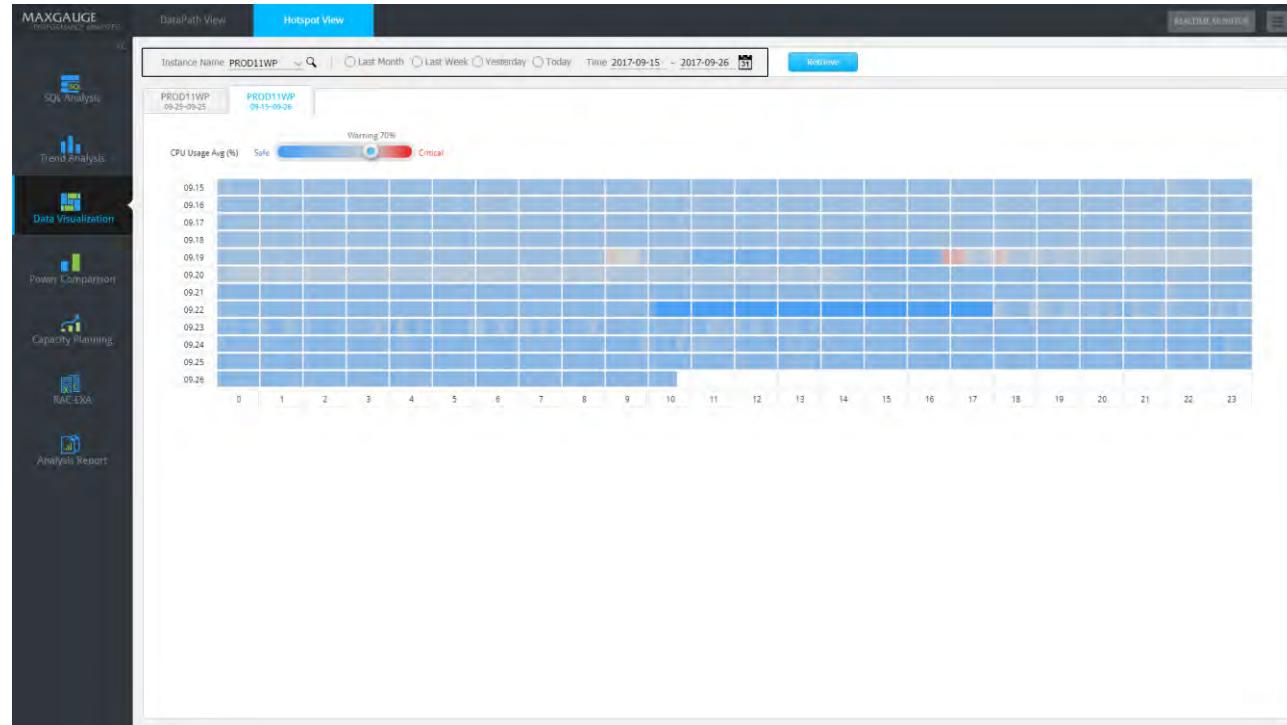
- 1 비교 대상 Instance를 선택합니다.(여러 개의 노드를 선택할 수 있습니다)
- 2 비교 하고자 하는 날짜를 선택 합니다. (다중 인스턴스의 경우 기간 검색이 불가능합니다.)

Hotspot View의 사용 방법

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계
- Multi instance
 - └ 1-Day Summary View 연계

» Hotspotview 검색 결과화면



- ✓ 검색 조건에서 원하는 instance 와 기간을 선택합니다.
- ✓ 선택 된 Instance 를 기준으로 일자와 10분 단위 시간 별 CPU 사용량 분석이 가능합니다.

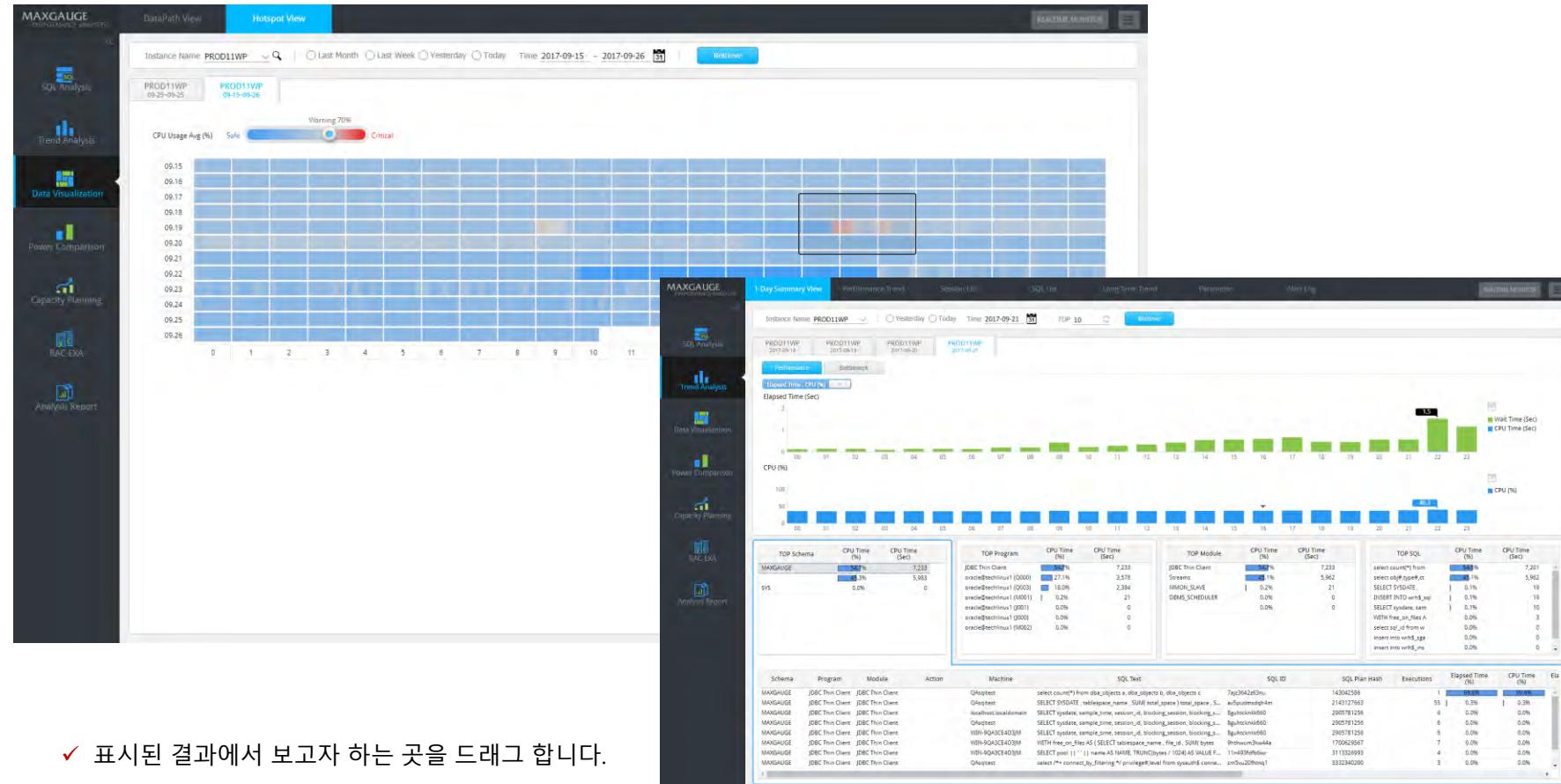
Hotspot View의 활용

Hotspot View의 사용 방법

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계
- Multi instance
 - └ 1-Day Summary View 연계

» Hotspotview 검색 결과화면

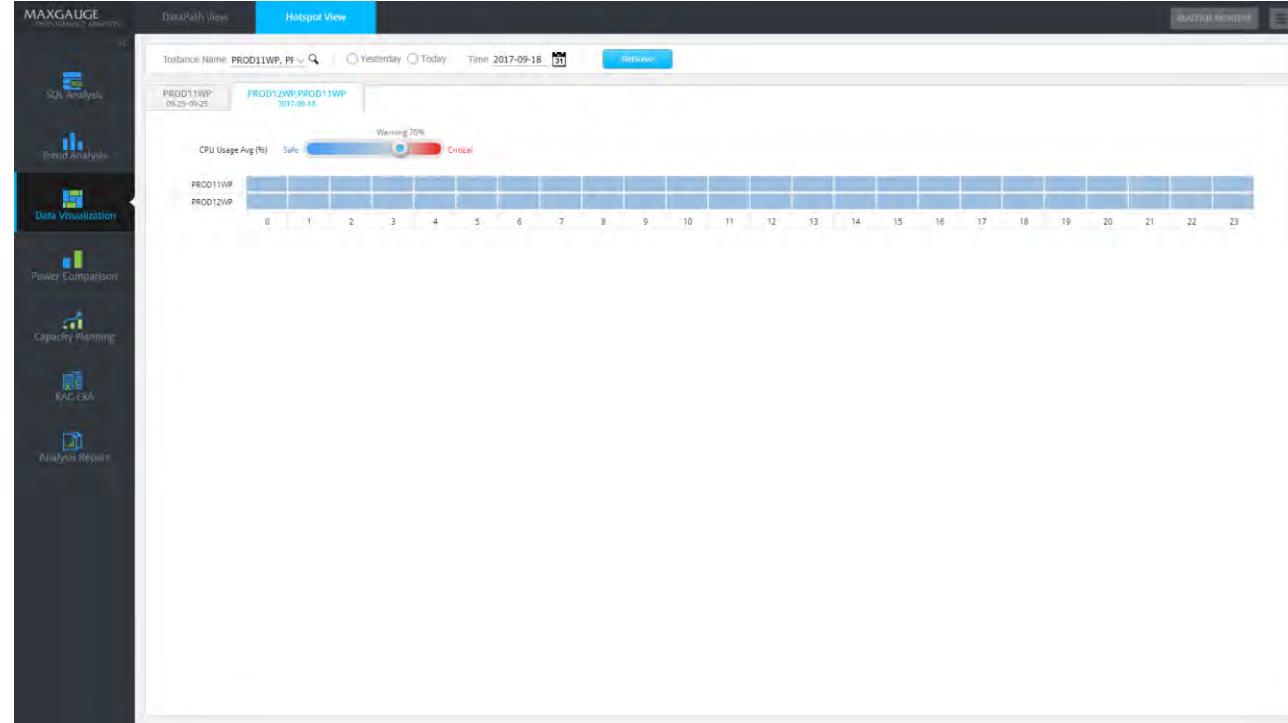


Hotspot View의 사용 방법

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계
- Multi instance
 - └ 1-Day Summary View 연계

» Hotspotview 검색 결과화면



- ✓ 검색 조건에서 원하는 instance들과 날짜 선택합니다. Multi Instance의 경우 하루치의 내용만 비교 가능 합니다.
- ✓ 선택 된 Instance 를 기준으로 일자와 10분 단위 시간 별 CPU 사용량 분석이 가능합니다.

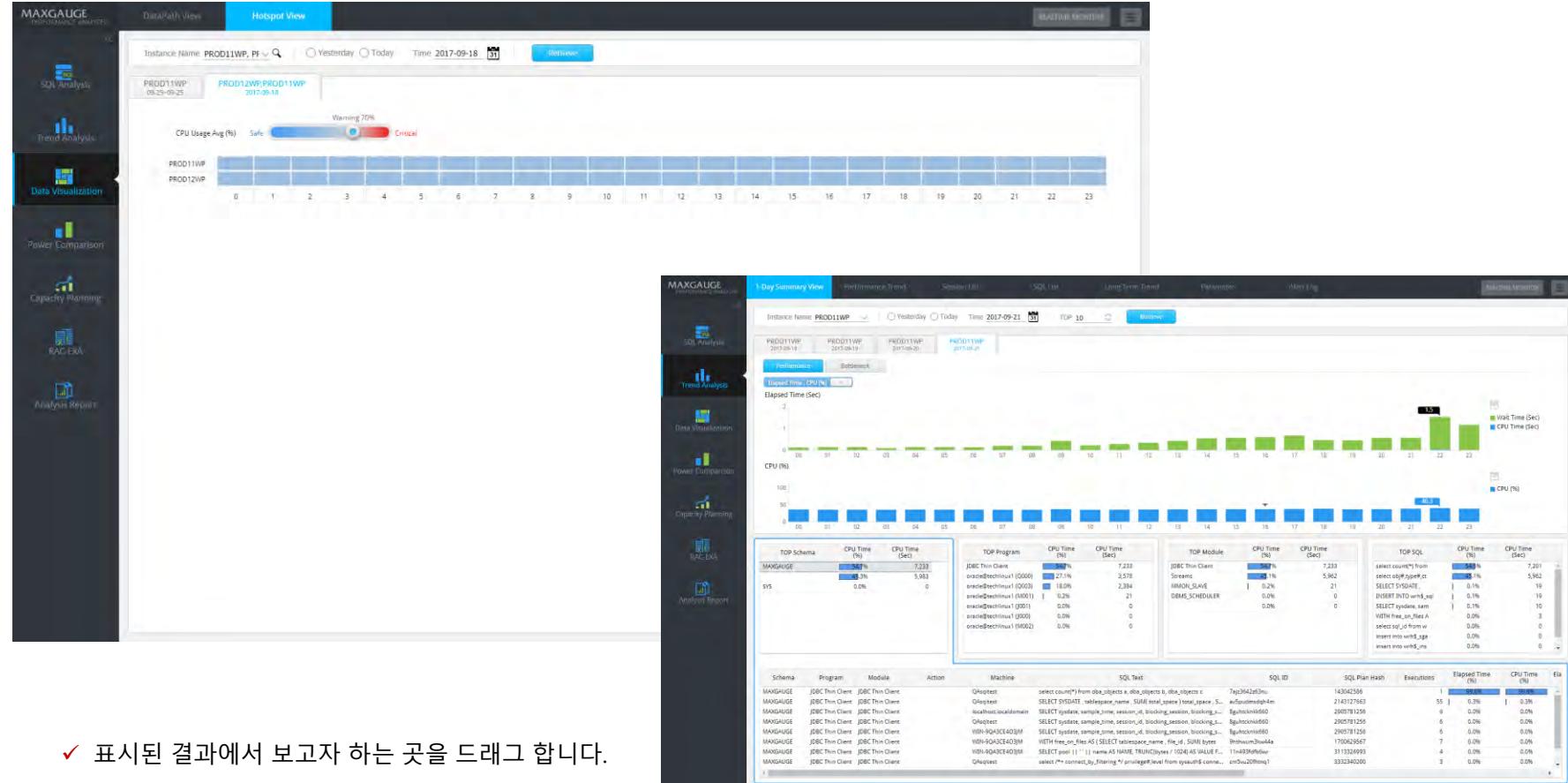
Hotspot View의 활용

Hotspot View의 사용 방법

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계
- Multi instance
 - └ 1-Day Summary View 연계

» Hotspotview 검색 결과화면

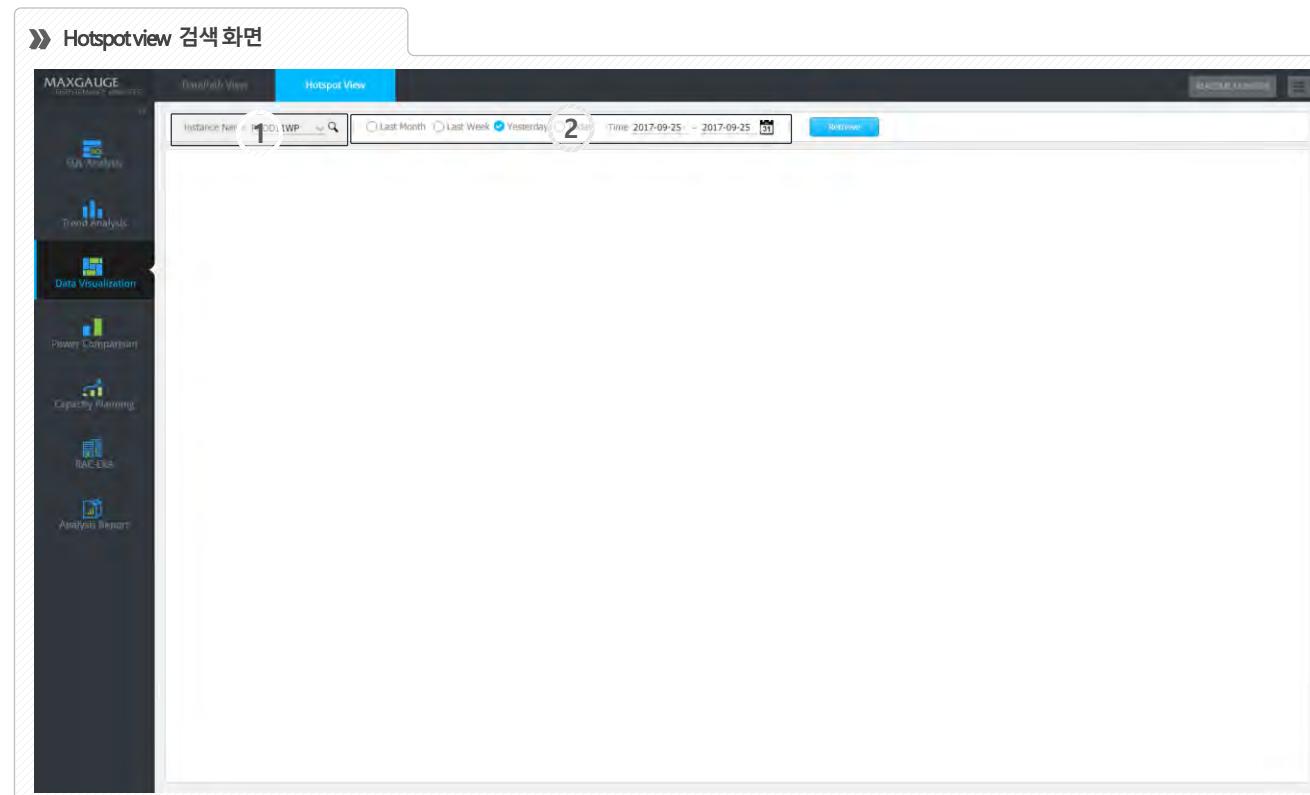


실전 분석 사례 Case 1.
**장기출장 기간 동안 언제 부하가
발생 했고, 원인은 무엇인지 확인
하고 싶어요.**

“언제 부하가 발생 했죠?”

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계



✓ 시나리오

- INSTANCE : PROD1WP
 대상 일자 : 2017년 9월 18일 ~ 26일
 현재상황 : 담당자가 약 열흘 정도 자리
 를 비운 뒤 복귀 했을 때, 그
 사이 리소스 관련 부하 발생.

✓ 목표

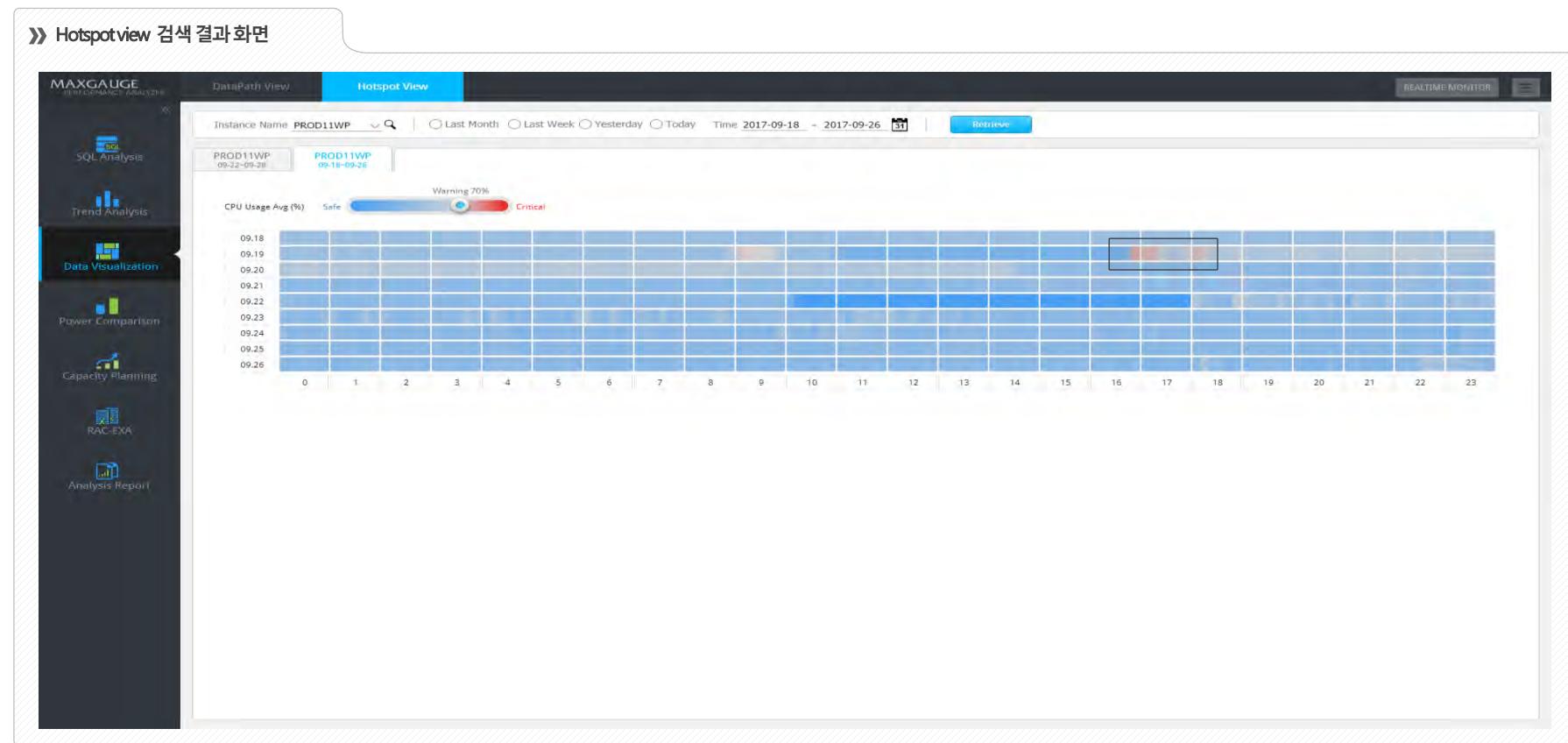
담당자가 자리를 비운 사이 발생한 리소스
 관련 부하 이력을 간단하게 조회 하기.

- 1 관제 대상 Instance (PROD1WP)를 선택합니다.
- 2 분석 날짜 (2017.09.18 ~ 26)는 원하는 기간 만큼 선택 합니다.
- 3 위의 설정한 조건으로 Hotspot View를 실행합니다.

“언제 부하가 발생 했죠?”

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계

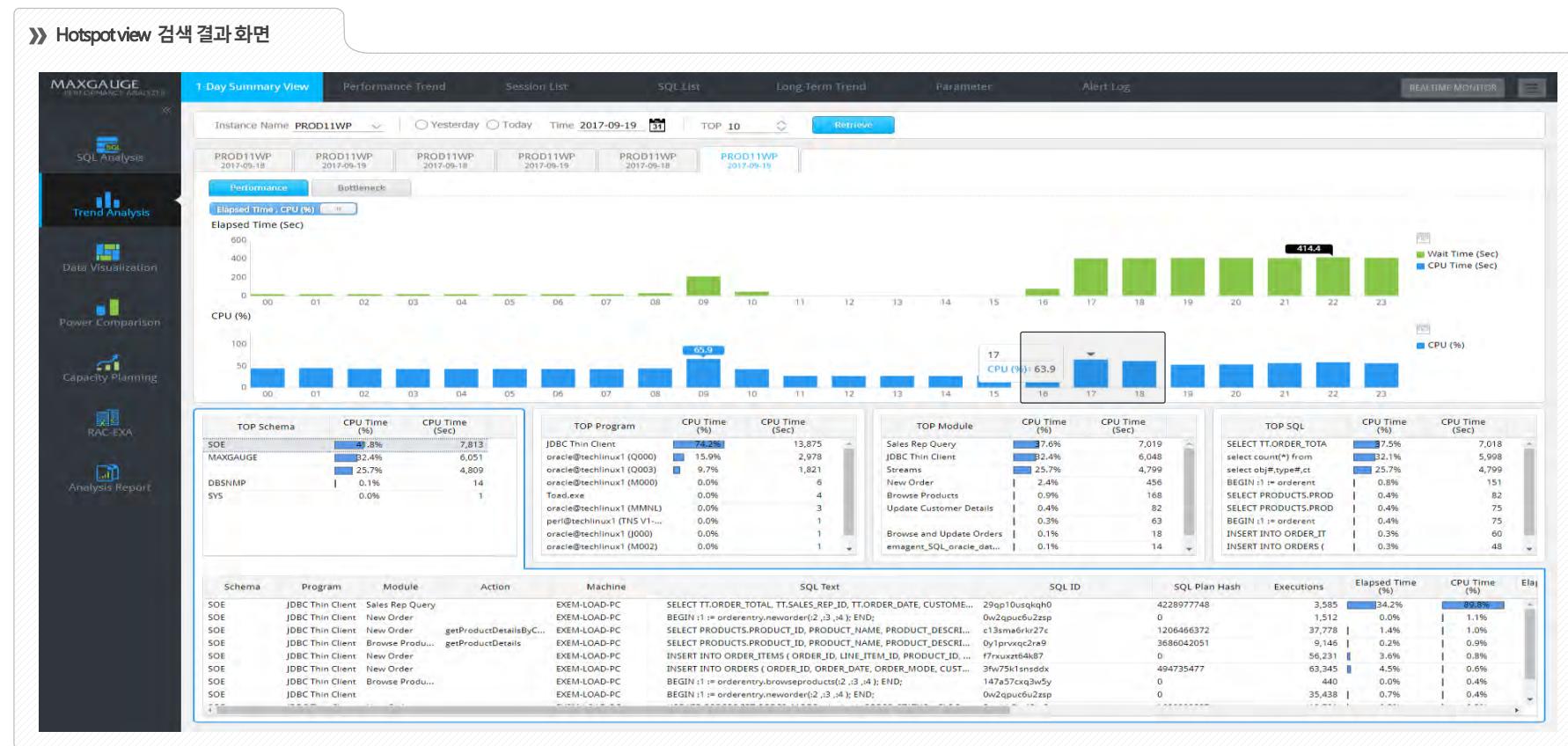


- ✓ 선택한 기간에 해당하는 리소스 부하 정보를 일자는 행으로, 시간은 열로 표시 합니다.
- ✓ 위 화면에서는 9월 19일 16시 ~18시 사이에 CPU 사용률이 70%에 가까운 것으로 보여 집니다.
- ✓ 9월 19일 16시 ~ 18시를 드래그하면, 선택한 날짜와 시간대의 1-Day Summary View로 연계 됩니다.

“언제 부하가 발생 했죠?”

Hotspot View

- 검색 조건 설정
- 데이터 분석
- Single instance
 - └ 1-Day Summary View 연계

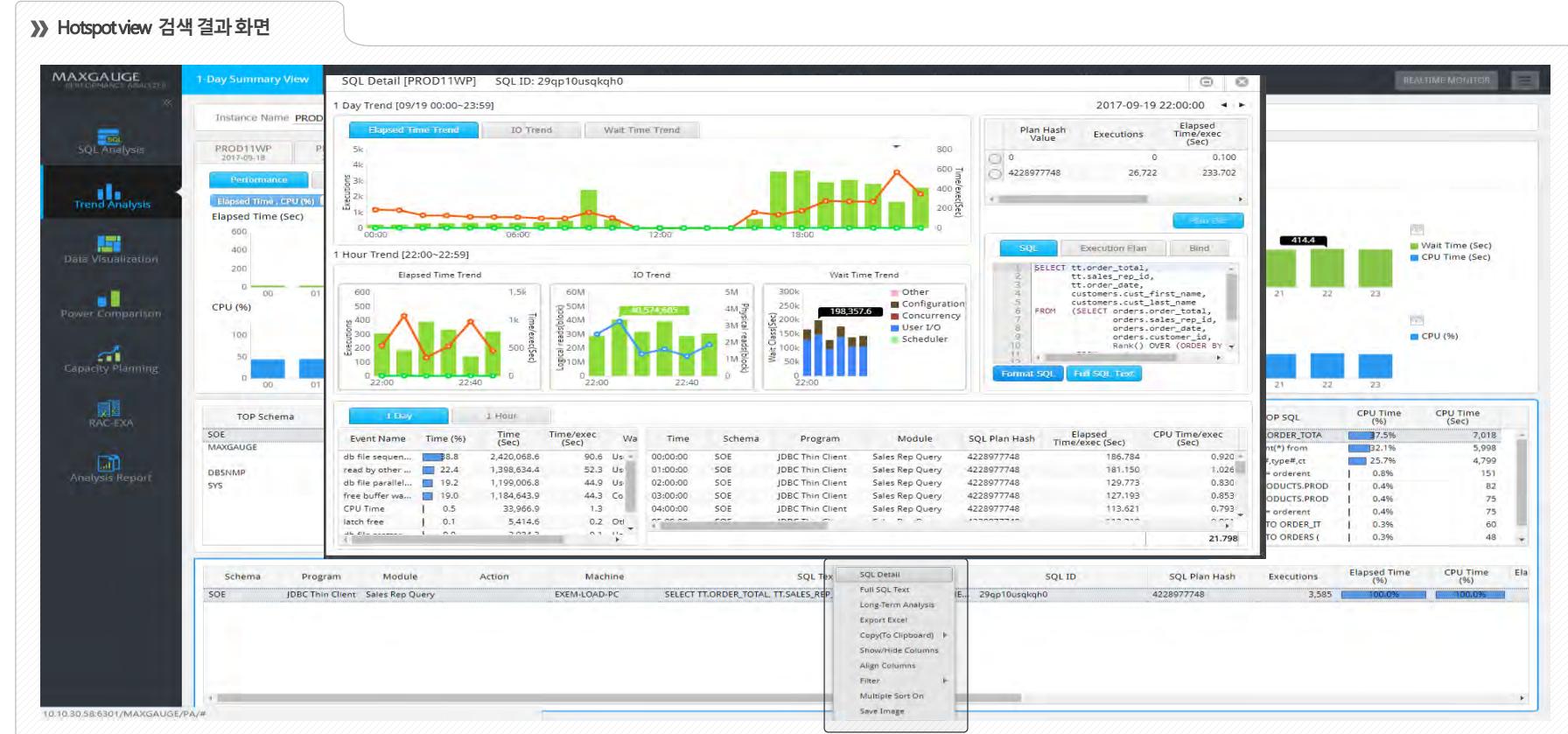


- ✓ 연계된 1-Day Summary View 화면에서 확인한 결과 63.9%의 사용률을 보였던 것으로 분석 됩니다.
- ✓ 화면 하단에서 해당 시간대에서 실행된 정보들을 확인 할 수 있습니다.

“언제 부하가 발생 했죠?”

Hotspot View

- 검색 조건 설정
- 데이터 분석
 - Single instance
 - └ 1-Day Summary View 연계



- ✓ 해당 시간에 실행 된 TOP – SQL 을 확인 할 수 있습니다.
- ✓ SQL List 를 마우스로 우클릭 하게 되면 SQL Detail 정보를 확인 할 수 있습니다.

MAXGAUGE



Practical Guide

Event Comparison [Performance Analyzer]

Contents

Event Comparison?

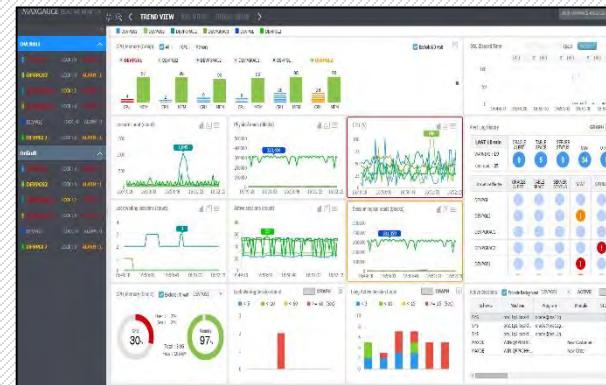
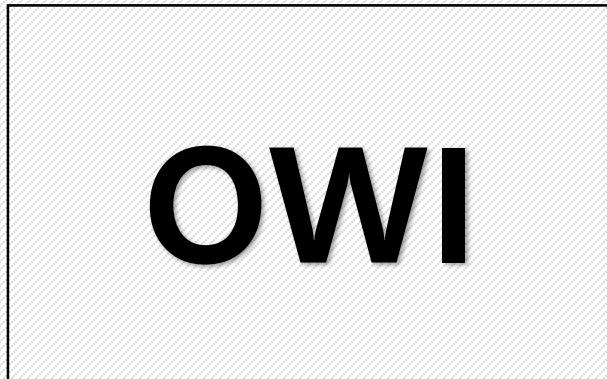
Event Comparison의 활용

Case 1. 전 주에 비해 늘어난 이벤트와 그 원인을 알고 싶어요 (N:M 분석)

Case 2. 신규로 추가 된 업무가 발생 시키는 대기 현상이 궁금해요(1:M 분석)

Event Comparison?

Event Comparison의 분석 방법론



Oracle Wait Interface

Oracle은 항상 단서를 남긴다
Oracle Process가 겪는 대기현상

OWI 방법론의 MaxGauge

Oracle이 남긴 단서(OWI)를 토대로
설계된 전문적인 성능관리 도구

OWI 방법론의 MaxGauge 활용

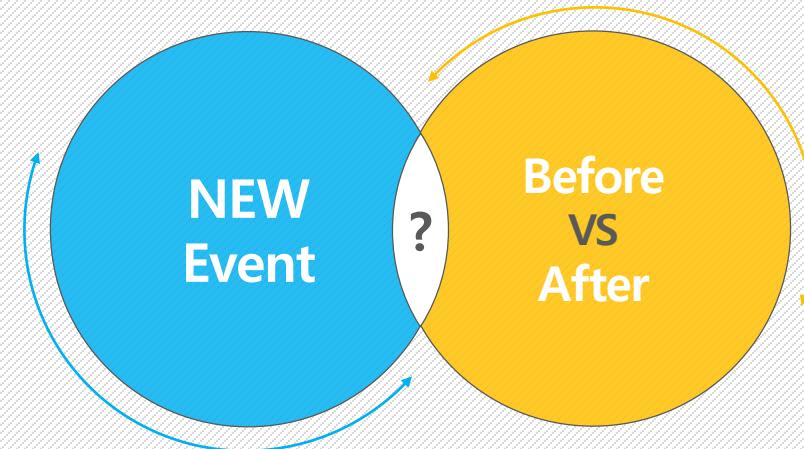
대기 이벤트 비교 분석을 통한
진단 및 성능 지연 현상 추적

Event Comparison은 언제 쓰나요?

신규로 추가된 업무가
발생 시키는 대기 현상이 궁금할 때



평소에 발생하지 않았던 대기 현상을
발견했을 때



» 기준 날짜 대비 비교 날짜에 발생한 대기
현상을 Wait Class 단위로 분석할 때.

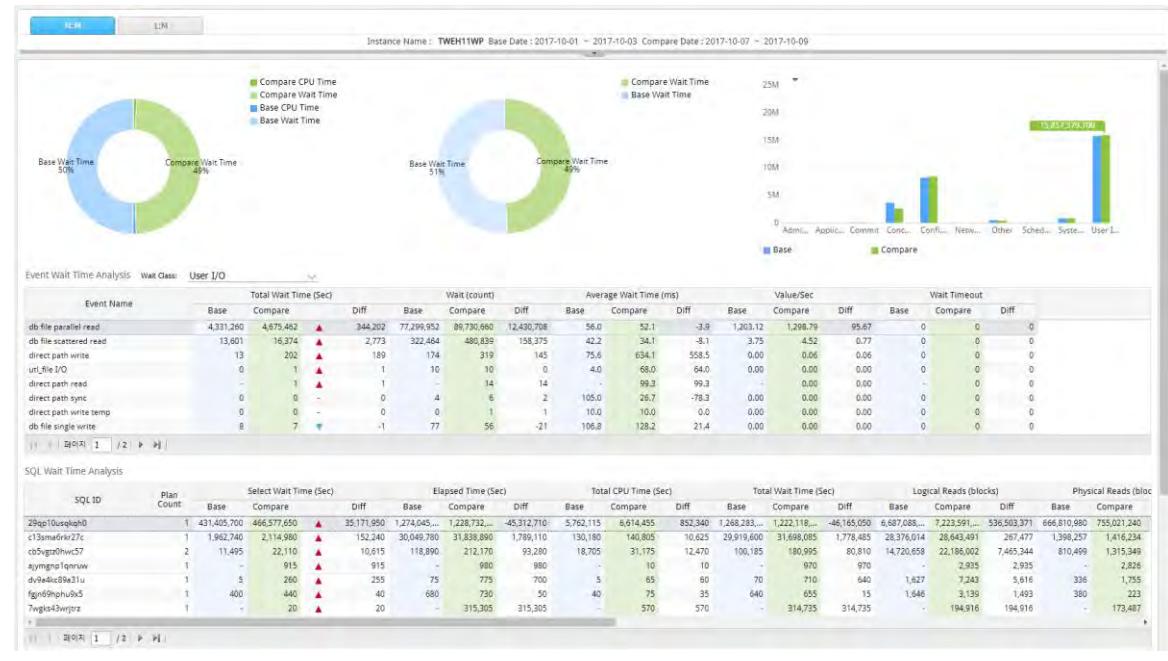
» 대기 현상을 유발하는 SQL을 추적할 때.

Event Comparison?

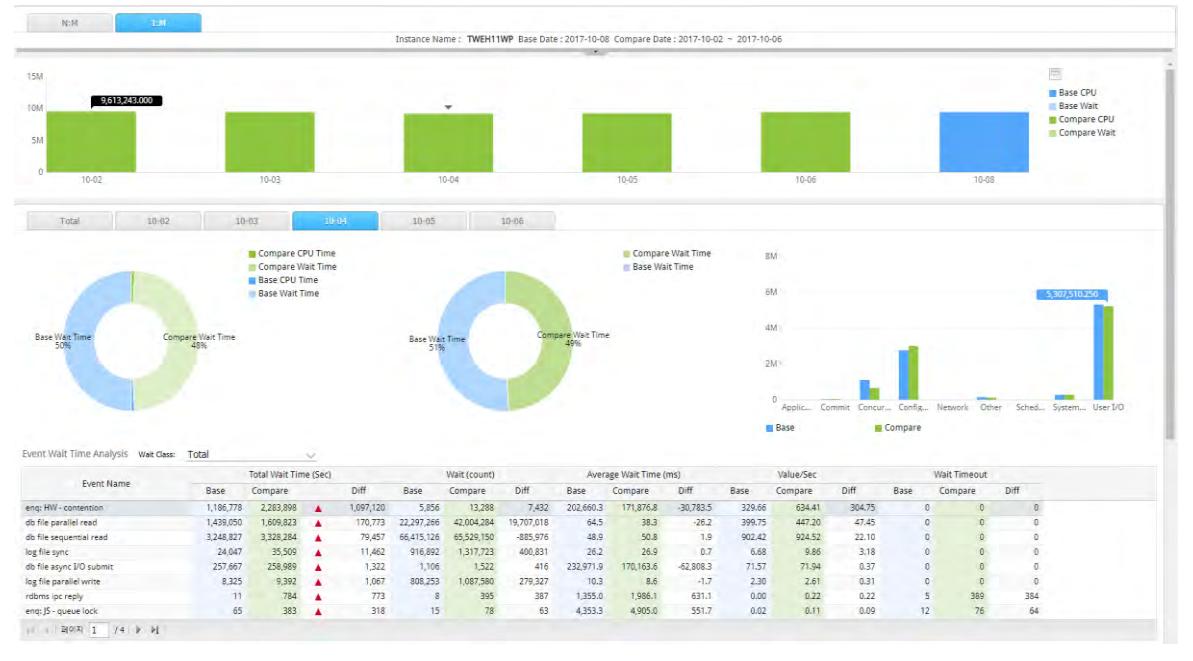
Event Comparison

- 기준 날짜 대비 비교 날짜들의 Wait Event 발생 현황과 특정 Event로 대기했던 SQL의 비교 기능이다.
- Total Wait Time, Wait Class** 기준으로 Base Time과 Compare Time 두 구간에 대한 1:M, N:M 비교 분석을 지원한다.
- 특정 Event를 선택하면, 선택된 Event로 대기했던 SQL List를 출력하고, SQL의 상세 정보를 확인할 수 있다.
- Base Time 기준으로 Wait Time이 증가한 Event와 SQL은 “▲”, Wait Time이 감소한 Event와 SQL은 “▼” 형태의 표현 방식을 지원한다.
- 해당 기능은 Performance Analyzer ▶ Power Comparison ▶ Event Comparison 경로를 통해 사용할 수 있다.

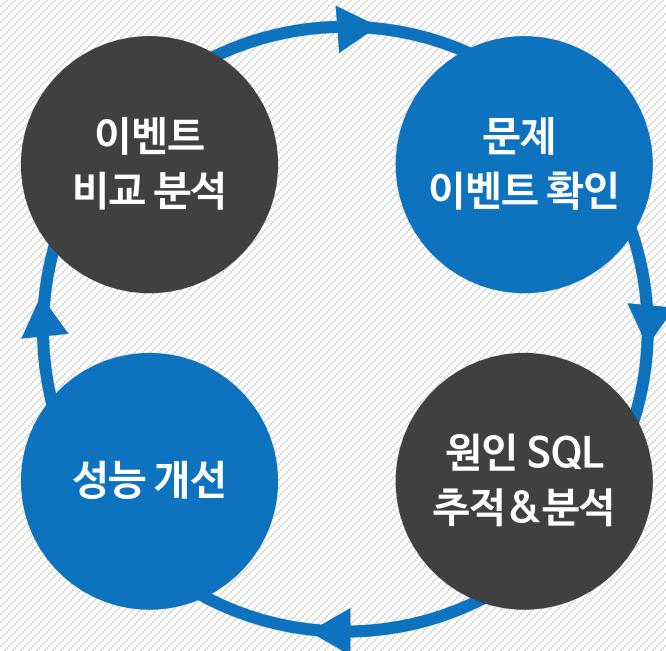
» N:M Comparison



» 1:M Comparison



OWI 기반 지속적인 성능 관리를 할 수 있습니다!



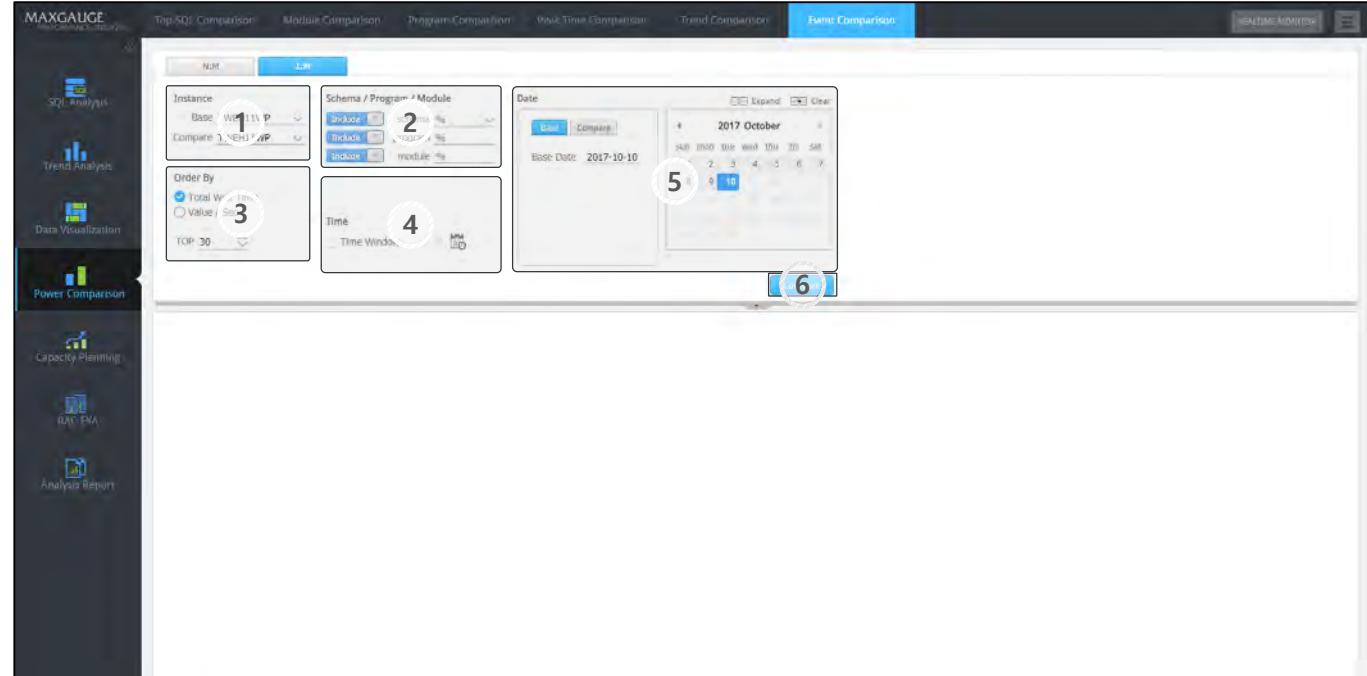
Event Comparison의 활용

Event Comparison의 사용 방법 (1:M)

Event Comparison (1:M)

- 검색 조건 설정
- 데이터 분석

» EventComparison 1:M 검색화면



- 1 비교 대상 Instance를 선택합니다.
- 2 Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 정렬 기준을 선택하고, 출력되는 Top Event의 개수를 설정합니다. 정렬 기준은 Total Wait Time과 Value / Sec이 있습니다.
- 4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- 5 기준 날짜(Base Date)는 하루만 선택되며, 비교 날짜(Compare Date)는 Working day, Week, 특정 날짜로 선택 가능합니다.
- 6 위의 설정한 조건으로 Event Comparison를 실행합니다.

Event Comparison의 활용

Event Comparison의 사용 방법 (1:M)

Event Comparison (1:M)

- 검색 조건 설정
- 데이터 분석

» Event Comparison 1:M 조회 화면

The screenshot displays the MAXGAUGE Event Comparison 1:M interface. At the top, there are several tabs: Top SQL Comparison, Manual Comparison, Program Comparison, Block Time Comparison, Event Comparison, and Event Comparison. The 'Event Comparison' tab is active.

1. Top-level Analysis: A large bar chart at the top shows CPU and Wait time for five days (10-02 to 10-06). A callout '1' points to the chart area.

2. Event Wait Time Analysis: A table comparing 'Event Name', 'Total Wait Time (Sec)', 'Wait (Count)', 'Avg Wait Time (msec)', 'Value/Sec', and 'Timeout (Count)' between 'Base (2017-10-02)' and 'Compare (2017-10-03)'. A callout '2' points to the table.

3. SQL Time Analysis: A table comparing 'SQL ID', 'Plan Count', 'Select Wait Time (Sec)', 'Elapsed Time (Sec)', 'Wait Time (Sec)', 'CPU Time (Sec)', 'L Reads (block)', 'P Reads (block)', and 'Execution (count)' between 'Base (2017-10-02)' and 'Compare (2017-10-03)'. A callout '3' points to the table.

4. SQL Text & Execution Plan: A table showing 'Plan Hash Value', 'Base', 'Compare', and 'Diff' for 'Select Wait Time (Sec)' and 'Elapsed Time (Sec)'. A callout '4' points to the table.

5. Detailed Execution Plan: A detailed view of the execution plan for a specific query, showing 'Plan Hash Value: 422897748'. It includes sections for 'Plan Hash Value', 'SQL', and 'Execution Plan'. A callout '5' points to the execution plan details.

기준 날짜와 비교 날짜의 CPU Time과 Wait Time을 일자별 그래프로 제공합니다.

선택한 일자별(base Date, Compare Date) Total Wait Time 기준으로 Top Event 정보를 제공합니다.

Event Wait Time Analysis에서 선택된 Event를 발생 시킨 SQL 정보를 Total Wait Time 기준으로 제공합니다.

SQL Time Analysis에서 선택된 SQL의 성능 정보를 Plan Hash Value 단위로 제공하며, 일자별 수행 이력을 제공합니다.

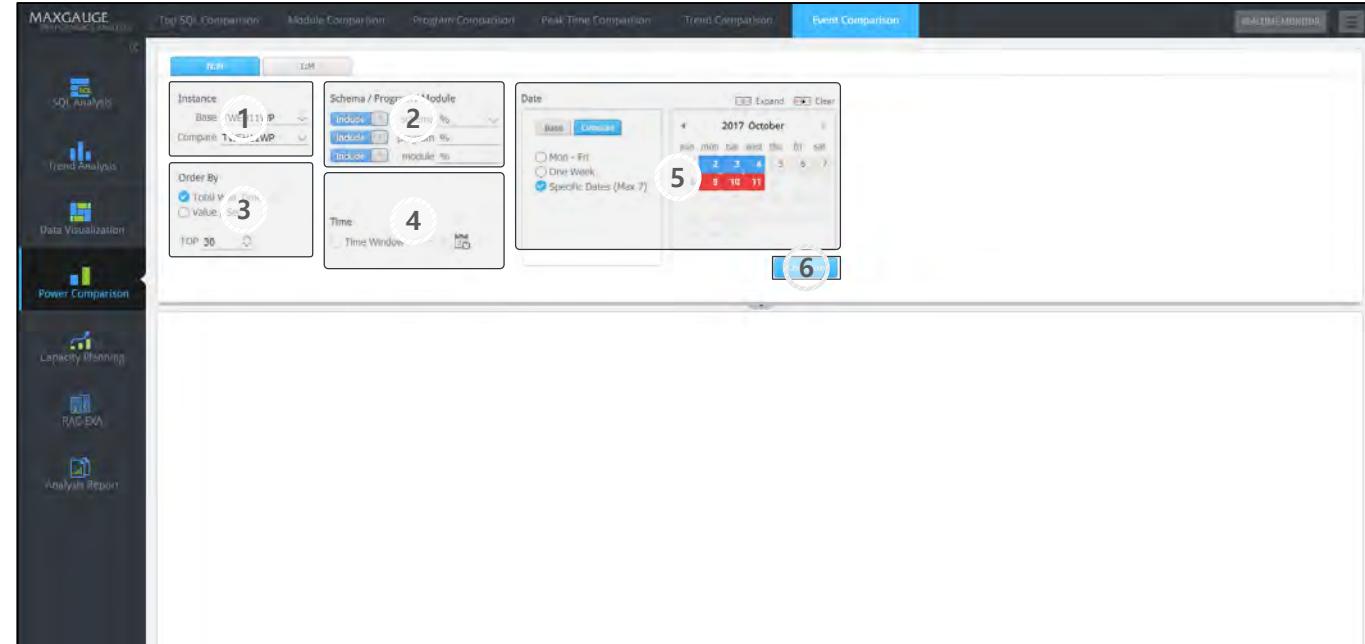
선택한 SQL에 대한 SQL Text 및 Bind, Execution Plan 정보를 제공합니다.

Event Comparison의 사용 방법 (N:M)

Event Comparison (N:M)

- 검색 조건 설정
- 데이터 분석

» Event Comparison N:M 검색 화면



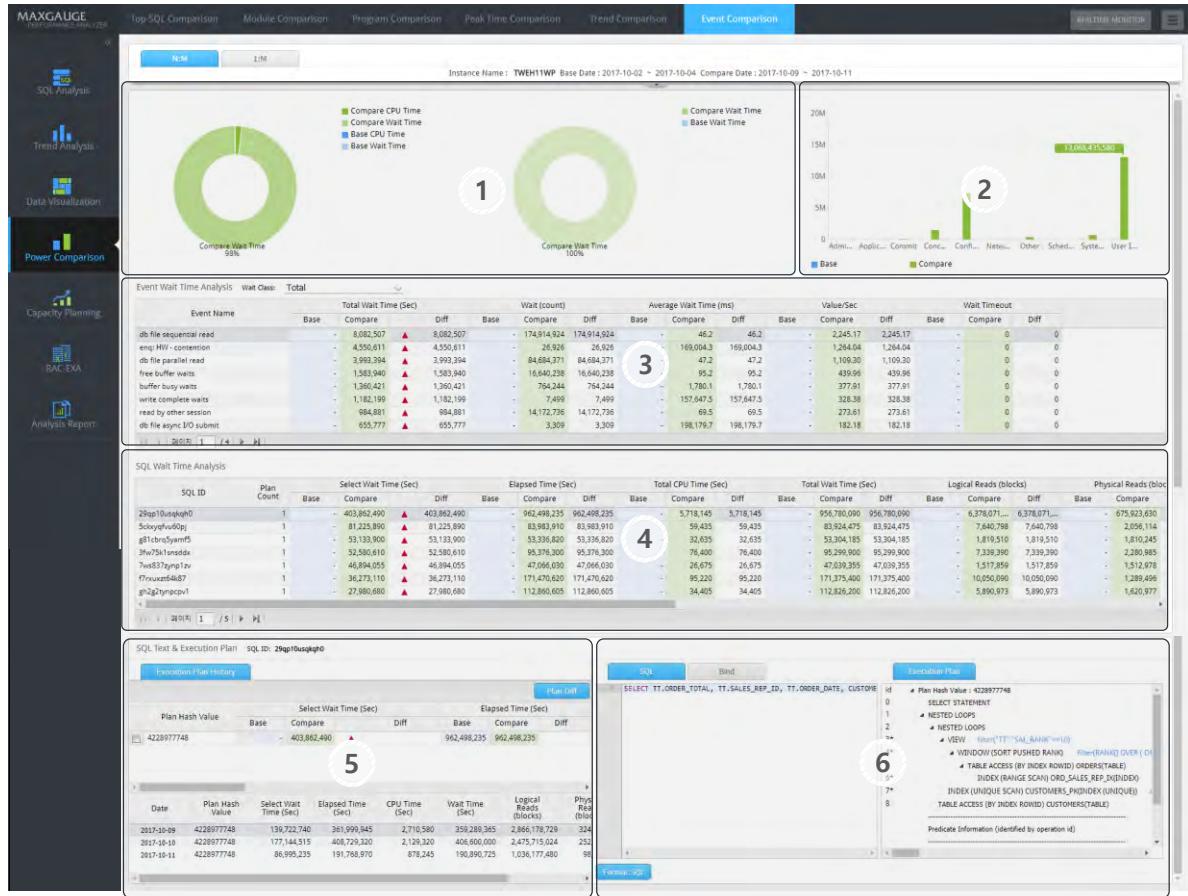
- 1 비교 대상 Instance를 선택합니다.
- 2 Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 정렬 기준을 선택하고, 출력되는 Top Event의 개수를 설정합니다. 정렬 기준은 Total Wait Time과 Value / Sec이 있습니다.
- 4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- 5 기준 날짜(Base Date)와 비교 날짜(Compare Data) 모두 Working day, Week, 특정 날짜로 선택 가능합니다.
- 6 위의 설정한 조건으로 Event Comparison 실행합니다.

Event Comparison의 사용 방법 (N:M)

Event Comparison (N:M)

- 검색 조건 설정
- 데이터 분석

» Event Comparison N:M 조회 화면



- 기준 날짜와 비교 날짜의 **CPU Time**과 **Wait Time**의 비율을 원형 그래프로 나타냅니다.
- 기준 날짜와 비교 날짜의 **Wait Class** 단위 **Wait Time**을 그래프로 제공합니다.
- 기준 날짜와 비교 날짜의 **Total Wait Time** 기준으로 **Top Event** 정보를 제공합니다.
- Event Wait Time Analysis에서 선택된 **Event**를 발생 시킨 SQL 정보를 **Total Wait Time** 기준으로 제공합니다.
- SQL Time Analysis에서 선택된 **SQL**의 성능 정보를 **Plan Hash Value** 단위로 제공하며, 일자 별 수행 이력을 제공합니다.
- 선택한 SQL에 대한 **SQL Text** 및 **Bind, Execution Plan** 정보를 제공합니다.

실전 분석 사례 Case 1.
**전 주에 비해 늘어난 대기이벤트와
그 원인을 알고 싶어요 (N:M 분석)**

“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- Wait Class 단위 분석
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인

» Event Comparison 검색 화면

- 1 비교 대상 Instance (TPROD11WO)를 선택합니다.
- 2 Schema (**MAXGAUGE**), Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 정렬 기준을 선택하고, 출력되는 Top Event의 개수(TOP 30)를 설정합니다. 정렬 기준은 Total Wait Time과 Value / Sec이 있습니다.
- 4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (09:00 ~ 18:00) 을 선택적으로 선택 할 수 있습니다.
- 5 기준 날짜(2017.09.18~21)와 비교 날짜(2017.09.25~29) 모두 Working day, Week, 특정 날짜로 선택 가능합니다.
- 6 위의 설정한 조건으로 Event Comparison 실행합니다.

✓ 시나리오

INSTANCE	: TPROD11WO
SCHEMA	: MAXGAUGE
Base Date	: 2017년 9월 18 ~ 22일
Compare Date	: 2017년 9월 25 ~ 29일
업무 집중 시간	: 09:00 ~ 18:00

✓ 목표

기준 날짜와 비교 날짜를 비교 분석하여 시스템에 영향을 주는 대기 이벤트를 확인하고, SQL 추출 및 성능 저하 원인 파악.

“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

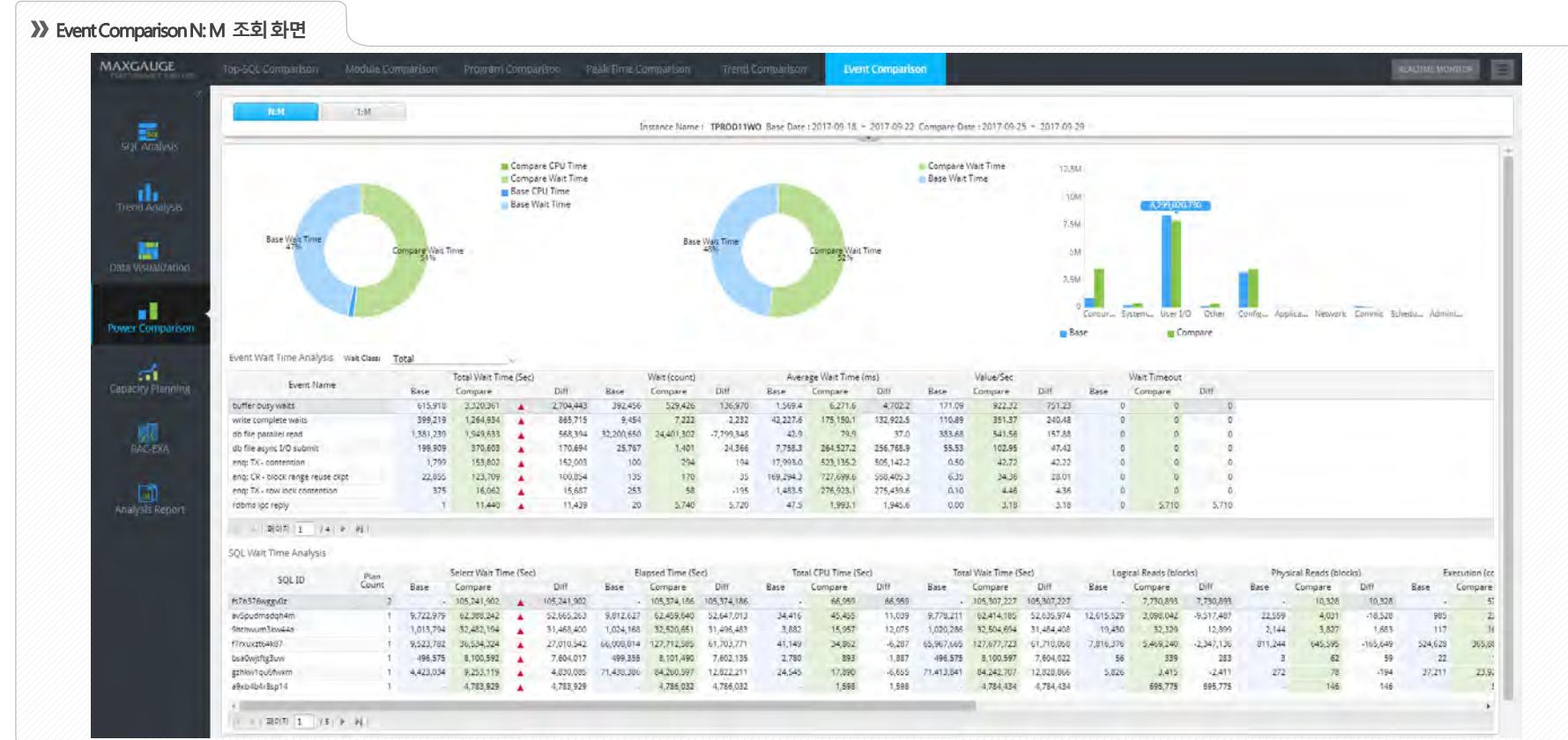
STEP

1. 검색 조건 설정

2. 데이터 분석

▪ 출력 데이터 확인

- Total Wait Time 비교
- Wait Class 단위 분석
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인



- ✓ 상단 그래프를 통하여 Base Wait Time과 Compare Wait Time 값을 비교 분석합니다.
- ✓ Event Wait Time Analysis에서 ▲로 표시된 이벤트가 있는지 확인합니다.
- ✓ ▲로 표시된 이벤트는 Wait Time이 증가한 이벤트이며, Diff 컬럼 값을 통하여 Wait Time의 차이를 확인할 수 있습니다.
- ✓ Event Wait Time Analysis에서 선택한 Event를 발생 시킨 SQL은 SQL Wait Time Analysis에서 확인할 수 있습니다.
- ✓ SQL Wait Time Analysis에서 ▲로 표시된 SQL은 Wait Time이 증가한 SQL입니다.

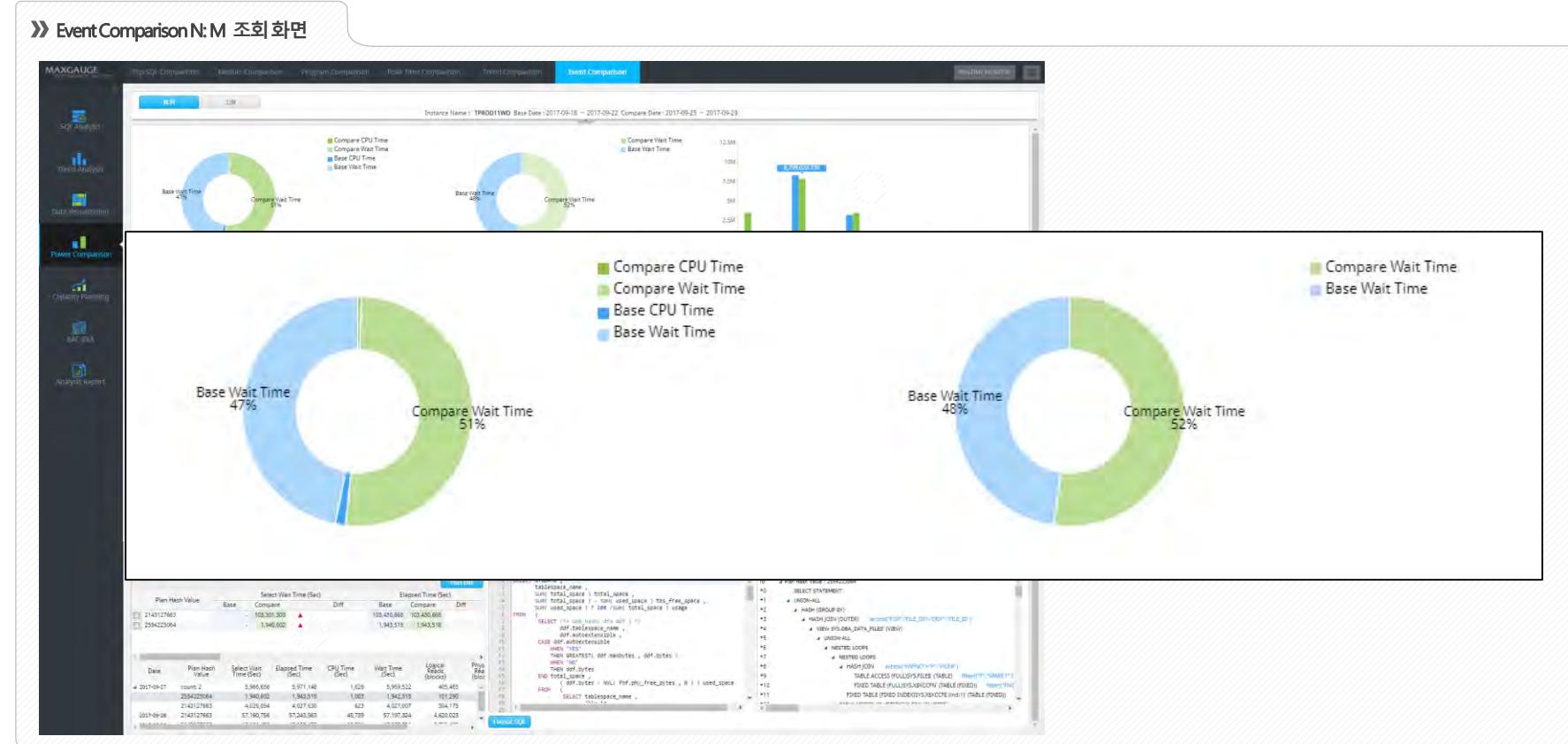
“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
 - **Total Wait Time** 비교
 - Wait Class 단위 분석
 - 발생 대기 이벤트 확인
 - 대기 이벤트 정보 확인
 - 대기 이벤트 발생 SQL
 - SQL 상세 정보 확인



- ✓ 상단 그레프를 통하여 Base Wait Time과 Compare Wait Time 값을 비교 분석합니다.
 - ✓ 좌측의 파란색은 base Wait Time을, 우측의 초록색은 Compare Wait Time을 나타냅니다.
 - ✓ Base Wait Time과 Compare Wait Time의 비율을 원형 그레프로 표현합니다.
 - ✓ Compare Wait Time이 차지하는 비율이 52%로, Wait Time이 증가했음을 확인할 수 있습니다.

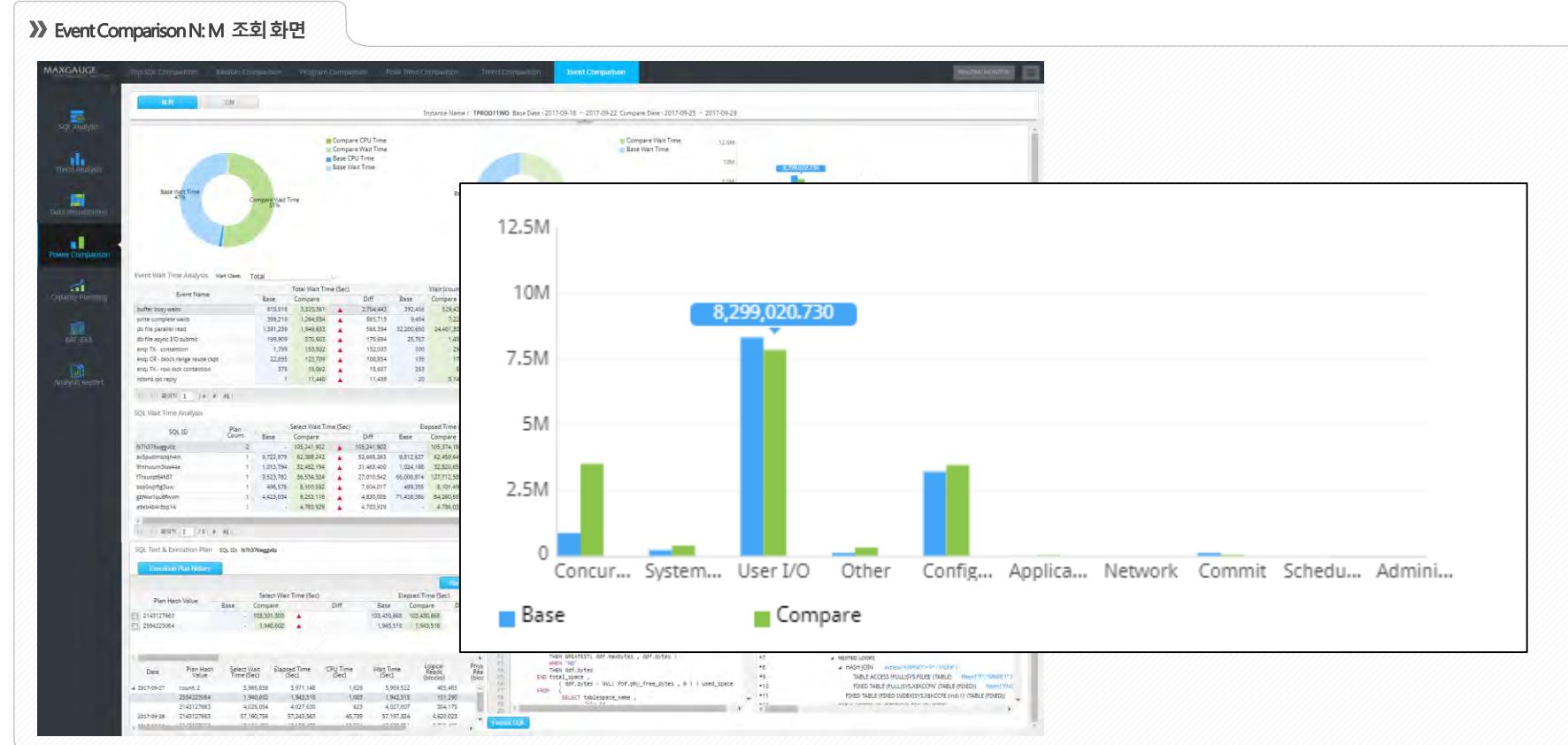
“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- **Wait Class 단위 분석**
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인



- ✓ Wait Class 단위 Base Wait Time과 Compare Wait Time 값을 비교 분석합니다.
- ✓ 좌측의 파란색은 base Wait Time을, 우측의 초록색은 Compare Wait Time을 나타냅니다.
- ✓ Concurrency Class에서 기준날짜 대비 비교 날짜의 Wait Time이 크게 증가한 것을 확인할 수 있습니다.

“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인

- Total Wait Time 비교

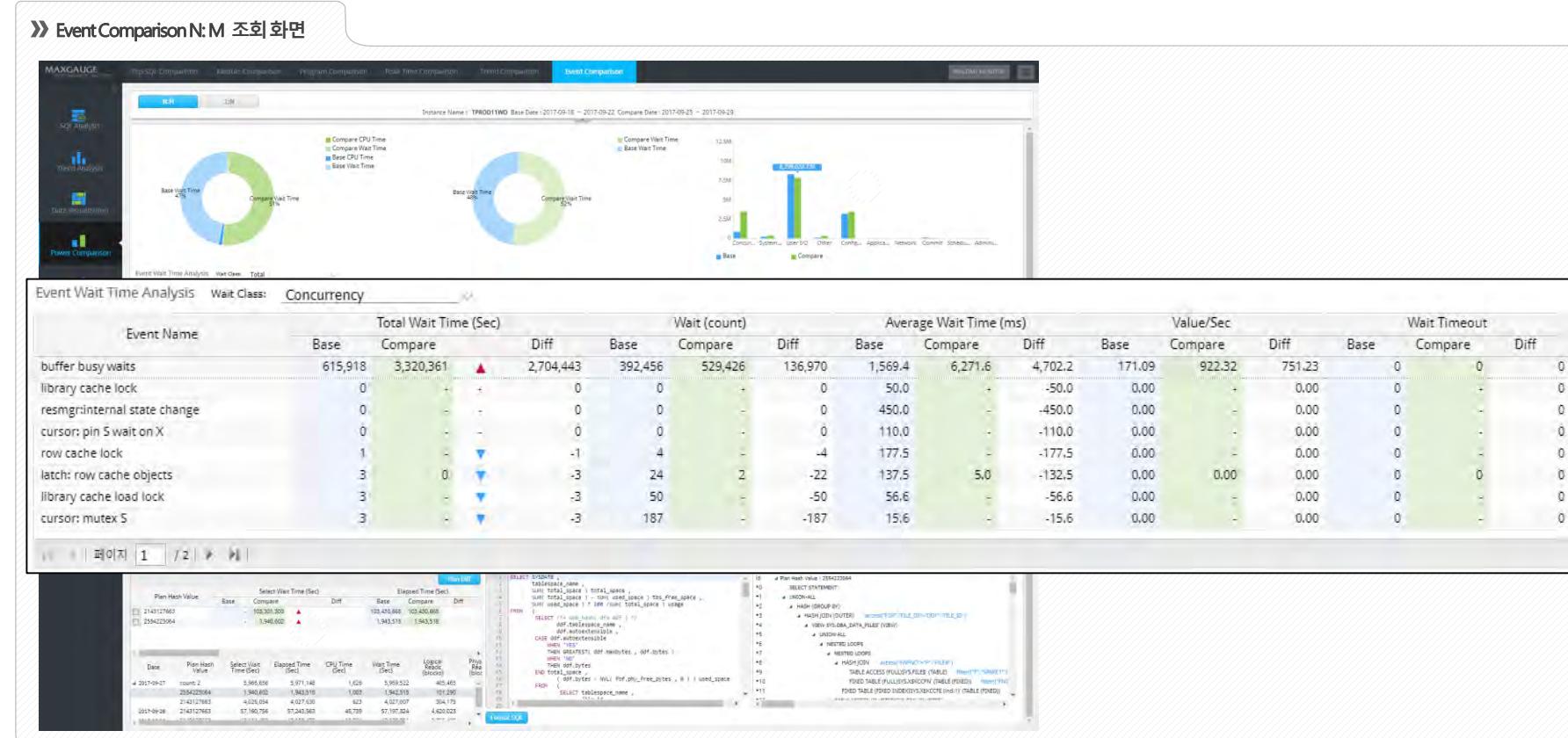
- Wait Class 단위 분석

▪ 발생 대기 이벤트 확인

- 대기 이벤트 정보 확인

- 대기 이벤트 발생 SQL 추적

- SQL 상세 정보 확인



✓ Event Wait Time Analysis에서 Wait Class를 Concurrency로 설정합니다.

✓ Concurrency Class에 포함된 이벤트 중에, Base 구간과 Compare 구간의 Total Wait Time의 차이(Diff)가 큰 이벤트 순으로 데이터가 출력됩니다.

✓ Buffer busy waits 이벤트가 base 구간에 비해서 2,704,443초 증가했음을 확인할 수 있습니다.

Event Comparison의 활용

"전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- Wait Class 단위 분석
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인

» Event Comparison N:M 조회 화면

The screenshot displays the MAXGAUGE Event Comparison N:M interface. It includes several sections:

- Event Description**: Shows a pie chart for 'Buffer busy waits' (Base Wait Time: 47%, Compare Wait Time: 51%) and another for 'Compare Wait Time' (Base Wait Time: 20%, Compare Wait Time: 52%). Below these are tables for 'Event Description' and 'Event'.

Event Name	Base	Compare	Total	Wait (Count)	Average Wait Time (ms)
buffer busy waits	615,916	3,320,381		2,704,440	592,456
do file acquire read	395,739	1,240,434		877,143	946,4
do file acquire write	133,239	1,045,655		1,045,655	1,222,4
do file agent IO submit	105,909	375,602		175,604	25,707
emp TX: contention	1,799	155,592		190	294
emp CR: block range resolve opt	22,855	125,709		108,854	135
emp TX: new lock contention	378	16,062		15,637	293
reconnect: reply	1	1,1446		1,1446	29
- Basic Info**: Describes four reasons for a session failing to pin a buffer in the buffer cache due to other sessions reading it or holding locks.
- Event Detail**: Shows detailed information for 'Buffer busy waits' including system calls like 'ALTER system set dispatcher', 'ARCH Remote Write', 'ASYNC Remote Write', etc., and specific events like 'batched allocate scn lock request', 'BFILE check if exists', 'BFILE check if open', 'BFILE closure', 'BFILE get length', 'BFILE get name object', 'BFILE get path object', 'BFILE internal seek', 'BFILE open', 'BFILE read', 'Broadcast mng queue transition', 'Broadcast mng recovery queue transition', 'buffer busy global cache', 'buffer busy global cr', and 'Buffer busy waits'.
- Parameter & Wait Time**: Lists parameters for 'buffer busy waits' and notes that the wait time is 1 second.
- Wait Time**: Notes that the normal wait time is 1 second.

✓ Event Description 기능을 통해 해당 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

STEP

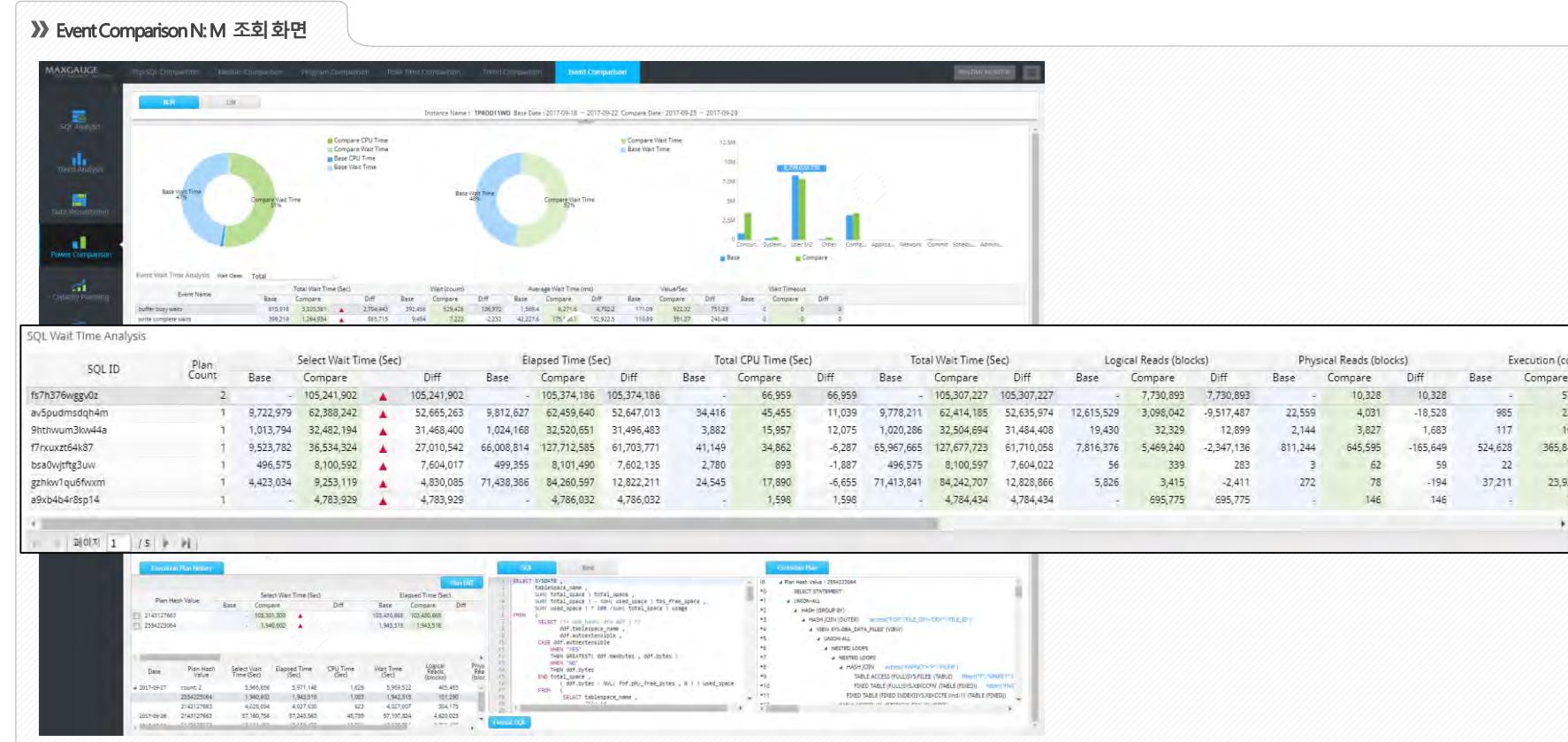
1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- Wait Class 단위 분석
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인

■ 대기 이벤트 발생 SQL 추적

- SQL 상세 정보 확인



- ✓ SQL Wait Time Analysis에서 buffer busy waits로 대기했던 SQL List를 확인할 수 있습니다.
- ✓ Select Wait Time, Elapsed Time, Total CPU Time, Total Wait Time, Logical Reads, Physical Reads, Execution count 등의 정보를 제공합니다.
- ✓ Base 구간에 Wait Time이 “-”인 SQL은 Compare Date 기간에 추가된 SQL입니다.
- ✓ “SQL ID : fs7h376wggv0z”이 대부분의 buffer busy waits 대기 현상을 유발하며, 이 SQL로 인해서 기존에 수행되던 SQL들의 buffer busy waits 대기 시간도 증가 했음을 알 수 있습니다.

“전 주에 비해 늘어난 대기이벤트와 그 원인을 알고 싶어요 (N:M 분석)

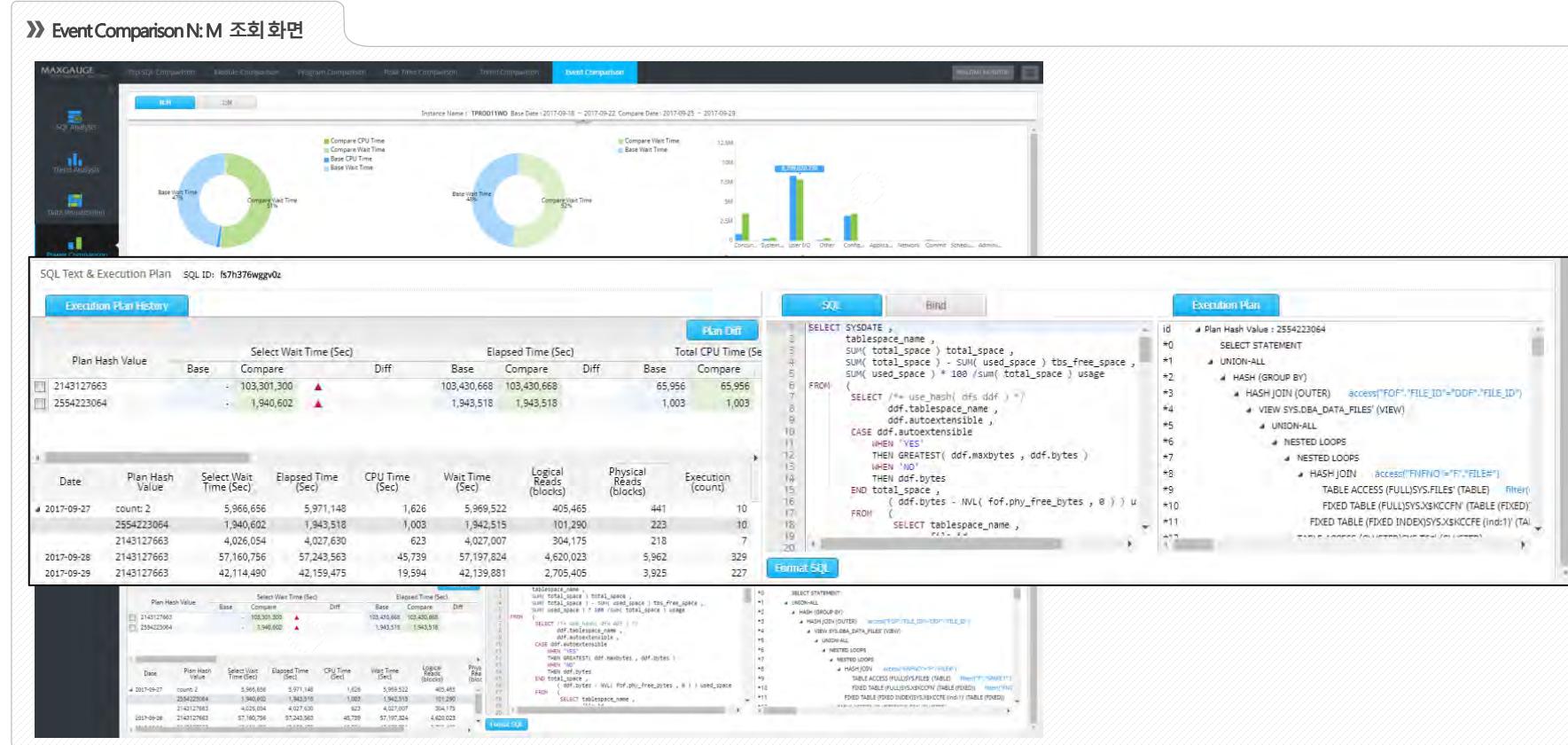
STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
 - Total Wait Time 비교
 - Wait Class 단위 분석
 - 발생 대기 이벤트 확인
 - 대기 이벤트 정보 확인
 - 대기 이벤트 발생 SQL 추적
 - **SQL 상세 정보 확인**

SQL 상세 정보 확인



- ✓ SQL Text & Execution Plan에서는 선택 된 SQL의 Execution Plan History, SQL_ID의 SQL Full Text, Bind 변수, 실행계획을 제공합니다.
 - ✓ Execution Plan History에서 Plan Hash Value단위, 일자 별 SQL Stat 정보를 제공합니다.
 - ✓ Buffer busy waits을 유발하는 위 SQL의 1회 수행 Logical Reads는 약 13,000(blocks)이며, Elapsed Time이 18만 초인 것으로 보아 악성 SQL일 확률이 높습니다.
 - ✓ Compare Date에 수행된 SQL(fs7h376wqqv0z)의 성능 점검 및 개선이 필요하다는 것을 확인할 수 있습니다.

실전 분석 사례 Case 2.
**신규 추가 된 업무가 발생시키는
대기 현상이 궁금해요 (1:M 분석)**

“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인

» Event Comparison 검색 화면

- 1 비교 대상 Instance (TPROD11WO)를 선택합니다.
- 2 Schema (**MAXGAUGE**), Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 정렬 기준을 선택하고, 출력되는 Top Event의 개수(TOP 30)를 설정합니다. 정렬 기준은 Total Wait Time과 Value / Sec이 있습니다.
- 4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (00:00 ~ 24:00) 을 선택적으로 선택 할 수 있습니다.
- 5 기준 날짜(2017.10.2)와 비교 날짜(2017.10.11~12) 모두 Working day, Week, 특정 날짜로 선택 가능합니다.
- 6 위의 설정한 조건으로 Event Comparison 실행합니다.

✓ 시나리오

INSTANCE : TPROD11WO
 SCHEMA : MAXGAUGE
 신규 업무 반영 일자 : 2017년 10월 11일 ~
 업무 집중 시간 : 00:00 ~ 24:00

✓ 목표

신규 업무 반영 후 시스템에 영향을 주는 대기 이벤트를 확인하고, SQL 추출 및 성능 저하 원인 파악.

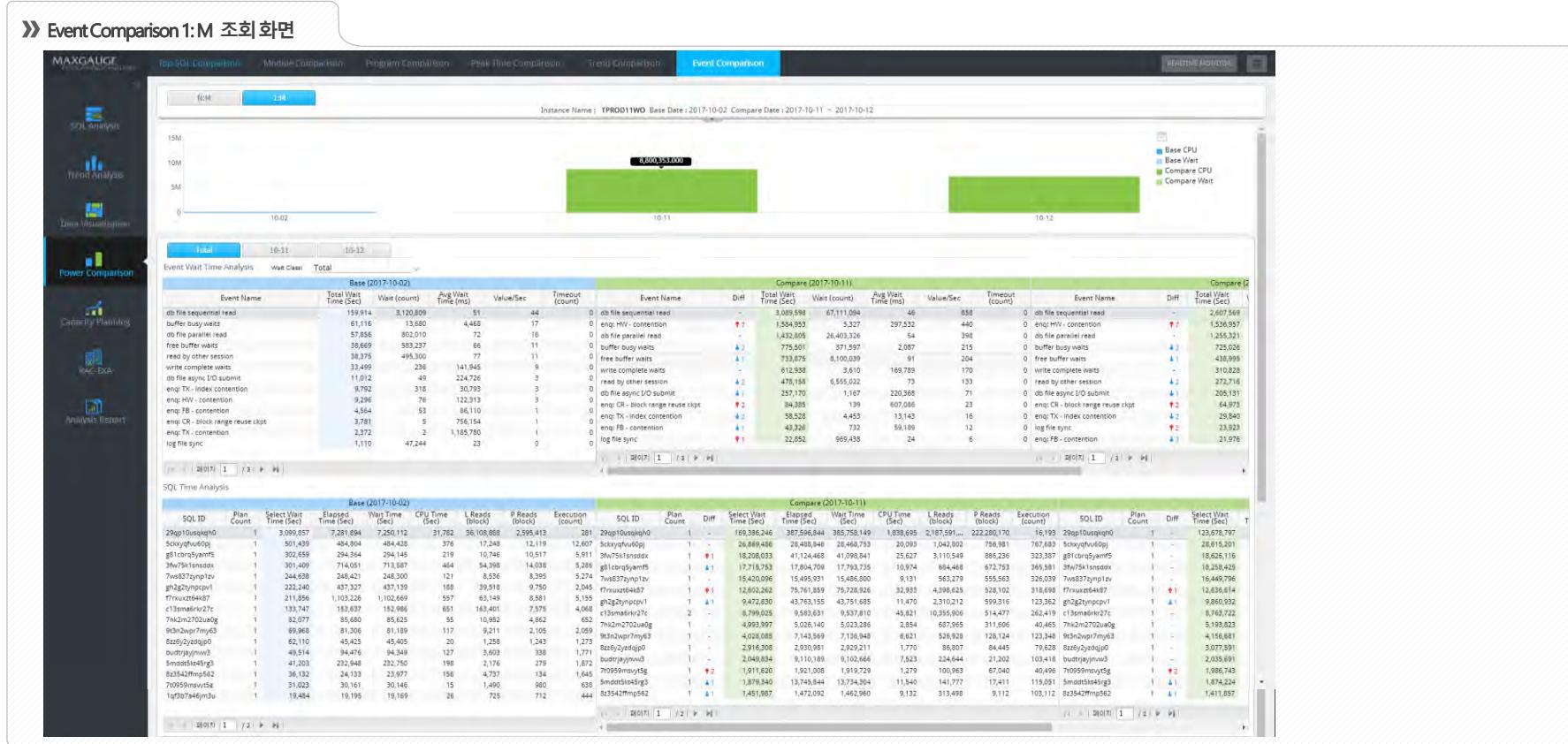
“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
 - Total Wait Time 비교
 - 발생 대기 이벤트 확인
 - 대기 이벤트 정보 확인
 - 대기 이벤트 발생 SQL 추적
 - SQL 상세 정보 확인



- ✓ 상단 그래프를 통하여 Base Wait Time과 Compare Wait Time 값을 일자 별로 비교 분석합니다.
 - ✓ Event Wait Time Analysis에서 비교 기간의 Diff 컬럼이 “**NEW**, ↑”로 표시된 이벤트가 있는지 확인합니다.
 - ✓ “**NEW**, ↑”로 표시된 이벤트는 Wait Time이 증가해서 대기 이벤트 순위가 상승한 이벤트입니다.
 - ✓ Event Wait Time Analysis에서 선택한 Event를 발생 시킨 SQL은 SQL Wait Time Analysis에서 확인할 수 있습니다.
 - ✓ SQL Wait Time Analysis에서 “**NEW**”로 표시된 SQL은 신규 업무 추가로 인해 Wait Time이 증가한 SQL입니다.

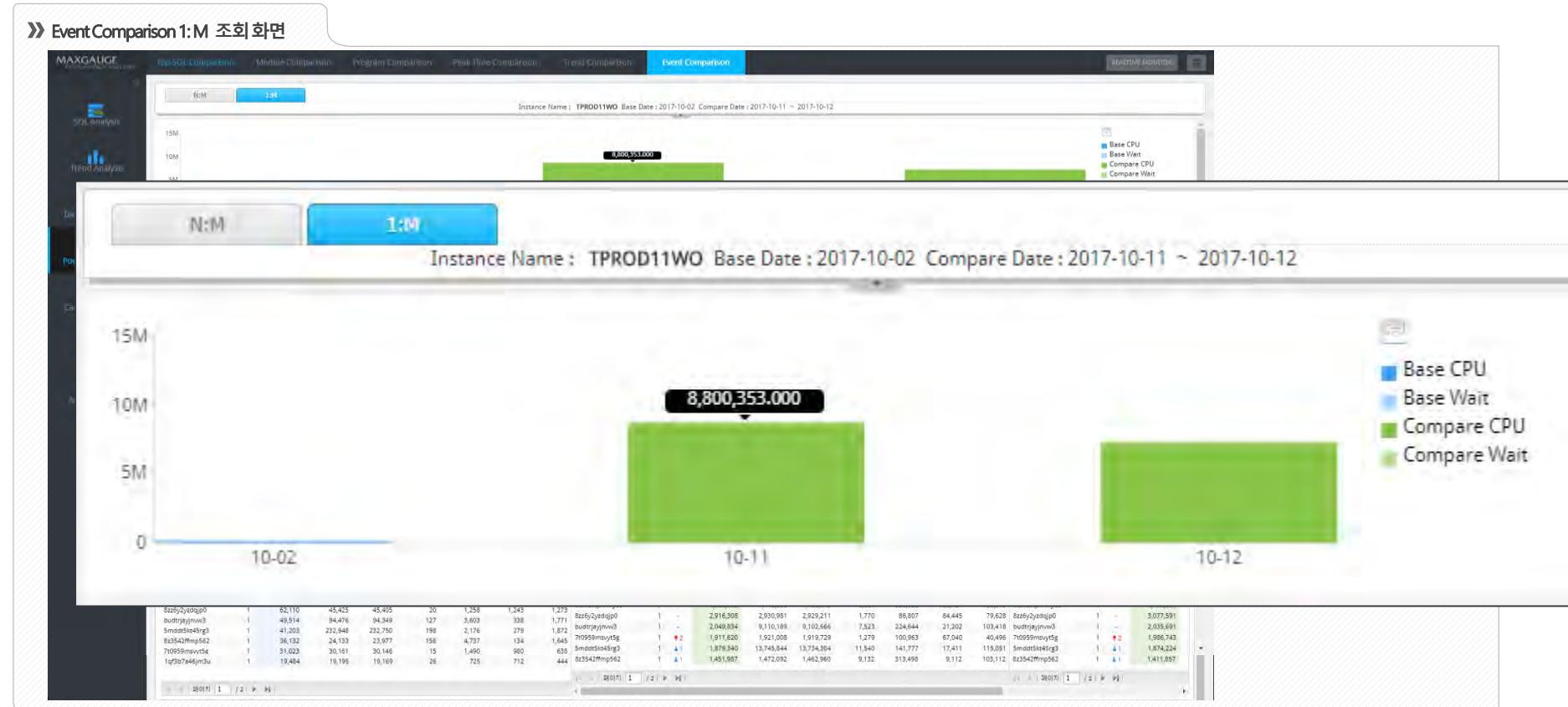
“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인



- ✓ 상단 그래프를 통하여 Base Wait Time과 Compare Wait Time 값을 일자 별로 비교 분석합니다.
- ✓ 좌측의 파란색은 base Wait Time을, 우측의 초록색은 일자 별 Compare Wait Time을 나타냅니다.
- ✓ Compare Wait Time이 Base Wait Time에 비해 수치가 높으며, 신규 업무 추가로 Wait Time이 증가했음을 유추할 수 있습니다.

Event Comparison의 활용

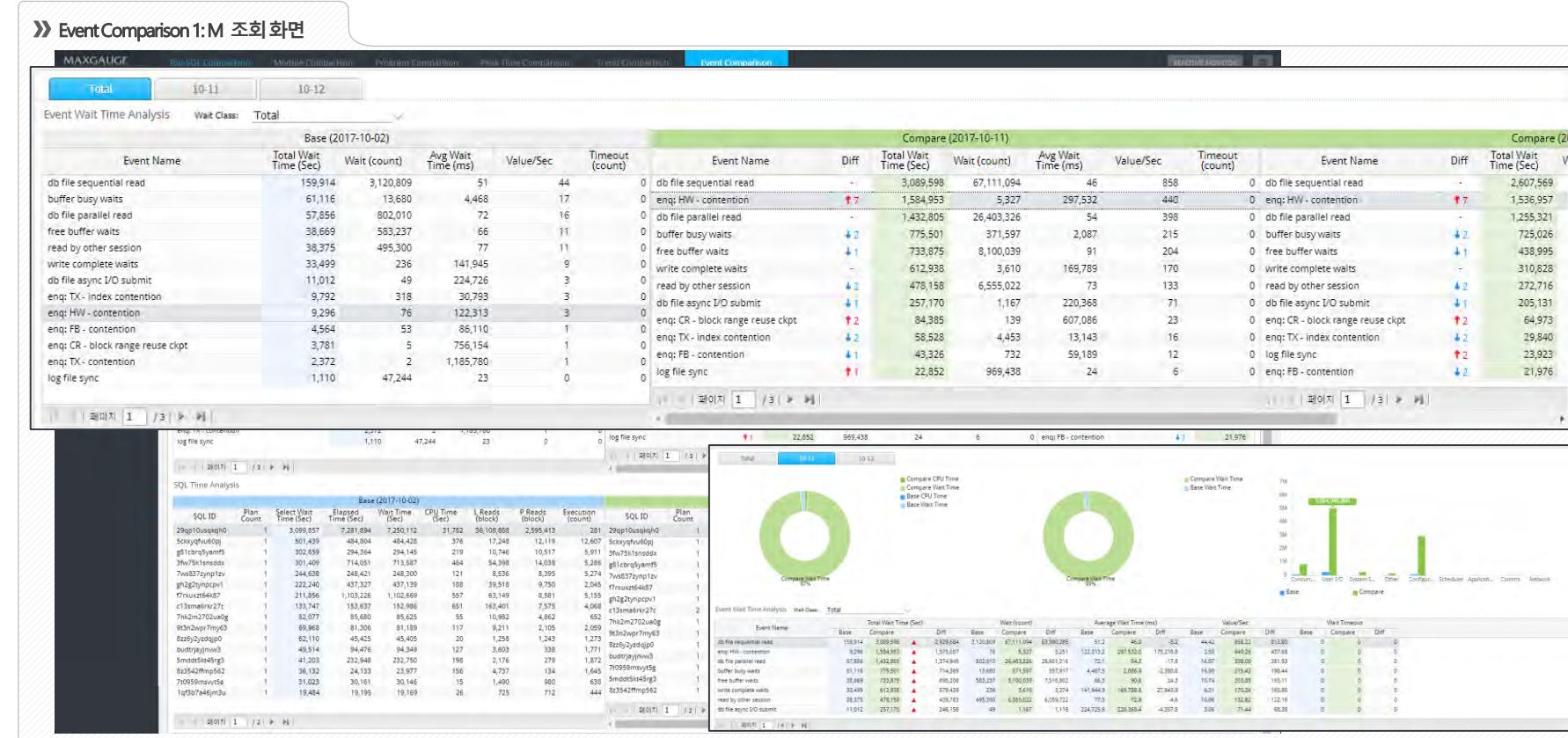
"신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)"

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인



- ✓ Event Wait Time Analysis에서 Total과 Compare Date 탭으로 분류되며, 최초에는 Total로 출력됩니다.
- ✓ Total 탭은 좌측의 Base Date와 우측에 Compare Date의 데이터가 모두 표시됩니다.
- ✓ Base Date와 Compare Date는 Total Wait Time이 높은 이벤트 순으로 데이터가 출력되며, 이벤트 순위의 변동은 Diff 컬럼에 "↑, ↓, NEW"로 표현됩니다.
- ✓ Compare Date(10-11)탭은 앞서 설명한 N:M 분석과 동일한 화면을 출력합니다.
- ✓ Enq: HW-contention의 이벤트 순위가 ↑7로 높아진 것으로 확인되며, 신규로 추가된 업무로 인해 Wait Time이 증가한 이벤트 중 하나인 것으로 확인됩니다.

“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인**
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인

» Event Comparison 1:M 조회 화면

Event Description

enq: HW - contention

enq: HW - contention 대기 이벤트

Basic Info
DML이 수행되는 종안 DML과 관련되 각제에 대기 연결을 방지하기 위해 DML을 수행하는 프로세스는 반드시 해당 테이블에 대해 TM 락을 획득해야 한다. TM 락을 획득하는 과정에서 경합이 발생하면 enq: TM - contention 이벤트를 대기하게 된다.

Parameter & Wait Time
Wait Parameters
enq: HW - contention 대기이벤트의 대기 파라미터는 다음과 같다.
P1 = Enqueue 정보
P2 = Tablespace#
P3 = Segment Header Block#

Wait Time
enqueue 대기이벤트와 동일하다. 최대 3초까지 기다린다. 단일 HW 락을 획득하기 못하면 획득할 때까지 대기한다.

Event Name	Total Time (Sec)	Total Wait (Sec)	Avg Wait (Sec)	Value/Sec	Timeout (count)
db file sequential read	159,914	3,120,809	51	44	0 db file seq...
buffer busy waits	61,116	13,680	4,465	17	0 eng:HW - c...
db file parallel read	57,856	802,010	72	16	0 db file par...
free buffer waits	38,669	583,237	66	11	0 buffer busy...
read by other session	38,375	499,300	77	11	0 free buffer...
writer process waits	32,434	2,259	141,925	9	0 writer process...
db file sync I/O submit	11,012	49	224,726	3	0 db file sync...
eng: TX - contention	9,792	318	30,793	3	0 read by othe...
eng: HW - contention	9,296	78	122,313	3	0 eng:HW - c...
eng: PB - contention	4,564	53	86,110	1	0 eng:PB - c...
eng: CR - block range reuse skip	3,781	5	756,154	1	0 eng:CR - b...
eng: TX - contention	2,372	2	1,185,780	1	0 eng:TX - c...
log file sync	1,110	47,244	23	0	0 log file sync...

SQL ID	Plan Count	Select_Wait (Sec)	Elapsed (Sec)	WaitTime (Sec)	CPU Time (Sec)	I_Ready (blocks)	P Reads (blocks)	Execution (count)	SQL Text
29ep10usqeh0	1	3,099,857	7,281,694	7,250,112	31,762	36,108,888	2,595,413	281	29ep10usqeh0
5cyygh4u0dpj0	1	501,439	484,864	484,428	376	17,248	12,119	12,607	5cyygh4u0dpj0
g1t1orqz9mpm0	1	259,155	267,455	267,455	216	10,746	10,317	10,317	g1t1orqz9mpm0
3hv537ynp0t0	1	301,469	714,051	713,597	464	54,398	14,038	8,936	3hv537ynp0t0
7hv837ynp0t0	1	244,638	245,421	248,300	121	8,536	8,395	5,574	7hv837ynp0t0
g1z2y0mp0p0v0	1	222,340	437,327	437,139	168	39,918	9,730	2,045	g1z2y0mp0p0v0
7rvuxct648487	1	211,856	1,103,226	1,102,669	557	63,149	8,581	5,155	g1z2y0mp0p0v0
c13mmekrkt2t	1	133,747	153,637	152,966	651	163,401	7,375	4,066	c13mmekrkt2t
7nkm2m702u0g	1	82,077	95,680	85,625	55	10,952	4,862	655	7nkm2m702u0g
7nkm2m702u0g	1	82,077	95,680	85,625	55	10,952	4,862	655	7nkm2m702u0g
9d3k2wq7my03	1	69,946	81,306	81,189	117	9,211	2,105	2,059	9d3k2wq7my03
9d3k2wq7my03	1	69,946	81,306	81,189	117	9,211	2,105	2,059	9d3k2wq7my03
9d3k2wq7my03	1	40,514	84,476	84,349	127	3,603	338	1,771	9d3k2wq7my03
5m0t5h4z4rg3	1	41,203	232,948	232,750	196	2,176	279	1,872	5m0t5h4z4rg3
8t3342fmnp562	1	36,132	24,133	23,977	156	4,737	134	1,645	8t3342fmnp562
70959rnsv154	1	31,023	30,161	30,146	15	1,490	980	633	5m0t5h4z4rg3
1qfb07a46m3u	1	19,484	19,195	19,169	26	723	712	444	8t3342fmnp562

✓ Event Description 기능을 통해 해당 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

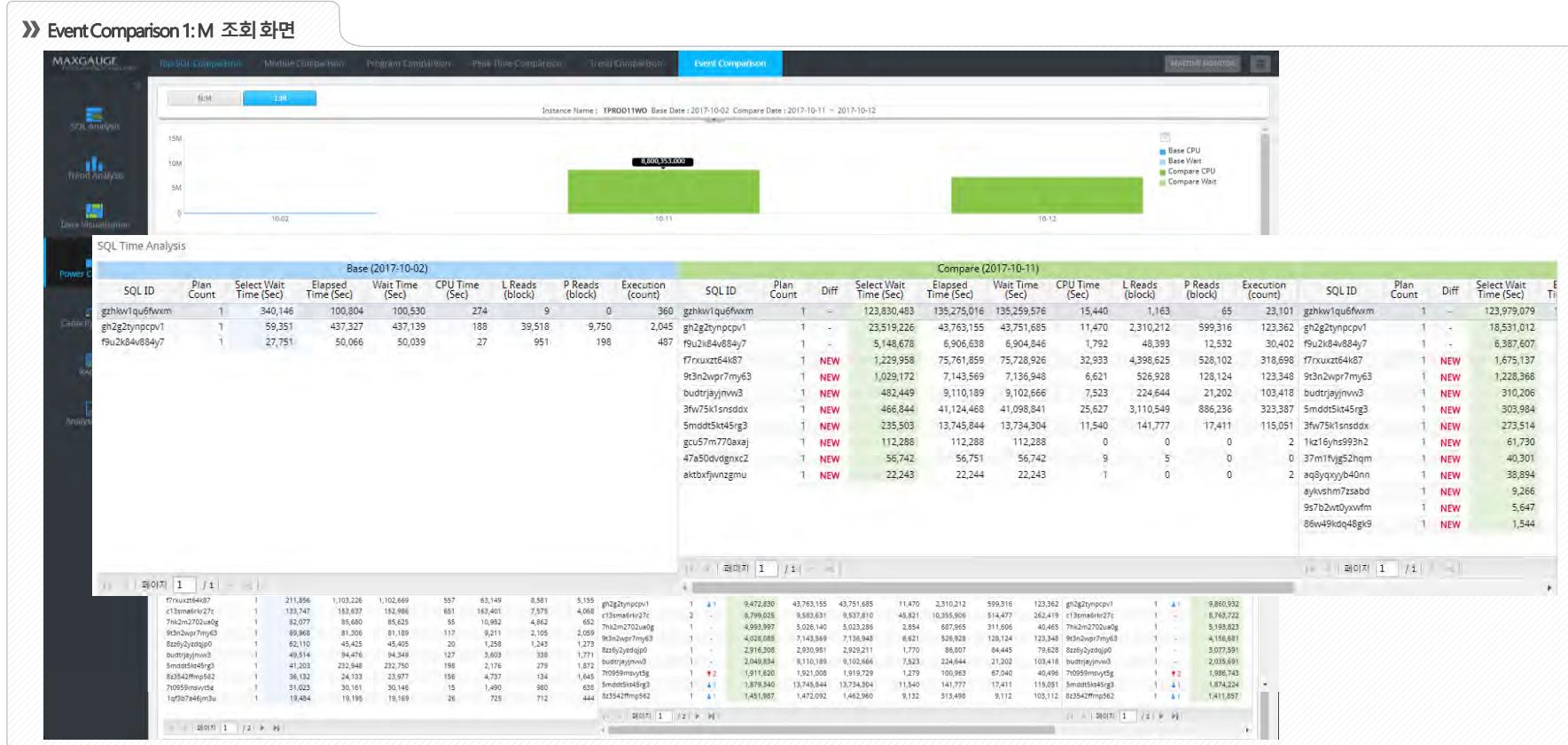
“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적**
- SQL 상세 정보 확인



- ✓ SQL Wait Time Analysis에서 enq: HW-contention으로 대기했던 SQL List를 일 단위로 확인할 수 있습니다.
- ✓ Select Wait Time, Elapsed Time, Total CPU Time, Total Wait Time, Logical Reads, Physical Reads, Execution count등의 정보를 제공합니다.
- ✓ Compare구간에 Diff가 “**NEW**”인 SQL은 신규로 추가된 업무의 SQL입니다.
- ✓ 신규로 추가된 업무의 SQL로 인해 enq: HW-contention 대기가 증가했음을 알 수 있습니다.

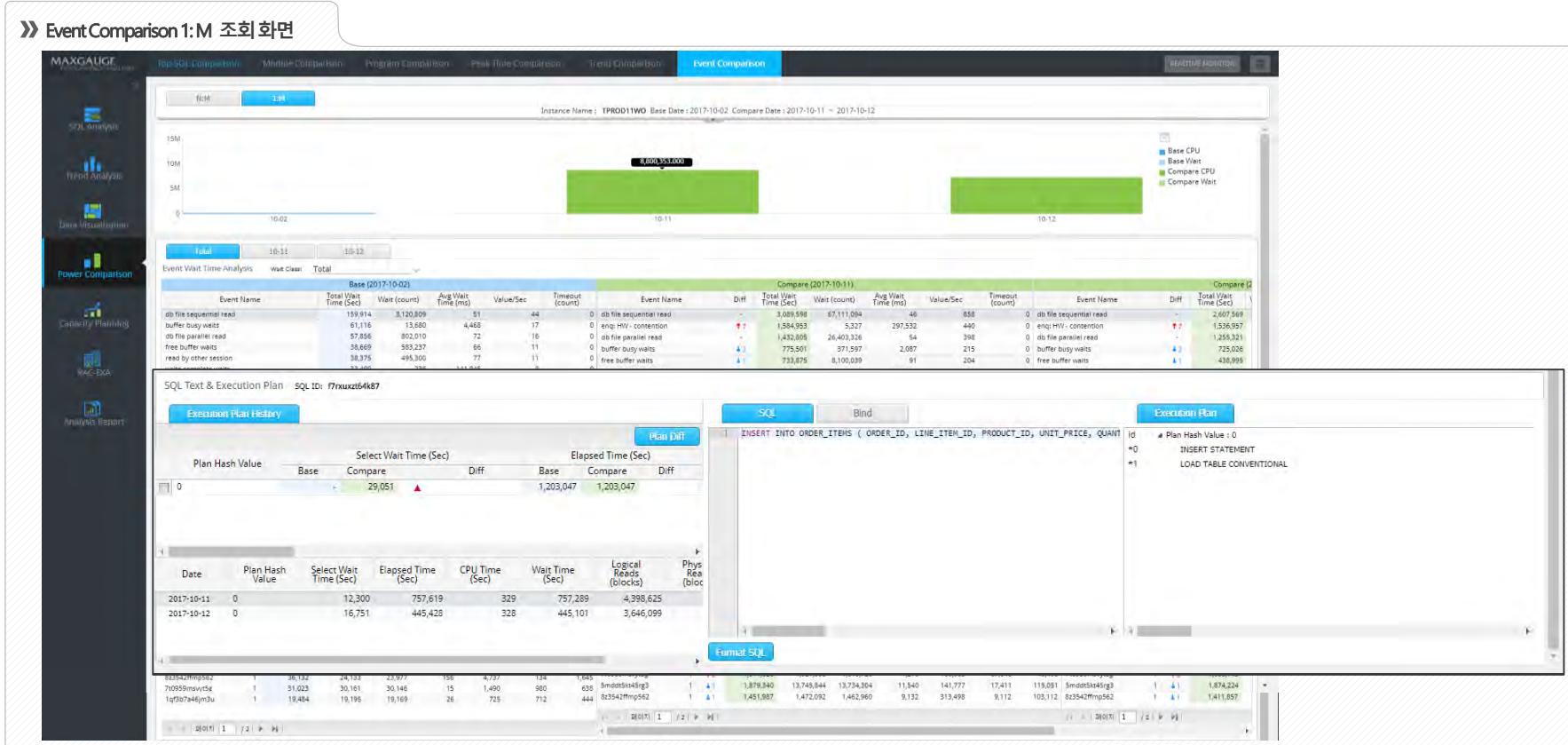
“신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Total Wait Time 비교
- 발생 대기 이벤트 확인
- 대기 이벤트 정보 확인
- 대기 이벤트 발생 SQL 추적
- SQL 상세 정보 확인



- ✓ SQL Text & Execution Plan에서는 선택 된 SQL의 Execution Plan History, SQL_ID의 SQL Full Text, Bind 변수, 실행계획을 제공합니다.
- ✓ Execution Plan History에서 Plan Hash Value 단위, 일자 별 SQL Stat 정보를 제공합니다.
- ✓ enq: HW-contention 을 유발하는 SQL들은 대부분 INSERT문이고, Execution Count가 하루에 20만~60만으로 매우 높습니다.
- ✓ 신규 업무로 추가된 SQL(INSERT문)의 성능을 점검 및 대기를 줄이기 위한 방안이 필요함을 확인할 수 있습니다.

MAXGAUGE



Practical Guide

Top-SQL Comparison [Performance Analyzer]

Contents

Top-SQL Comparison?

Top-SQL Comparison의 활용

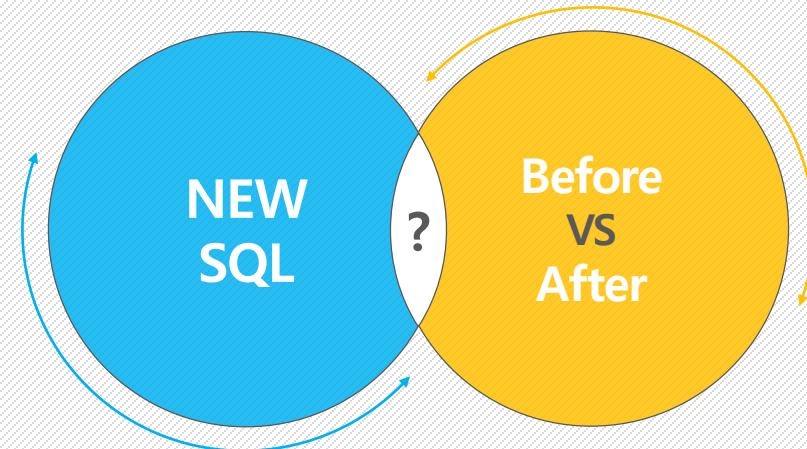
Case 1. 신규로 추가 된 업무 성능이 궁금해요

Case 2. 지난 달과 이번 달의 월 단위 작업을 비교하고 싶어요

Top-SQL Comparison?

Top SQL Comparison은 언제 쓰나요?

신규로 추가된 업무가
성능에 미치는 영향이 궁금할 때



» 지난 달과 이번 달의
월 말 결산 작업을 비교하고 싶을 때

Top-SQL Comparison?

Top-SQL Comparison

- 기준 날짜 대비 비교 날짜들의 Top SQL간 비교 기능이다.
- Base Time과 Compare Time 두 구간에 대한 1:M, N:M 비교 분석을 지원한다.
- Base Time 기준으로 신규 SQL은 “**NEW**”, 기존 SQL의 순위 변동에 대해서는 RANK “↑”, “↓” 형태의 표현 방식을 지원한다.
- 해당 기능은 Performance Analyzer ▶ Power Comparison ▶ Top-SQL Comparison 경로를 통해 사용할 수 있다.

» 1:M Comparison

Elapsed Time				CPU Time		Logical Reads		Physical Reads		Executions											
RANK	2017-08-09	Elapsed Time (%)	Value	RANK	2017-08-14	Diff	Elapsed Time (%)	Value	RANK	2017-08-15	Diff	Elapsed Time (%)	Value	RANK	2017-08-16	Diff	Elapsed Time (%)	Value	RANK	2017-08-17	Diff
1	f711myt0q6cma	25.3%	6	1	av5pudmsdqh...	NEW	72.0%	382	1	av5pudmsdqh...	NEW	84.2%	344	1	av5pudmsdqh...	NEW	91.9%	454	1	av5pudmsdqh...	NEW
2	459f3z9u4fb3u	25.1%	6	2	424h0nf7bhqzd	NEW	5.5%	29	2	424h0nf7bhqzd	NEW	3.1%	13	2	459f3z9u4fb3u	-	2.1%	10	2	f711myt0q6cma	↓1
3	8guhtcknnkk660	10.3%	2	3	459f3z9u4fb3u	↓1	2.0%	11	3	f711myt0q6cma	↓2	3.0%	12	3	459f3z9u4fb3u	↓1	2.0%	10	3	459f3z9u4fb3u	↓1
4	4vs91dcv7u1p6	10.1%	2	4	f711myt0q6cma	↓3	1.9%	10	4	459f3z9u4fb3u	↓2	2.8%	12	4	4vs91dcv7u1p6	-	1.1%	5	4	b6usrig82hwsa3	NEW
5	cm5vu20fhtnq1	4.2%	1	5	gs3g1d5621h60	NEW	1.7%	9	5	g5m0bnvyy37b1	NEW	1.5%	6	5	8guhtcknnkk660	↓2	0.9%	4	5	424h0nf7bhqzd	NEW
6	2h0gb24h6zpnpu	2.7%	1	6	b6usrg82hwsa3	NEW	1.6%	9	6	4vs91dcv7u1p6	↓2	1.4%	6	6	cm5vu20fhtnq1	↓1	0.5%	2	6	4vs91dcv7u1p6	↓2
7	8swypbbr0m372	2.7%	1	7	4vs91dcv7u1p6	↓3	1.1%	6	7	8guhtcknnkk660	↓4	0.9%	4	7	ok8522rmdzg4k	NEW	0.3%	2	7	g5m0bnvyy37b1	NEW
8	b3abwkm67yg8r	2.5%	1	8	884mcmv0d7u6z	NEW	0.8%	4	8	cm5vu20fhtnq1	↓3	0.5%	2	8	4u8m4fn1apxpt	NEW	0.2%	1	8	8guhtcknnkk660	↓5
9	5z8zq6mk1gtsz	2.3%	1	9	0x57a7fkzngvr	NEW	0.7%	4	9	6j3rn680twb13	NEW	0.3%	1	9	b3abwkm67yg8r	↓1	0.2%	1	9	9lxy3qcgmruhh	NEW
10	a9xb4b4r8sp14	2.1%	1	10	8guhtcknnkk660	↓7	0.7%	4	10	0k8522rmdzg4k	NEW	0.2%	1	10	4yyb4104skrwj	NEW	0.1%	1	10	dffy0786aaah6kx	NEW

» N:M Comparison

Elapsed Time				CPU Time		Logical Reads		Physical Reads		Executions								
RANK	2017-08-14 ~ 2017-08-18	Elapsed Time (%)	Value	RANK	2017-08-14 ~ 2017-08-18	Elapsed Time (%)	Value	RANK	2017-08-07 ~ 2017-08-11	Diff	Elapsed Time (%)	Value	RANK	2017-08-07 ~ 2017-08-11	Diff	Elapsed Time (%)	Value	
1	av5pudmsdqh...	82.3%	1,755	11	0k8522rmdzg4k	-	0.3%	6	1	av5pudmsdqh...	-	35.7%	76	11	6j3rn680twb13	NEW	1.4%	3
2	f711myt0q6cma	2.4%	52	12	63kf3an2j0pkc	-	0.3%	6	2	459f3z9u4fb3u	↑2	15.9%	34	12	0k8522rmdzg4k	↓1	1.0%	2
3	424h0nf7bhqzd	2.2%	48	13	9lxy3qcgmruhh	-	0.3%	5	3	f711myt0q6cma	↓1	15.1%	32	13	8swypbbr0m372	NEW	0.9%	2
4	459f3z9u4fb3u	2.2%	46	14	g9m7794s0k622	-	0.2%	5	4	4vs91dcv7u1p6	↑1	5.9%	13	14	2h0gb24h6zpnpu	NEW	0.9%	2
5	4vs91dcv7u1p6	1.2%	26	15	884mcmv0d7u6z	-	0.2%	4	5	8guhtcknnkk660	↑1	5.8%	12	15	4u8m4fn1apxpt	NEW	0.8%	2
6	8guhtcknnkk660	0.8%	17	16	0x57a7fkzngvr	-	0.2%	4	6	cm5vu20fhtnq1	↑2	2.2%	5	16	anyj2gn8hcd5y	NEW	0.8%	2
7	b6usrg82hwsa3	0.7%	15	17	f260gh1ks21rz	-	0.2%	3	7	53c2k4c43zcfx	NEW	2.2%	5	17	0ws7ahf1d78qa	NEW	0.6%	1
8	cm5vu20fhtnq1	0.5%	11	18	b3abwkm67yg8r	-	0.1%	3	8	b3abwkm67yg8r	↑10	1.8%	4	18	bz6h5qv8kc02q	NEW	0.6%	1
9	g5m0bnvyy37b1	0.5%	11	19	ffrxyzt4415k	-	0.1%	3	9	4yyb4104skrwj	NEW	1.5%	3	19	27k1sg83rqtcp	NEW	0.6%	1
10	gs3g1d5621h60	0.4%	9	20	d0xmnp08rhfbg	-	0.1%	3	10	a9xb4b4r8sp14	NEW	1.5%	3	20	9s0xa5dgvuq55	NEW	0.5%	1

Top-SQL의 비교를 더욱 쉽게 할 수 있습니다!

MaxGauge

Performance Analyzer → Top-SQL Comparison

튜닝 대상 SQL 선별

SQL
분석 방법

AWR

날짜 별
AWR Report 추출

SQL 성능 이력 추출

AWR 자료 수집

Top-SQL 확인

SQL 실행계획 및 Bind
수집

SQL
Tuning

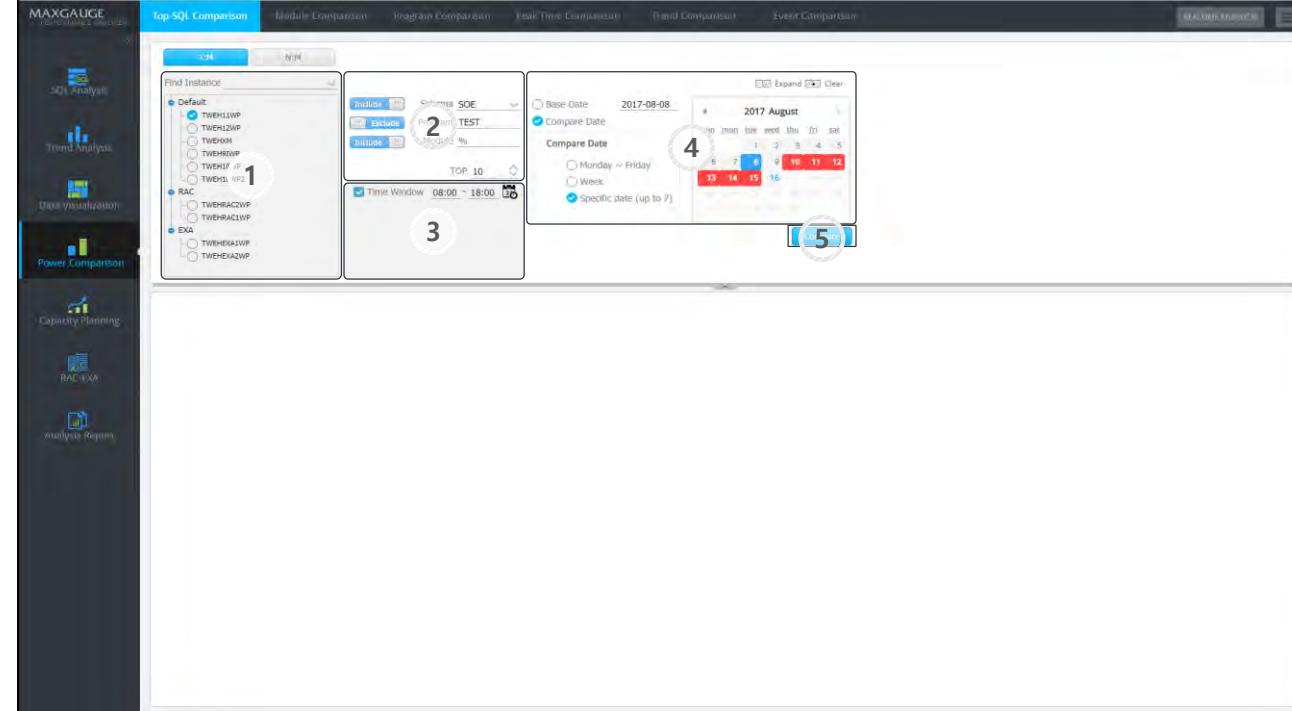
Top-SQL Comparison의 활용

Top-SQL Comparison의 사용 방법 (1:M)

Top-SQL Comparison (1:M)

- 검색 조건 설정
- 데이터 분석

» Top-SQL Comparison 1:M 검색화면



- 1 비교 대상 Instance를 선택합니다.
- 2 Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- 4 기준 날짜(Base Date)는 하루만 선택되며, 비교 날짜(Compare Date)는 Working day, Week, 특정 날짜로 선택 가능합니다.
- 5 위의 설정한 조건으로 Top-SQL Comparison를 실행합니다.

Top-SQL Comparison의 사용 방법 (1:M)

Top-SQL Comparison (1:M)

- 검색 조건 설정
- 데이터 분석

» Top-SQL Comparison 1:M 조회화면

The screenshot displays the Top-SQL Comparison interface with the following sections:

- Top SQL Comparison:** Shows four tables comparing SQL performance metrics across four dates (2017-07-28, 2017-08-08, 2017-08-09, 2017-08-10). The first table lists top SQLs by logical reads.
- Performance Metrics:** A chart showing Elapsed Time (Sec) over time, with bars for each day from 2017-08-08 to 2017-08-11.
- Timeline:** A timeline bar at the bottom indicating the duration of the comparison period.
- Execution Plan:** A detailed view of the execution plan for a selected SQL query, showing the plan hash, bind variables, and the execution plan steps.

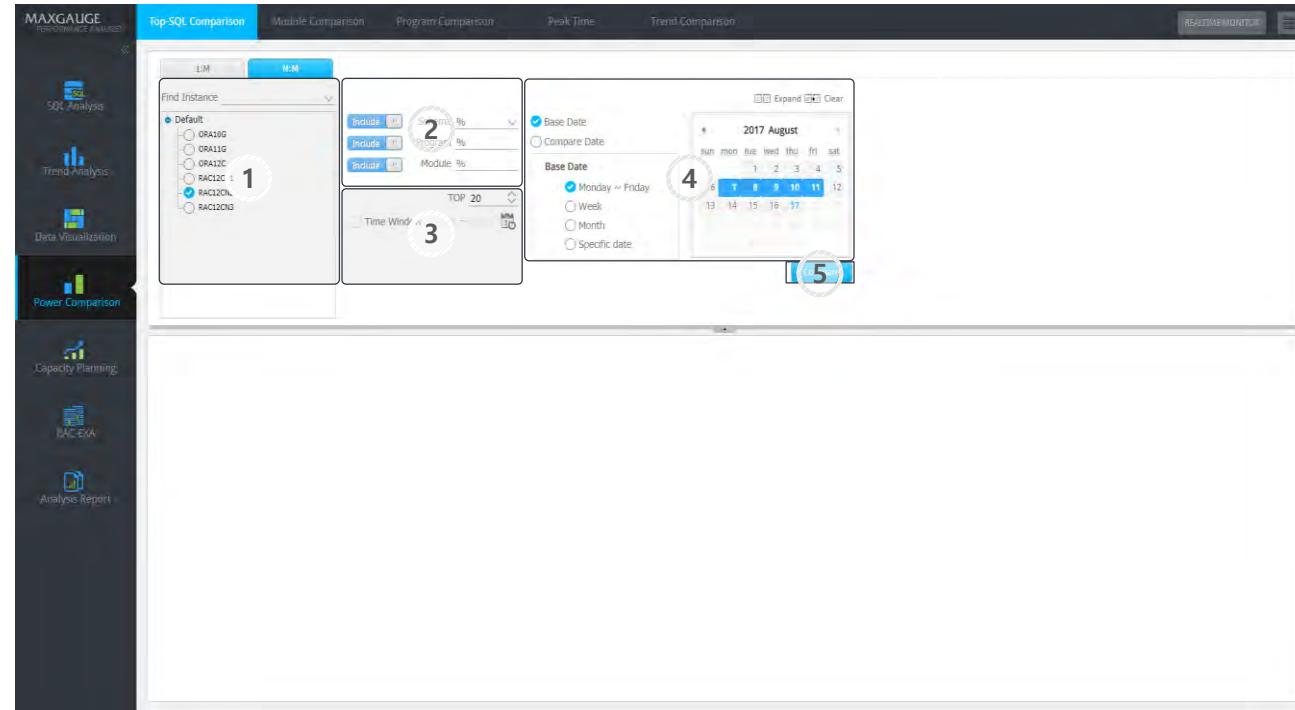
- TOP SQL 추출 기준은 Elapsed Time, CPU Time, Logical Reads, Physical Reads, Executions 으로 분류 됩니다.
- 신규 업무 추가 전 기준 날짜 (Base Date) 기준 TOP SQL
- 신규 업무 추가 후, 각 비교 날짜 (Compare Date) 기준으로 추출된 TOP SQL
- Compare Date별 수행 정보를 통한 SQL 성능 및 SUM / AVG 정보 확인
- 선택한 SQL에 대한 SQL Text 및 Bind, Execution Plan 정보 제공

Top-SQL Comparison의 사용 방법 (N:M)

Top-SQL Comparison (N:M)

- 검색 조건 설정
- 데이터 분석

» Top-SQL Comparison N:M 검색 화면



- 1 비교 대상 Instance를 선택합니다.
- 2 Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- 4 기준 날짜(Base Date)와 비교 날짜(Compare Data) 모두 Working day, Week, Month, 특정 날짜로 선택 가능합니다.
- 5 위의 설정한 조건으로 Top-SQL Comparison 실행합니다.

Top-SQL Comparison의 사용 방법 (N:M)

Top-SQL Comparison (N:M)

- 검색 조건 설정
- 데이터 분석

» Top-SQL Comparison N:M 조회화면

The screenshot displays the Top-SQL Comparison interface for comparing N:M operations. It includes:

- Performance Metrics:** A grid showing top SQL statements by Elapsed Time (Base) and Compare Date.
- Execution Details:** A table showing detailed execution statistics for a specific SQL ID over time.
- Elapsed Time (Sec):** A chart showing the average elapsed time across different dates.
- SQL Text and Execution Plan:** A panel showing the selected SQL statement, its bind variables, and the execution plan with detailed information about the access path and predicates.

- TOP SQL 추출 기준은 Elapsed Time, CPU Time, Logical Reads, Physical Reads, Executions 으로 분류 됩니다.
- 월 단위 업무의 기준 날짜 (**Base Date**)의 TOP SQL 정보입니다.
- 월 단위 업무의 비교 날짜 (**Compare Date**) 기준으로 추출된 TOP SQL 정보입니다.
- 비교 날짜(**Compare Date**) 별 수행 정보를 통한 SQL 성능 및 SUM / AVG 정보를 확인할 수 있습니다.
- 선택한 SQL에 대한 SQL Text 및 Bind, Execution Plan 정보를 제공합니다.

실전 분석 사례 Case 1.
**신규로 추가 된 업무 성능이
궁금해요**

“ 신규로 추가 된 업무 성능이 궁금해요! ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 추출 기준 선택
- “NEW” SQL 성능 확인
- “NEW” SQL 수행 추이 확인
 - └ SQL Analysis ▶ Long-Term Analysis 연계
- “NEW” SQL 상세 정보 확인
 - └ SQL Detail 연계
- 관련 대기 이벤트 정보 확인
 - └ Event Description 연계

» Top-SQL Comparison 검색 화면

The screenshot shows the Top-SQL Comparison search interface. It includes a sidebar with various analysis modules like SQL Analysis, Trend Analysis, Data Visualization, Power Comparison, Capacity Planning, RAC, and Multiple Reporting. The main area has tabs for Module Comparison, Program Comparison, Peak Time Comparison, Trend Comparison, and Event Comparison. The search panel on the right allows selecting instances (1), schemas (2), programs (3), time windows (4), and specific dates (5) for comparison.

- 1 비교 대상 Instance (TWEH11WP)를 선택합니다.
- 2 Schema (MAXGAUGE), Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (09:00 ~ 18:00) 을 선택적으로 선택 할 수 있습니다.
- 4 기준 날짜 (2017.07.28)는 하루만 선택되며, 비교 날짜는 Working day, Week, 특정 날짜 (2017.08.08 ~ 2017.08.11)로 선택 가능합니다.
- 5 위의 설정한 조건으로 Top-SQL Comparison 실행합니다.

✓ 시나리오

INSTANCE	: TWEH11WP
SCHEMA	: SOE
PROGRAM	: TEST
신규 업무 반영 일자	: 2017년 8월 10일 ~
업무 집중 시간	: 09:00 ~ 18:00

✓ 목표

신규 업무 반영 후 시스템에 영향을 주는 SQL
추출 및 성능 저하 원인 파악.

Top-SQL Comparison의 활용

"신규로 추가 된 업무 성능이 궁금해요!"

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 추출 기준 선택

- "NEW" SQL 성능 확인
- "NEW" SQL 수행 추이 확인
 - SQL Analysis ▶ Long-Term Analysis 연계
- "NEW" SQL 상세 정보 확인
 - SQL Detail 연계
- 관련 대기 이벤트 정보 확인
 - Event Description 연계

» Top-SQL Comparison 1:M 조회 화면

RANK	2017-07-28	Logical Reads	Value	RANK	2017-08-08	Logical Reads	Value	RANK	2017-08-09	Logical Reads	Value	RANK	2017-08-10	Logical Reads	Value	RANK	2017-08-11	
1	bz6h5wv8k02q	49.4%	24,425	1	av5pudmsdqh...	NEW	7.5%	1,374,1...	1	av5pudmsdqh...	NEW	7.6%	6,677,8...	1	9vrfbx4r3nu4ka	NEW	7.6%	58,278...
2	6j3rn680wvb13	23.1%	11,409	2	bz6h5wv8k02q	↓	1	22,644	2	62ifzru1s1...	NEW	1.6%	139,987	2	av5pudmsdqh...	NEW	9.9%	6,586,8...
3	4u8m4fn1apxpt	17.4%	8,591	3	5j614ffc949y	NEW	1.4%	21,547	3	75m9ryysp65v5	NEW	1.6%	127,779	3	dc6dhv6f0uyq	NEW	0.9%	614,850
4	azsdnbv63ypwa	3.3%	1,610	4	4u8m4fn1apxpt	↓	1.2%	17,243	4	92g7g64q0cpz	NEW	1.5%	127,400	4	0bj4cvw0x3t2	NEW	0.4%	283,351
5	a9xb4b48sp14	2.4%	1,201	5	cbrnn2bnbg9...	NEW	0.7%	11,227	5	f07kahnhs930n	NEW	1.3%	111,167	5	g73b1n9r2msej	NEW	0.3%	212,544
6	6bqaev6dk7msgc	2.1%	1,048	6	5g1x85q3vmx4k	NEW	0.7%	10,388	6	23aw2rav80ptv	NEW	1.3%	111,147	6	a4n5mw2qpqlk	NEW	0.1%	47,854
7	8guhtclcnkk660	1.8%	908	7	6hpq4n88634c	NEW	0.5%	8,059	7	90vc5nrcp4tn	NEW	1.0%	85,228	7	1gj1n97gh25w	NEW	0.1%	43,964
8	9zg9qd9bm4pspu	0.6%	273	8	bgnkar02knys7	NEW	0.5%	7,019	8	6qjwvp27j4ap2	NEW	1.0%	83,985	8	6j3rn680wvb13	↓	0.0%	28,102
9	bvkiczy5hye	0.0%	0	9	8guhtclcnkk660	↓	0.4%	6,190	9	ck1tep3j3dky	NEW	0.9%	76,006	9	65yf706fmrw8	NEW	0.0%	26,690
10	fon5pzmhju11	0.0%	0	10	6bqaev6dk7msgc	↓	0.3%	4,927	10	9dgr23fbvk0p	NEW	0.9%	74,217	10	25yrahshsnk8f	NEW	0.0%	22,494

- ✓ Top SQL중 Logical Reads이 높은 Top SQL들을 확인합니다.
- ✓ 신규 업무 추가 된 비교 날짜 TOP SQL 중, "NEW"로 표시된 SQL들이 있는지 확인합니다.
- ✓ "New" 표시 된 SQL들은 신규 업무로 추가된 SQL 이거나, 기존에 수행되었으나 Plan 변경 등의 이유로 TOP으로 선정된 SQL 일 수 있습니다.

“신규로 추가 된 업무 성능이 궁금해요!”

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 추출 기준 선택

■ “NEW” SQL 성능 확인

■ “NEW” SQL 수행 추이 확인

└ SQL Analysis ▶ Long-Term Analysis 연계

■ “NEW” SQL 상세 정보 확인

└ SQL Detail 연계

■ 관련 대기 이벤트 정보 확인

└ Event Description 연계

» Top-SQL Comparison 1: M 조회 화면

RANK	Elapsed Time	CPU Time	Logical Reads	Physical Reads	Executions
1	2017-07-28	24,425	24,425	24,425	1
2		11,409	23,196	11,409	2
3		8,591	17,496	8,591	3
4		1,610	4,084	1,610	4
5		1,201	5,014	1,201	5
6		1,048	2,416	1,048	6
7		908	1,836	908	7
8		273	0,656	273	8
9		0	0,923	0	9
10		0	0,000	0	10

- 1 추출 된 Top SQL의 Value 값은 Summary 값으로 항목별 총 합을 말합니다.
- 2 추출 된 SQL 중 특정 SQL_ID 선택 시 2번 영역에 날짜별 일량을 보여줍니다.
- 3 3번 영역에 추출 된 일량을 그래프 형태로 보여지며 오른쪽 상단 토글 버튼으로 Summary 값과 Avg 값을 선택 할 수 있습니다.
- 4 선택 된 SQL_ID의 SQL Full Text, Bind 변수, 실행계획을 제공합니다.

Top-SQL Comparison의 활용

"신규로 추가 된 업무 성능이 궁금해요!"

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 추출 기준 선택

■ "NEW" SQL 성능 확인

■ "NEW" SQL 수행 추이 확인

└ SQL Analysis ▶ Long-Term Analysis 연계

■ "NEW" SQL 상세 정보 확인

└ SQL Detail 연계

■ 관련 대기 이벤트 정보 확인

└ Event Description 연계

» Top-SQL Comparison 1:M 조회 화면

The screenshot shows the MAXGAUGE Performance Analyzer interface. The main title bar says "» Top-SQL Comparison 1:M 조회 화면". Below it, the "Top-SQL Comparison" tab is selected. The interface includes tabs for "Module Comparison", "Program Comparison", "Peak Time Comparison", "Trend Comparison", and "Event Comparison". A "REALTIME MONITOR" button is also present.

The main content area displays a grid of performance metrics for various SQL statements. The columns include "Elapsed Time", "CPU Time", "Logical Reads", "Physical Reads", and "Executions". The rows represent different dates: 2017-07-28, 2017-08-09, 2017-08-10, and 2017-08-1. Each row contains several entries, each with a rank, SQL ID, and various performance values.

Below the main grid, there is a SQL editor window with the following query:

```

SELECT SYSDATE,
       tablespace_name ,
       SUM( total_space ) total_space ,
       SUM( total_space ) - SUM( used_space ) tbs_free_space ,
       SUM( used_space ) * 100 / sum( total_space ) usage
  FROM (
    SELECT /*+ use_hashf dts ddff */
           ddf.tablespace_name ,
           ddf.autoextensible ,
           CASE ddf.autoextensible
             WHEN 'NO'
               THEN GREATEST( ddf.maxbytes , ddf.bytes )
             WHEN 'YES'
               THEN ddf.bytes
           END bytes ,
       total_space ,
       ( ddf.bytes - NVL( fof.phy_free_bytes , 0 ) ) used_space
      FROM (
        SELECT tablespace_name ,
               file_id ,
               SUM( bytes ) phy_free_bytes
             FROM (
               SELECT ts.name tablespace_name ,
                      f.file# file_id ,
                      f.block_size block_size ,
                      f.length + ts.blocksize bytes ,
                      f.length blocks ,
                      f.length * ts.blocksize_free
                     FROM sys.ts$ ts
                    WHERE sys.ts$ f ,
                           sys.ts$ f.ts#
                      AND f.ts# = f.fts#
                      AND f.file# = f.reffile#
                  GROUP BY ts.name , f.file#
                  ORDER BY ts.name , f.file#
                  ) f
                GROUP BY ts.name , f.file#
                ORDER BY ts.name , f.file#
              ) fof
      GROUP BY tablespace_name ,
               file_id
      ORDER BY tablespace_name ,
               file_id
    ) ddf
  GROUP BY tablespace_name ;
  
```

Below the SQL editor, there is a "Format SQL" button. To the right of the SQL editor, there is a "Execution Plan" window displaying the execution plan for the query, which includes various stages like HASH (GROUP BY), HASH JOIN (OUTER), and TABLE ACCESS (FULL).

✓ 해당 SQL의 Plan 정보를 확인합니다.

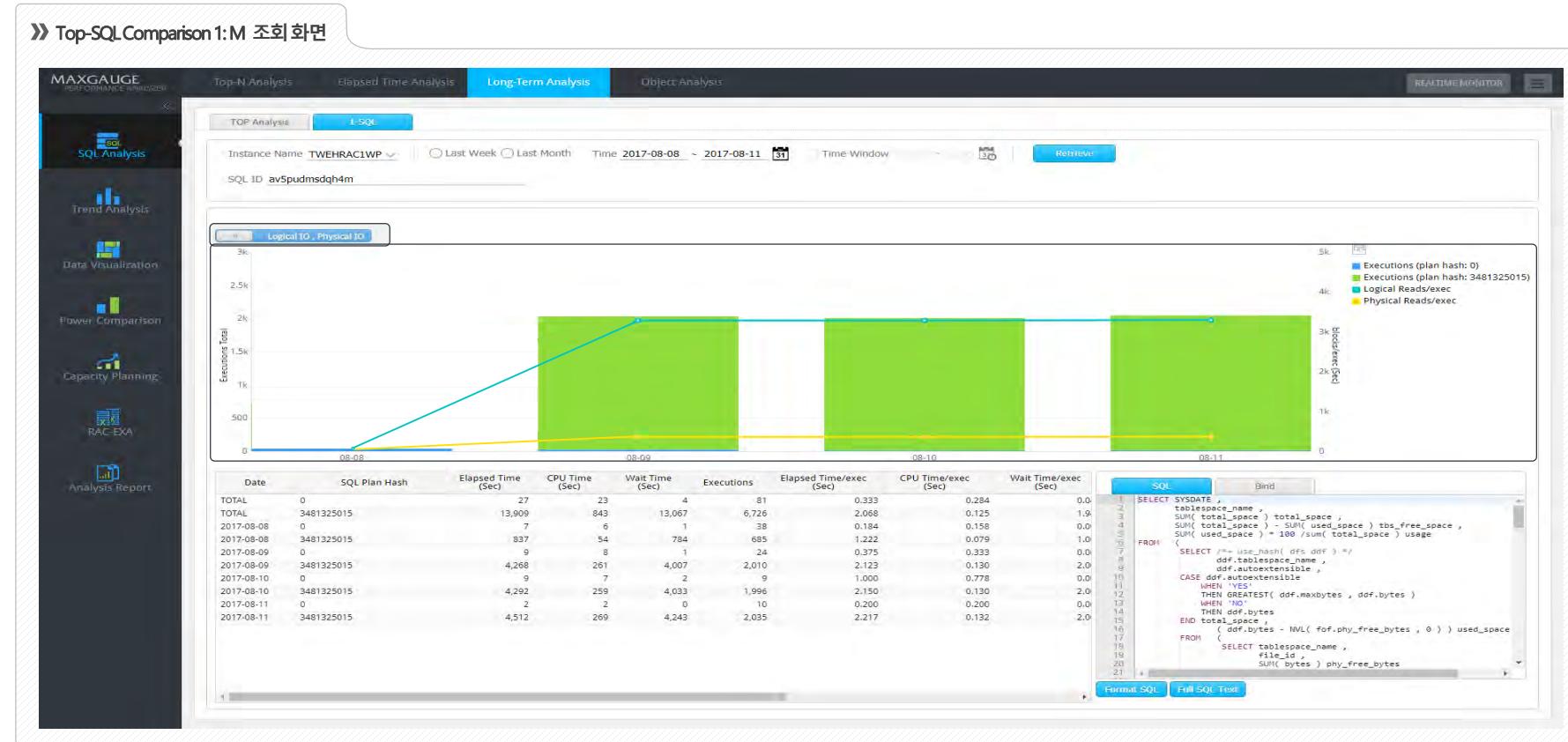
“ 신규로 추가 된 업무 성능이 궁금해요! ”

STEP

1. 검색 조건 설정

2. 데이터 분석

- 추출 기준 선택
- “NEW” SQL 성능 확인
- “NEW” SQL 수행 추이 확인
 - └ SQL Analysis > Long-Term Analysis 연계
- “NEW” SQL 상세 정보 확인
 - └ SQL Detail 연계
- 관련 대기 이벤트 정보 확인
 - └ Event Description 연계



- ✓ “New” 표시 된 SQL이 실제로 신규 업무에 해당되는 SQL인지 확인하기 위해, Long-Term Analysis를 확인합니다.
- ✓ 왼쪽 중앙 토글 버튼을 통해 각종 일 량을 비교 합니다.
- ✓ 과거 SQL 수행 이력이 없음을 통해, 신규 업무로 추가된 SQL임을 알 수 있습니다.

Top-SQL Comparison의 활용

“ 신규로 추가 된 업무 성능이 궁금해요! ”

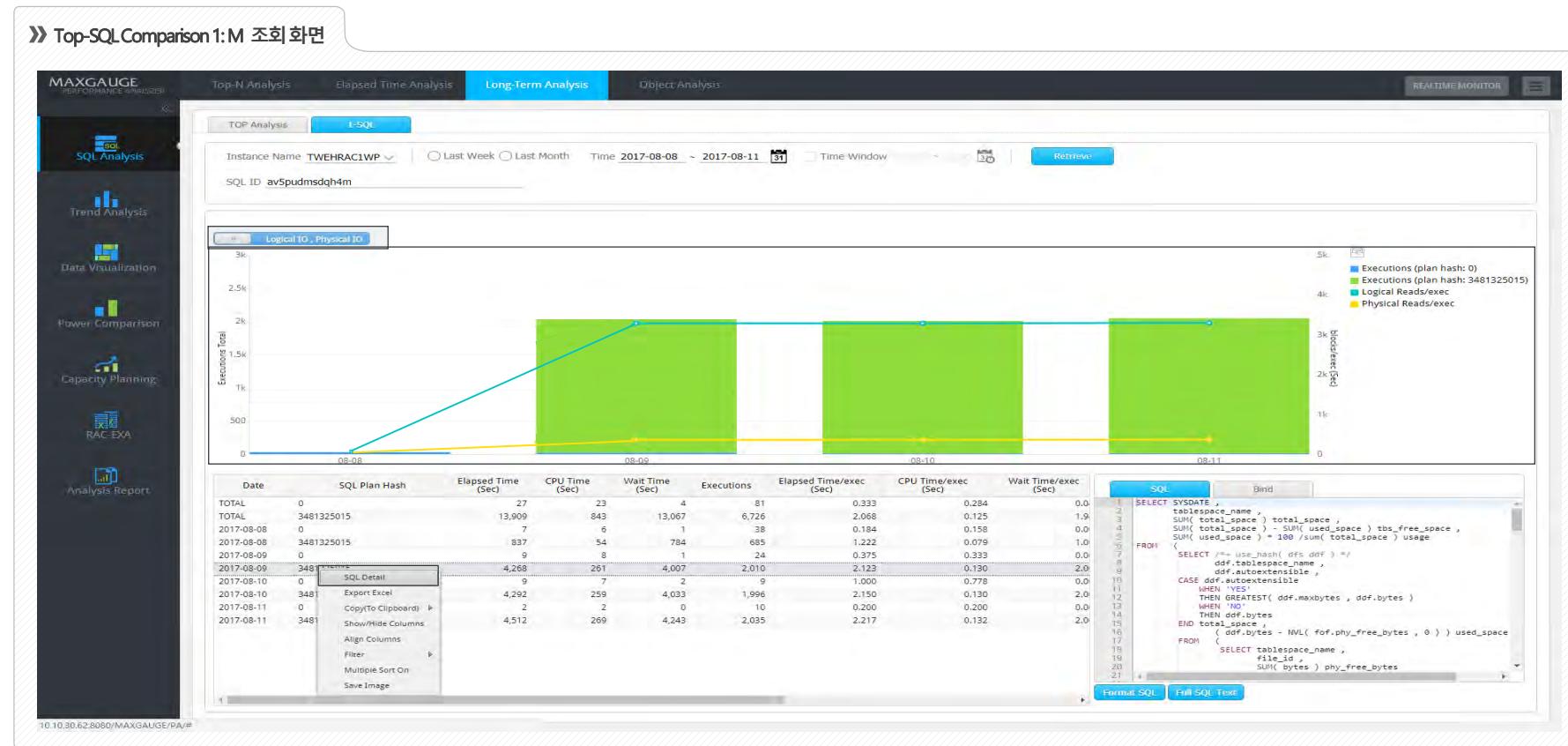
STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

▪ “NEW” SQL 상세 정보 확인

- ### ↳ SQL Detail 연결



✓ 해당 SQL의 자세한 정보를 확인하기 위해 마우스 오른쪽 클릭 후, SQL Detail을 선택합니다.

Top-SQL Comparison의 활용

“신규로 추가 된 업무 성능이 궁금해요!”

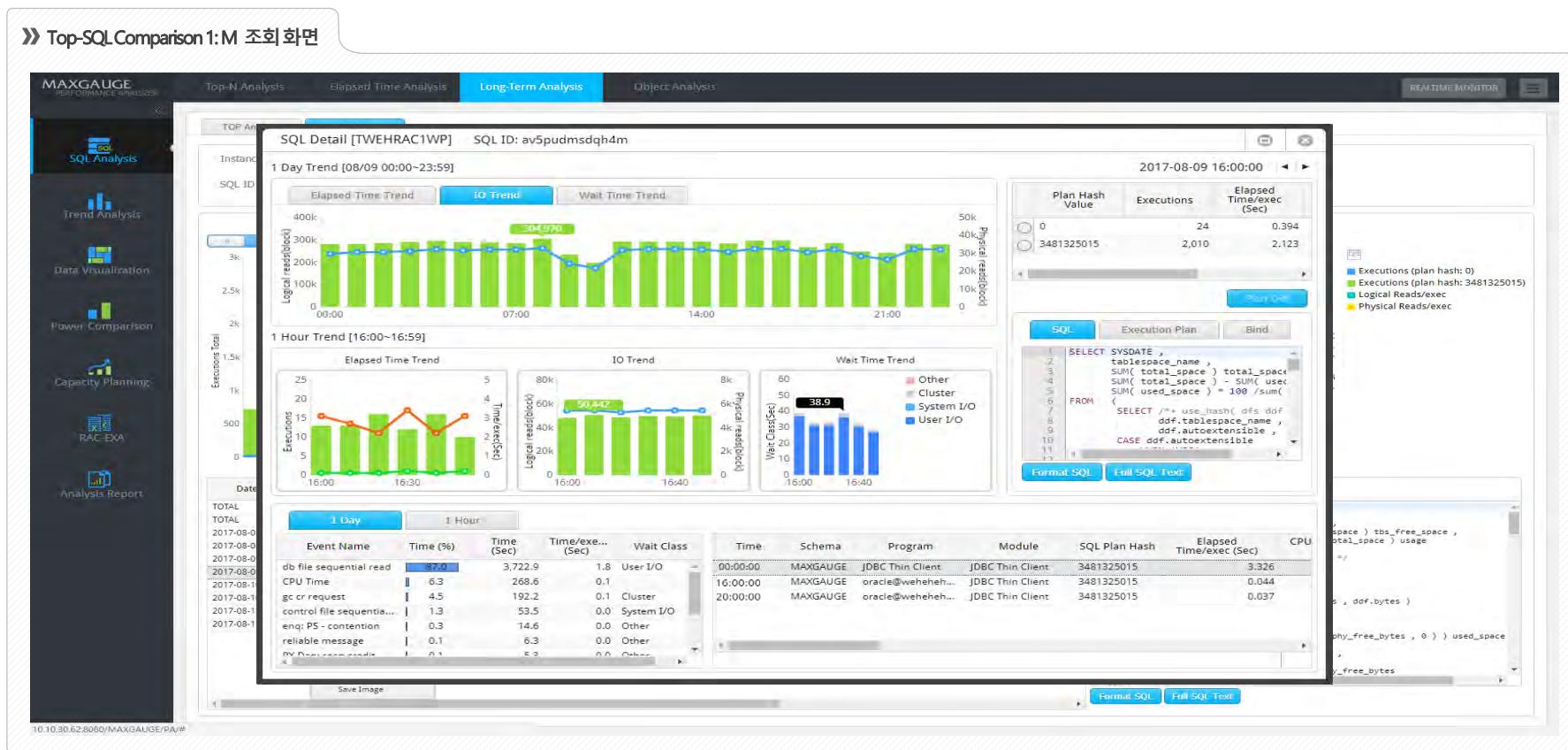
STEP

- ## 1. 검색 조건 설정

2. 데이터 분석

■ “NEW” SQL 상세 정보 확인

- ## ↳ SQL Detail 연계



- ✓ SQL Detail을 통해 선택한 SQL의 상세한 정보를 확인할 수 있습니다.
 - ✓ 해당 SQL의 시간 별 수행 정보를 통해 성능 지연의 원인을 파악할 수 있습니다.

Top-SQL Comparison의 활용

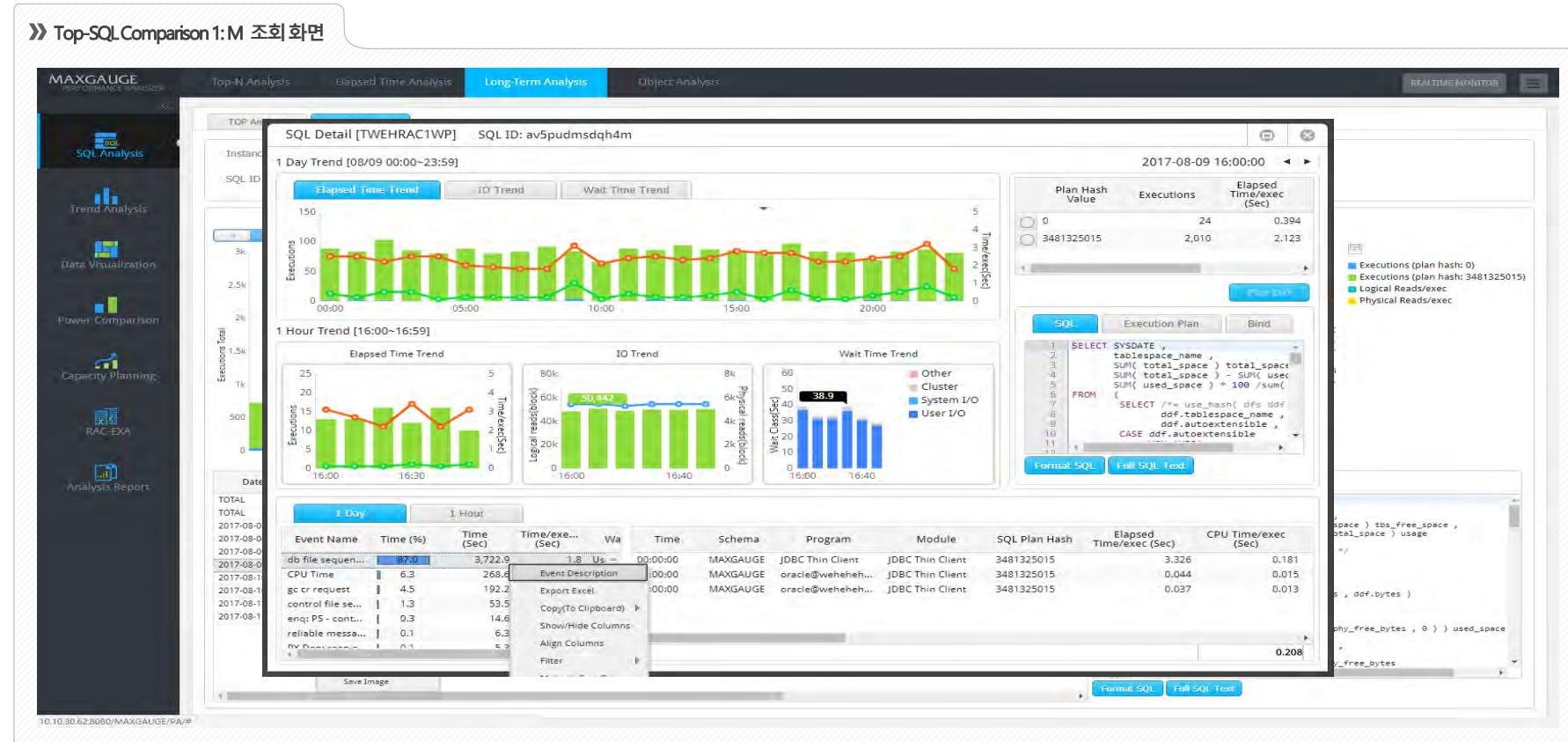
" 신규로 추가 된 업무 성능이 궁금해요! "

STEP

1. 검색 조건 설정

2. 데이터 분석

- 추출 기준 선택
- "NEW" SQL 성능 확인
- "NEW" SQL 수행 추이 확인
 - └ SQL Analysis ▶ Long-Term Analysis 연계
- "NEW" SQL 상세 정보 확인
 - └ SQL Detail 연계
- 관련 대기 이벤트 정보 확인
 - └ Event Description 연계



- ✓ "db file Sequential Reads"로 대기하는 비중이 큰 것을 알 수 있습니다.
- ✓ 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 Event Description으로 연계합니다.

Top-SQL Comparison의 활용

"신규로 추가 된 업무 성능이 궁금해요!"

STEP

1. 검색 조건 설정

2. 데이터 분석

■ 추출 기준 선택

■ "NEW" SQL 성능 확인

■ "NEW" SQL 수행 추이 확인

└ SQL Analysis ▶ Long-Term Analysis 연계

■ "NEW" SQL 상세 정보 확인

└ SQL Detail 연계

■ 관련 대기 이벤트 정보 확인

└ Event Description 연계

» Top-SQL Comparison 1:M 조회화면

The screenshot shows the MAXGAUGE Performance Analyzer interface. The main menu includes 'Top-N Analysis', 'Elapsed Time Analysis', 'Long-Term Analysis' (which is selected), and 'Object Analysis'. A sidebar lists various analysis modules: SQL Analysis, Trend Analysis, Data Visualization, Power Comparison, Capacity Planning, RAC-EXA, and Analysis Report. The 'Event Description' window is open, showing a list of events with their execution counts over time. The selected event is 'db file sequential read'. The right side of the window provides detailed help text about this event, mentioning it's a large event and providing a breakdown of its components like 'Index range scan' and 'Index unique scan'. It also includes a snippet of associated SQL code.

✓ Event Help Description 기능을 통해 해당 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

실전 분석 사례 Case 2.
지난 달과 이번 달의
월 단위 작업을 비교하고 싶어요

STEP

1. 검색 조건 설정

2. 데이터 분석

- Top-SQL 확인
- SQL의 Plan 정보 확인
- SQL 상세 정보 확인
 - ↳ SQL Detail 연계
- Plan의 자세한 정보 확인
 - ↳ SQL Detail ▶ Plan diff 연계

“지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요 !”

» Top-SQL Comparison N:M 검색 화면

The screenshot shows the Top-SQL Comparison interface with the following configurations:

1. Find Instance: RAC12CN2 selected.
2. Schema, Program, Module selection.
3. Time Window: TOP 20 selected.
4. Base Date: Monday ~ Friday selected, date range 17.08.07 ~ 17.08.11.
5. Calendar view showing the week of August 14, 2017.

- 1 비교 대상 Instance (RAC12CN2)를 선택합니다.
- 2 Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
- 3 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
- 4 기준 날짜(17.08.07 ~ 17.08.11)와 비교 날짜(17.07.10~17.07.14) 모두 Working day, Week, Month, 특정 날짜로 선택 가능합니다.
- 5 위의 설정한 조건으로 Top-SQL Comparison 실행합니다.

✓ 시나리오

INSTANCE	: RAC12CN2
SCHEMA	: (ALL)
PROGRAM	: (ALL)
월 단위 업무 일자	: 월 2주차 (2017년 7월 10일 ~ 14일 ~ 2017년 8월 7일 ~ 11일)

✓ 목표

매월 2주차마다 수행하는 월단위 작업을 비교하여 SQL의 효율성 및 변동 사항에 대한 파악

“지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

▪ Top-SQL 확인

- SQL의 Plan 정보 확인
- SQL 상세 정보 확인
 - ↳ SQL Detail 연계
- Plan의 자세한 정보 확인
 - ↳ SQL Detail ▶ Plan diff 연계

» Top-SQL Comparison N:M 조회 화면

The screenshot shows the MAXGAUGE Top-SQL Comparison interface. At the top, there are tabs for 'Top-SQL Comparison', 'Module Comparison', 'Program Comparison', 'Peak Time', and 'Trend Comparison'. The 'Top-SQL Comparison' tab is selected. Below the tabs, it displays 'Instance Name : RAC12CN2' and 'Base Date : 2017-08-09'. It also shows 'Compare Date : 2017-07-31 ~ 2017-08-04' and 'Time : 09:00 ~ 18:00'. The main area contains three tables comparing SQL execution metrics across different time periods.

Elapsed Time			CPU Time			Logical Reads			Physical Reads			Executions					
RANK	10/17~10/21 (Base)	Elapsed Time(%)	Value	RANK	10/17~10/21	Elapsed Time(%)	Value	RANK	11/21~11/25	Diff	Elapsed Time(%)	Value	RANK	11/21~11/25	Diff	Elapsed Time(%)	Value
1	f7rxuxzt64k87	20.0%	7,270,183	11	7hk2m2702ua0g	2.9%	1,038,349	1	3fw75k1snsddx	▲1	28.0%	7,270,183	11	7hk2m2702ua0g	-	2.9%	1,038,349
2	3fw75k1snsddx	13.9%	5,062,566	12	gzhkw1qu6vwxm	1.9%	677,113	2	f7nxz1t64k87	▼1	14.5%	5,062,566	12	gzhkw1qu6vwxm	-	1.9%	677,113
3	gh2g2tynpcpv1	9.6%	3,488,709	13	cv959u044n88s	1.6%	578,696	3	gh2g2tynpcpv1	-	9.6%	3,488,709	13	8zz6y2zdqjpo	▲2	1.6%	578,696
4	5ckoyqfvu60pj	6.4%	2,327,986	14	6j3rn680twb13	1.5%	542,269	4	5ckoyqfvu60pj	-	6.4%	2,327,986	14	6j3rn680twb13	-	1.5%	542,269
5	7r7636982atn9	6.1%	2,231,542	15	8z2g2tynpcpv1	1.3%	479,497	5	7r7636982atn9	-	6.1%	2,231,542	15	cv959u044n88s	▼1	1.3%	479,497
6	gzhkw1qu6vwxm	5.0%	1,814,165	16	Oy1prvxpcc2ra9	1.2%	420,159	6	gzhkw1qu6vwxm	-	5.0%	1,814,165	16	Oy1prvxpcc2ra9	-	1.2%	420,159
7	7t0959msvyt5g	4.9%	1,778,660	17	f9u2k84v884y7	1.1%	403,680	7	7t0959msvyt5g	-	4.9%	1,778,660	17	f9u2k84v884y7	-	1.1%	403,680
8	budtrrajnvw3	4.0%	1,464,767	18	cv959u044n88s	0.9%	335,452	8	dkxcqpkkyxc22	NEWW	4.0%	1,464,767	18	cv959u044n88s	-	0.9%	335,452
9	cv959u044n88s	3.9%	1,431,532	19	1qf3b7a46jm3u	0.9%	326,492	9	cv959u044n88s	-	3.9%	1,431,532	19	1qf3b7a46jm3u	-	0.9%	326,492
10	g81cbrq5yamfs	3.6%	1,303,894	20	dma0vxbwh325p	0.9%	314,119	10	g81cbrq5yamfs	-	3.6%	1,303,894	20	dma0vxbwh325p	-	0.9%	314,119

Below these tables, there is a detailed view for the SQL ID f7rxuxzt64k87. It shows the execution plan, bind variables, and execution statistics for each day from 11/21 to 11/25. The execution plan shows a SELECT statement with COUNT(*) over ORDER BY ROWNUM, a table access by index scan on ORDER_ITEMS, and a range scan on ORDER_ID. The bind variables include :B1 and :B2. The execution statistics show the number of executions, elapsed time, CPU time, and logical/physical reads for each day.

- ✓ Top SQL중 가장 Elapsed Time이 높은 Top-1 SQL을 확인합니다.
- ✓ 해당 SQL은 이전 달보다 Elapsed Time의 비중이 감소(20.0% -> 14.5%)하여 Top-1에서 Top-2로 변경 된 것을 확인 할 수 있습니다.

“지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

■ Top-SQL 확인

■ SQL의 Plan 정보 확인

■ SQL 상세 정보 확인

↳ SQL Detail 연계

■ Plan의 자세한 정보 확인

↳ SQL Detail ▶ Plan diff 연계

» Top-SQL Comparison N:M 조회 화면

The screenshot shows the MAXGAUGE Top-SQL Comparison interface. At the top, there are tabs for 'Top-SQL Comparison', 'Module Comparison', 'Program Comparison', 'Peak Time', and 'Trend Comparison'. The 'Top-SQL Comparison' tab is selected. Below the tabs, it displays 'Instance Name : RAC12CN2' and 'Base Date : 2017-08-09'. The 'Compare Date' is set to '2017-07-31 ~ 2017-08-04'. The time range for the comparison is '09:00 ~ 18:00'. There are two main sections of tables:

- Top-SQL Comparison (Left):** Shows the top 20 SQL statements by Elapsed Time for the base period (10/17~10/21). The columns include Rank, SQL ID, Elapsed Time (Sec), CPU Time (Sec), Logical Reads, Physical Reads, and Executions.
- Top-SQL Comparison (Right):** Shows the top 20 SQL statements by Elapsed Time for the compare period (11/21~11/25). The columns include Rank, SQL ID, Elapsed Time (Sec), CPU Time (Sec), Logical Reads, Physical Reads, and Executions.

Below these tables, a specific SQL statement is highlighted: **f7rxuxzt64k87**. A detailed execution plan is shown on the right, including the SQL query, bind variables, and execution plan steps. The execution plan steps are numbered 1 through 14, with comments like 'Plan Hash Value : 1063065610', 'SELECT STATEMENT', and 'TABLE ACCESS BY INDEX ROWID ORDER ITEMS(TABLE)'. The execution plan also includes predicate information and filter/access conditions.

- ✓ 선택한 SQL의 Plan 정보와 실행 횟수(Execution)당 Elapsed Time의 SUM / AVG 정보를 확인 할 수 있습니다.
- ✓ 해당 SQL의 Plan Hash Value가 2개 뜨는 것을 확인 할 수 있는데, 이를 통해 SQL의 Plan이 변경 되었다는 것을 확인 할 수 있습니다.
- ❖ SQL 실행계획이 변경 되는 원인
 1. 통계정보 갱신으로 인한 실행계획 변경
 2. 신규인덱스 생성으로 인한 실행계획 변경
 3. OPTIMIZER와 관련된 파라미터 설정 값에 의한 실행계획으로 변경

Top-SQL Comparison의 활용

"지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요!"

STEP

1. 검색 조건 설정

2. 데이터 분석

- Top-SQL 확인
- SQL의 Plan 정보 확인
- **SQL 상세 정보 확인**
 - └ SQL Detail 열기
- Plan의 자세한 정보 확인
 - └ SQL Detail ▶ Plan diff 열기

» Top-SQL Comparison N:M 조회 화면

The screenshot shows the MAXGAUGE Top-SQL Comparison interface. At the top, there are tabs for 'Top-SQL Comparison', 'Module Comparison', 'Program Comparison', 'Peak Time', and 'Trend Comparison'. The 'Top-SQL Comparison' tab is selected. Below the tabs, it displays 'Instance Name : RAC12CN2' and 'Base Date : 2017-08-09'. It also shows 'Compare Date : 2017-07-31 ~ 2017-08-04' and 'Time : 09:00 ~ 18:00'. The main area contains two large tables comparing SQL execution metrics across different dates.

Elapsed Time		CPU Time		Logical Reads		Physical Reads		Executions		Elapsed Time		CPU Time		Logical Reads		Physical Reads		Executions	
RANK	10/17~10/21 (Base)	Elapsed Time(%)	Value	RANK	10/17~10/21	Elapsed Time(%)	Value	RANK	11/21~11/25	Diff	Elapsed Time(%)	Value	RANK	11/21~11/25	Diff	Elapsed Time(%)	Value		
1	f7rxuxzt64k87	20.0%	7,270,183	11	7hk2m2702ua0g	2.9%	1,038,349	1	3fw75k1snsddx	▲1	28.0%	7,270,183	11	7hk2m2702ua0g	-	2.9%	1,038,349		
2	3fw75k1snsddx	13.9%	5,062,566	12	gzhkw1qu6vwxm	1.9%	677,113	2	f7nxz1t64k87	▼1	14.5%	5,062,566	12	gzhkw1qu6vwxm	-	1.9%	677,113		
3	gh2g2tynpcpv1	9.6%	3,488,709	13	cv959u044n88s	1.6%	578,696	3	gh2g2tynpcpv1	-	9.6%	3,488,709	13	8zz6y2zdqjpo0	▲2	1.6%	578,696		
4	5ckoyqfvu60pj	6.4%	2,327,986	14	6j3rn680twb13	1.5%	542,269	4	5ckoyqfvu60pj	-	6.4%	2,327,986	14	6j3rn680twb13	-	1.5%	542,269		
5	7r7636982atn9	6.1%	2,231,542	15	8zz6y2zdqjpo0	1.3%	479,497	5	7r7636982atn9	-	6.1%	2,231,542	15	cv959u044n88s	▼1	1.3%	479,497		
6	gzhkw1qu6vwxm	5.0%	1,814,165	16	Oy1prvxpcc2ra9	1.2%	420,159	6	gzhkw1qu6vwxm	-	5.0%	1,814,165	16	Oy1prvxpcc2ra9	-	1.2%	420,159		
7	7t0959rmsvyt5g	4.9%	1,778,660	17	f9u2k84v884y7	1.1%	403,680	7	7t0959rmsvyt5g	-	4.9%	1,778,660	17	f9u2k84v884y7	-	1.1%	403,680		
8	budtrrajnvw3	4.0%	1,464,767	18	cv959u044n88s	0.9%	335,452	8	dkxcapqkyc22	NEWW	4.0%	1,464,767	18	cv959u044n88s	-	0.9%	335,452		
9	cv959u044n88s	3.9%	1,431,532	19	1qf3b7a46jm3u	0.9%	326,492	9	cv959u044n88s	-	3.9%	1,431,532	19	1qf3b7a46jm3u	-	0.9%	326,492		
10	g81cbqr5yamfs	3.6%	1,303,894	20	dma0vxbwh325p	0.9%	314,119	10	g81cbqr5yamfs	-	3.6%	1,303,894	20	dma0vxbwh325p	-	0.9%	314,119		

Below these tables, there is a detailed view for SQL ID f7rxuxzt64k87. It shows a table of execution details for various dates, a 'SQL Detail' dropdown menu, and a 'Bind' and 'Execution Plan' section.

SQL ID f7rxuxzt64k87

Date	SQL Plan Hash	Executions	Elapsed Time(ms)	CPU Time(ms)	Wait Time(ms)	Logical Reads	Physical Reads	Elapsed Time(ms)	CPU Time(ms)	Wait Time(ms)	Logical Reads	Physical Reads
11/21	37538028	1772	33,104	35,198	0.006	71264	0	58,819	58,819	10	1260360509	0
11/22	67707464	1522	20,388	20,388	0	286125	0	47,437	47,437	10	980369234	0
11/23	32338623	0	239184	462721	21	0	5022863	0	0	0	0	0
11/24	677077464	0	711264	0	58,819	58,811	10	1260360509	0	0	0	0
11/25	32338623	0	479,688	96,5442	42	42	0	100457260	0	0	0	0

Bind and Execution Plan sections are also visible on the right side of the interface.

✓ 선택한 SQL Plan의 자세한 정보를 확인하기 위해 마우스 오른쪽 클릭 후, SQL Detail을 선택합니다.

“지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Top-SQL 확인
- SQL의 Plan 정보 확인
- **SQL 상세 정보 확인**
 - └ SQL Detail 연계
- Plan의 자세한 정보 확인
 - └ SQL Detail ▶ Plan diff 연계

» Top-SQL Comparison N:M 조회 화면

The screenshot shows the Top-SQL Comparison application's main interface. On the left, there's a sidebar with various analysis modules like SQL Analysis, Trend Analysis, Data Visualization, Power Comparison, Capacity Planning, PAGE-XA, and Analysis Report. The main area is titled "SQL Detail [TWEH11WP] SQL ID: 29qp1ousqkqh0". It displays several charts: "1 Day Trend [08/07 00:00~23:59]" with Elapsed Time, IO, and Wait Time trends; "1 Hour Trend [10:00~10:59]" for the same metrics; and a "1 Day" event timeline table. To the right, there's a "Plan Hash" section showing execution counts and elapsed times for different plans, and a "Plan Diff" button. Below these are tabs for "SQL", "Execution Plan", and "Bind". A large text area at the bottom contains the raw SQL query: "SELECT TT.ORDER_TOTAL, TT.SALES_REP_ID, ...".

- ✓ SQL Detail을 통해 선택한 SQL의 상세한 정보를 확인 할 수 있습니다.
- ✓ 변경 된 Plan의 정보를 오른쪽 상단에서 확인 할 수 있으며, 변경 된 Plan 간의 비교를 위해 Plan을 선택 후 Plan diff를 클릭합니다.

Top-SQL Comparison의 활용

"지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요!"

STEP

1. 검색 조건 설정

2. 데이터 분석

- Top-SQL 확인
- SQL의 Plan 정보 확인
- SQL 상세 정보 확인
 - ↳ SQL Detail 연계
- Plan의 자세한 정보 확인
 - ↳ SQL Detail ▶ Plan diff 연계

» Top-SQL Comparison N:M 조회 화면

RANK	Time	Sql Plan Hash	Executions	Elapsed_Time/exec (Sec)	CPU Time/exec (Sec)	Logical Reads/exec (block)	Physical Reads/exec (block)
1	2017-08-07 22:10:00	999494841	6	43.425	1.183	19,266	5,239
2	2017-08-07 22:20:00	549764709	14	136.329	0.786	25,092	3,465
3							
4							
5							
6							
7							
8							
9							
10							

✓ 이전 Plan과 변경된 현재 Plan을 동시에 확인하여 어떻게 Plan이 변경되었는지 확인 할 수 있습니다.

❖ Plan의 변경 내용뿐만 아니라 변경 원인까지 파악할 수 있는 기능을
현재 개발 중에 있으며 자세한 내용은 첨부 된 자료에서 자세히 확인 하실 수 있습니다.



MAXGAUGE



Practical Guide

Trend Comparison [Performance Analyzer]

Contents

Trend Comparison?

Trend Comparison의 활용

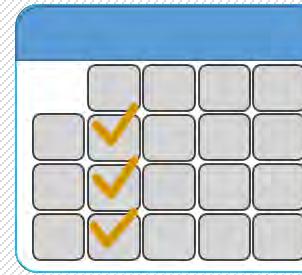
Case 1. Storage 이전 작업 전후의 성능을 비교하고 싶어요.

Case 2. 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요.

Trend Comparison?

Trend Comparison은 언제 쓰나요?

"(Storage 이전 작업과 같은)
데이터베이스 환경 변화 발생 시
전후 시점의 성능을 비교하고 싶을 때



» 매주 같은 시간 반복적으로
장애를 발생시키는 원인을 알고 싶을 때

Trend Comparison?

Trend Comparison

- 여러 개의 인스턴스, 혹은 여러 시점의 동일한 시간 축으로 Trend Performance 간 비교 기능이다.
- 최대 6일까지의 로그에 대한 Stat, Wait, Ratio, OS Stat에 대한 지표에 대해 비교 분석 및 Active 세션에 대한 겹쳐진 이미지를 제공한다.
- Default로 Active session / session logical reads / lock waiting session / CPU / execute count 성능 지표에 대한 성능 추이 비교 화면을 제공한다.
- 해당 기능은 Performance Analyzer ▶ Power Comparison ▶ Trend Comparison 경로를 통해 사용할 수 있다.



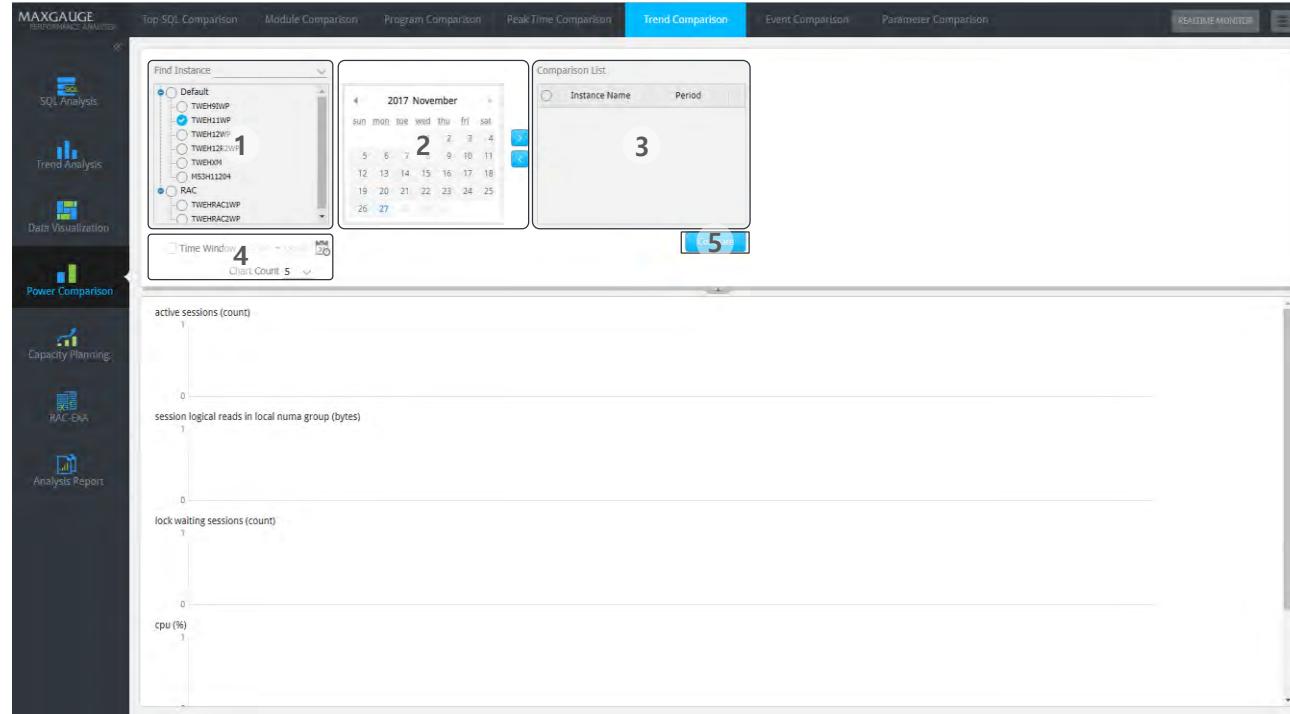
Trend Comparison의 활용

Trend Comparison의 사용 방법

Trend Comparison

- 검색 조건 설정
- 데이터 분석

» Trend Comparison 검색 화면



- 1 분석 대상 Instance를 입력하거나, 선택합니다.
- 2 달력 UI를 이용하여, 분석 대상 일자를 선택하여, 설정할 수 있습니다.
- 3 상세 시간분석 대상의 인스턴스 및 일자가 표시됩니다. 최대 6일 까지 설정 가능합니다.
- 4 구분 및 분석 대상 지표의 수 설정(최소 4개~최대 16개) 시 선택하는 탭이며 상세 시간은 00:00~23:59 사이를 선택적으로 선택 할 수 있습니다.
- 5 위의 설정한 조건으로 **Trend Comparison**를 실행합니다.

Trend Comparison의 사용 방법

Trend Comparison

- 검색 조건 설정
- 데이터 분석

» Trend Comparison 조회 화면



- 선택한 분석 대상의 정보 및 Trend 표시 색상을 나타냅니다.
- Default로 다음과 같은 성능 지표(MAX)에 대한 성능 추이 비교 화면을 제공

Active session (count) / session logical reads (blocks) / lock waiting session (count) / cpu (%) / execute count (count)

Trend Comparison의 사용 방법

Trend Comparison

- 검색 조건 설정
- 데이터 분석

» Trend Comparison 조회 화면



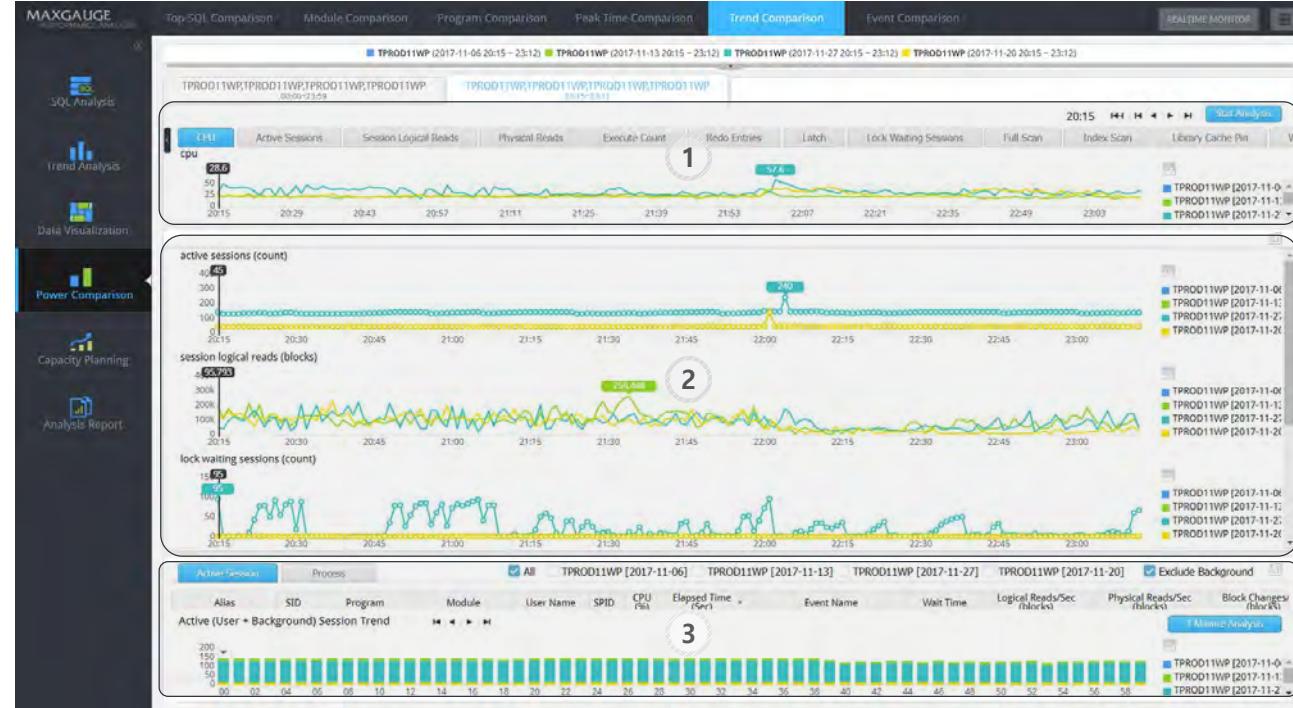
- 1 Trend Comparison Area의 Peak 구간을 마우스로 Drag 하여, 상세 분석 화면으로 연계합니다.
- 2 Time Slice Window에서 분석 구간을 재 설정하거나 "OK" 버튼을 클릭합니다.

Trend Comparison의 사용 방법

Trend Comparison

- 검색 조건 설정
- 데이터 분석

» Trend Comparison 조회 화면



- 1 초기 화면의 Performance Trend Area에서 제공하던 성능 지표의 1분 단위 추이를 나타냅니다.
- 2 기본적으로 4개의 성능 지표의 추이를 보여주며, 마우스 더블 클릭으로 지표명 변경이 가능합니다.
- 3 초 단위 active session 정보와 분 단위 Top-N process 정보 제공

실전 분석 사례 Case 1.
**Storage 이전 작업 전후의 성능을
비교하고 싶어요.**

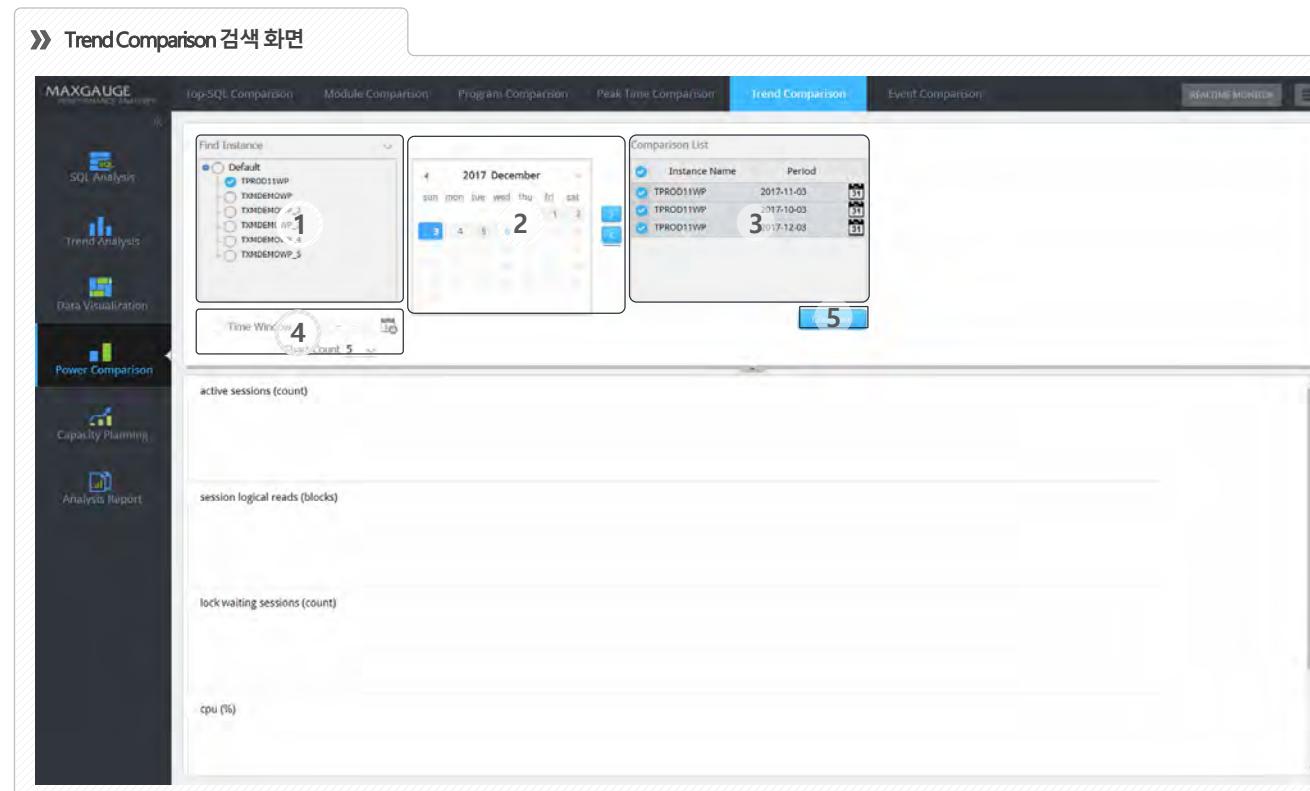
“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Trend 비교 및 확인
- 상세 Trend 비교 확인
 - └ Stat Analysis 연계
- Top Event 확인
 - └ TOP Events ▶ Event Description 연계



- 1 비교 대상 Instance (TROD11WP)를 선택합니다.
- 2 분석하고자 하는 날짜(17.10.03, 17.11.03, 17.12.03)를 원하는 날짜로 선택 가능합니다.
- 3 분석 대상의 인스턴스 및 일자가 표시되는 부분으로, **최대 6일** 까지 설정 가능합니다.
- 4 분석 지표의 수(Default: 5개) 및 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택할 수 있습니다.
- 5 위의 설정한 조건으로 **Trend Comparison 실행합니다.**

✓ 시나리오

INSTANCE : TROD11WP
 DATE : Storage 이전 작업 전후 날짜
 (2017년 10월 3일,
 2017년 11월 3일,
 2017년 12월 3일)

Storage 이전 작업일 : 2017년 11월 2일

✓ 목표

Storage 이전 작업 이전, 이후의 Trend를
 비교하여 변동 사항 및 성능 비교 분석

“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

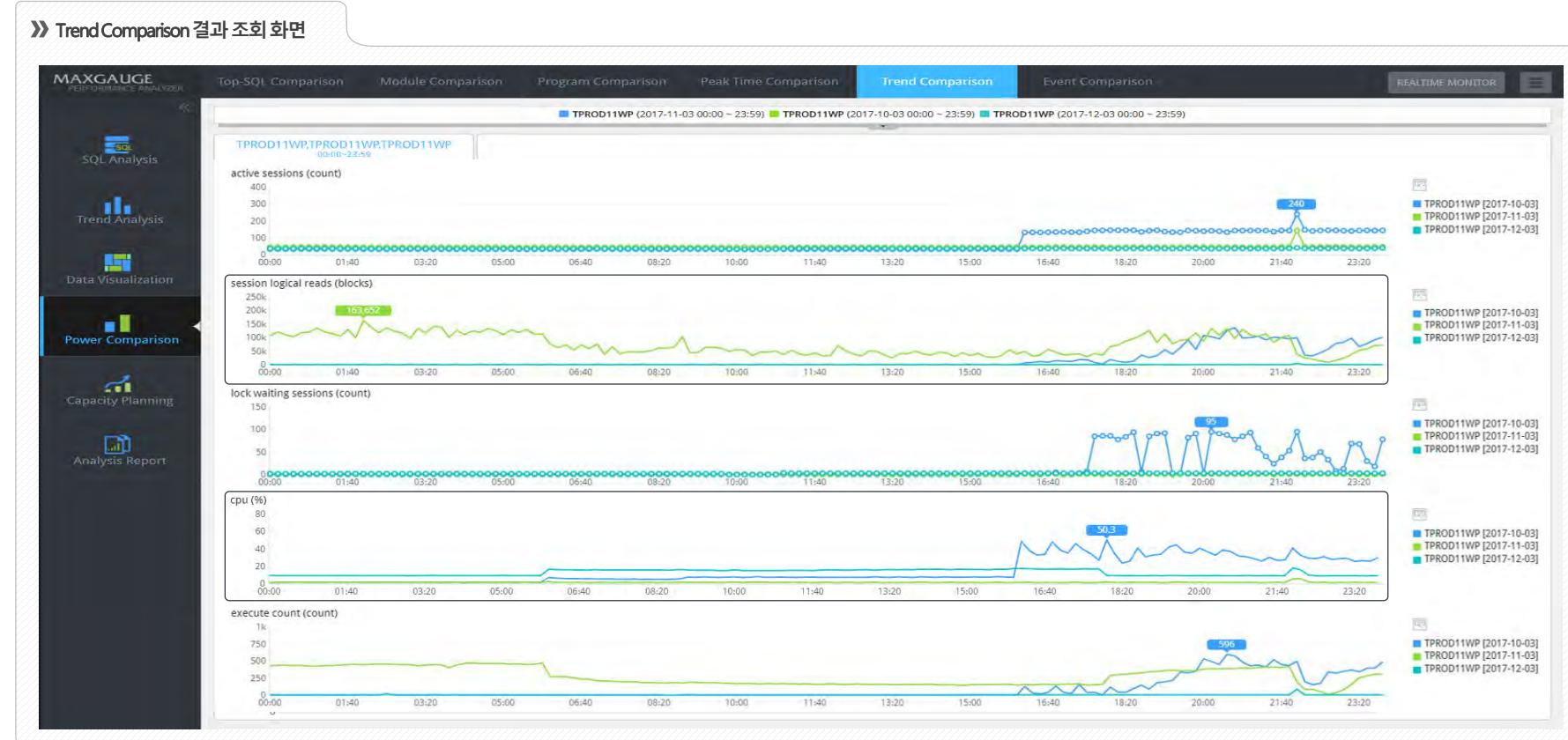
▪ Trend 비교 및 확인

▪ 상세 Trend 비교 확인

↳ Stat Analysis 연계

▪ Top Event 확인

↳ TOP Events ▶ Event Description 연계



- ✓ Storage 이전 작업 전(17.10.03)과 직후(17.11.03) 및 이후(17.12.03)의 지표의 Trend 추이 변화를 확인합니다.
- ✓ CPU 사용률의 추이를 각 날자별로 비교해 보면 서버교체 이전 보다 Storage 교체 직후에 CPU 사용률이 30~40% 수준으로 향상된 것을 확인할 수 있습니다.
- ✓ 반면 Session Logical reads의 추이는 모두 비슷한 것으로 보아, I/O 발생량 자체는 큰 변화가 없는 것으로 보여집니다.

“ Storage 이전 작업 전후의 성능을 비교하고 싶어요 ! ”

STEP

1. 검색 조건 설정

2. 데이터 분석

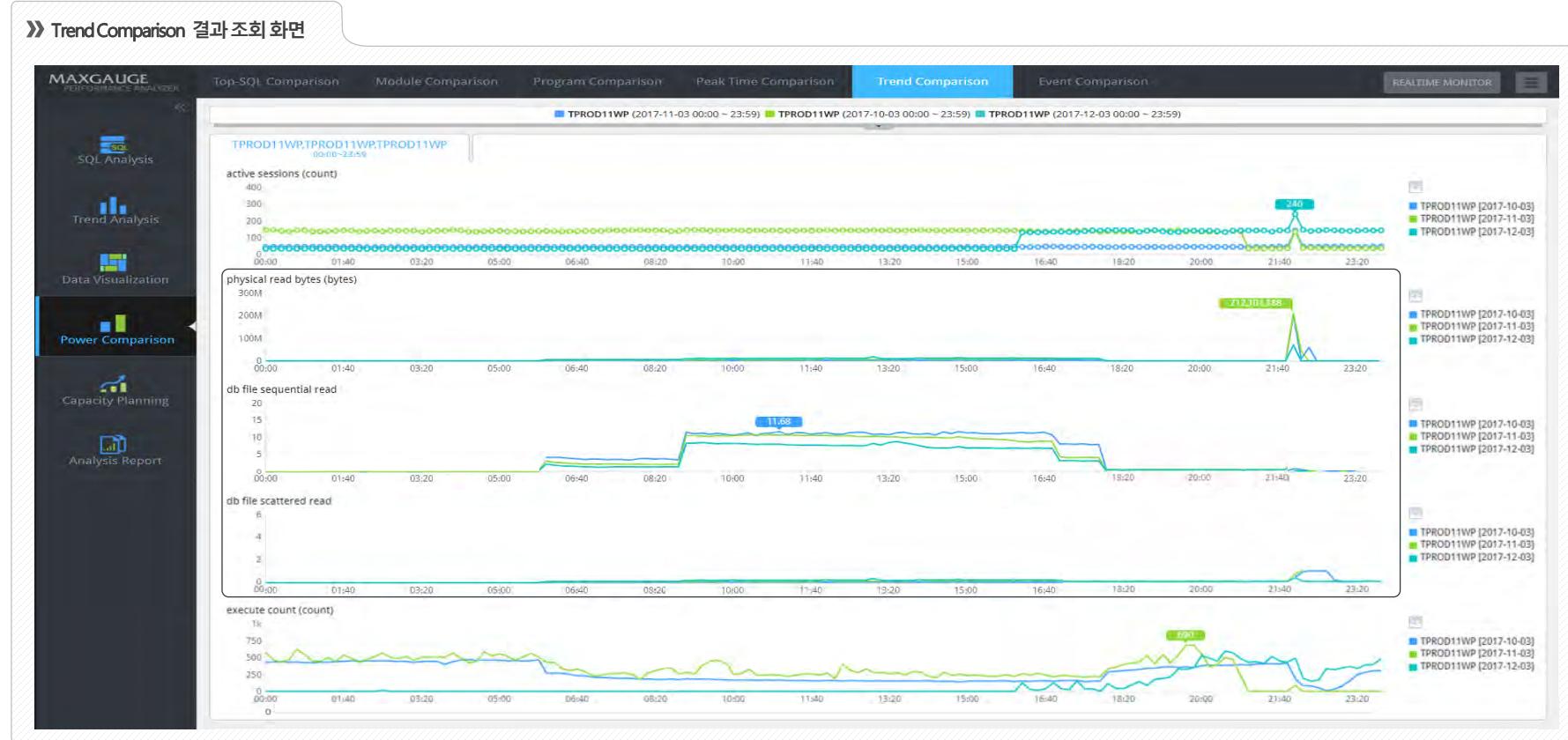
▪ Trend 비교 및 확인

▪ 상세 Trend 비교 확인

↳ Stat Analysis 연계

▪ Top Event 확인

↳ TOP Events ▶ Event Description 연계



- ✓ 지표를 변경하여 서버 교체 작업 이전, 이후의 Trend 추이를 비교합니다.
- ✓ db file sequential read 대기 이벤트는 storage 이전 작업 이후 대기 시간이 감소한 것으로 보여지고 있습니다.
- ✓ 또한, Physical reads, db file scattered read 등 다른 I/O 오퍼레이션의 처리 시간의 경우, 서버 교체 전후 모두 비슷한 추이를 나타내고 있습니다.

“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

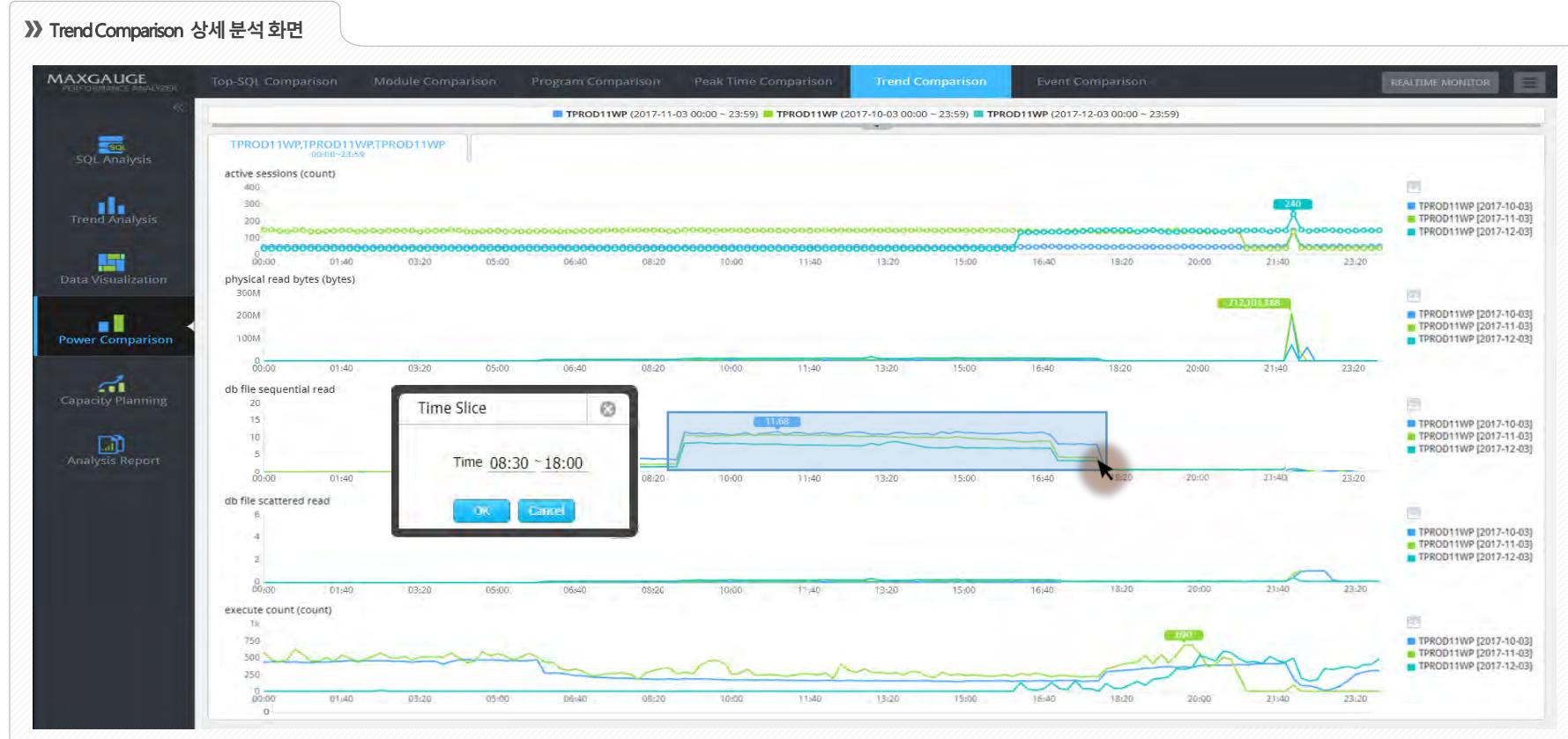
- Trend 비교 및 확인

- 상세 Trend 비교 확인

 - Stat Analysis 연계

- Top Event 확인

 - TOP Events ▶ Event Description 연계



✓ 업무 시간대에 대한 상세 분석을 위해, 분석 구간을 마우스로 Drag 후 시간 설정하여 “OK” 버튼을 클릭합니다.

“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

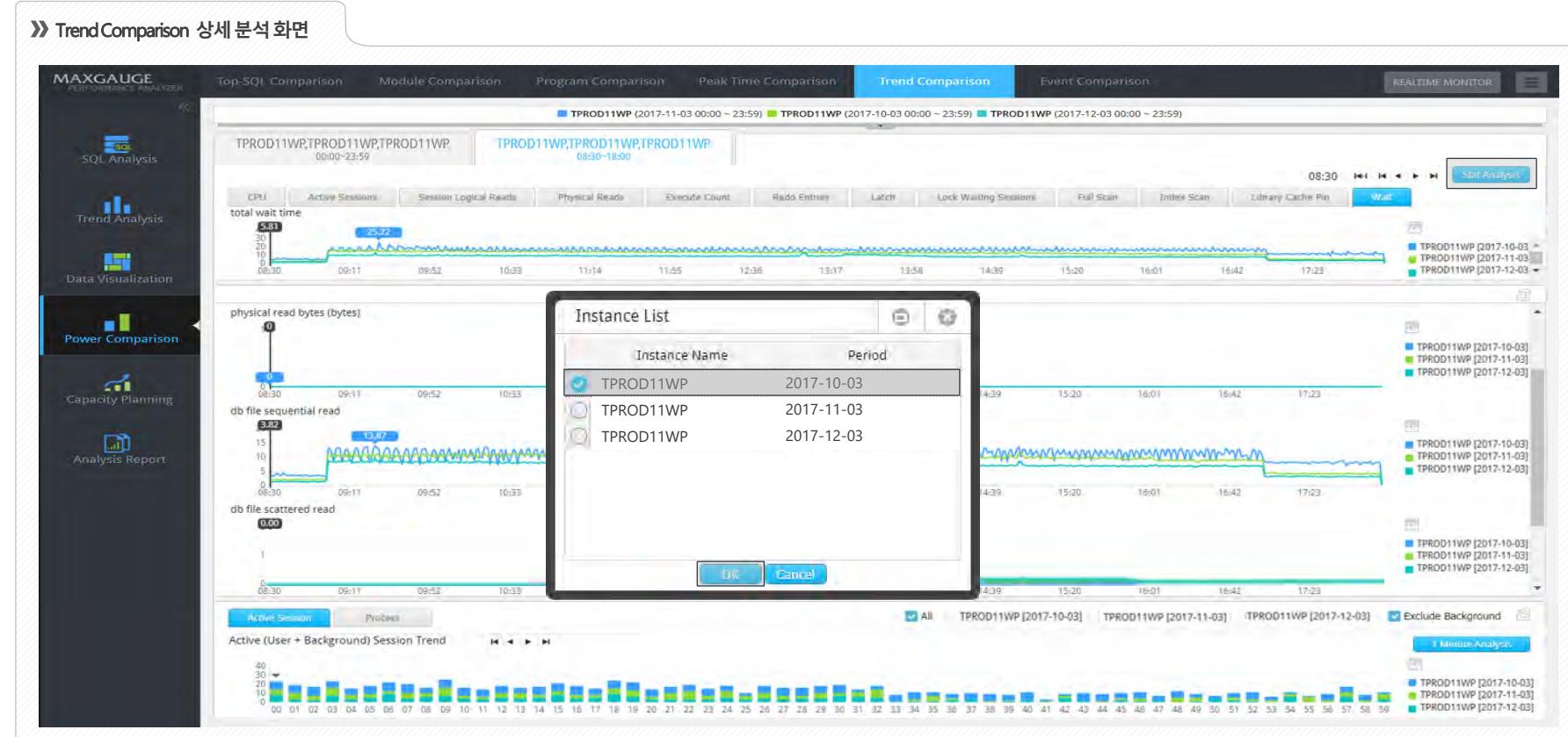
▪ Trend 비교 및 확인

▪ 상세 Trend 비교 확인

└ Stat Analysis 연계

▪ Top Event 확인

└ TOP Events ▶ Event Description 연계



- ✓ 작업 전후의 Total wait time 지표를 비교해보고, 각 날짜별 분석을 위해 Stat Analysis 를 클릭하여 해당 인스턴스 명, 날짜를 선택합니다.

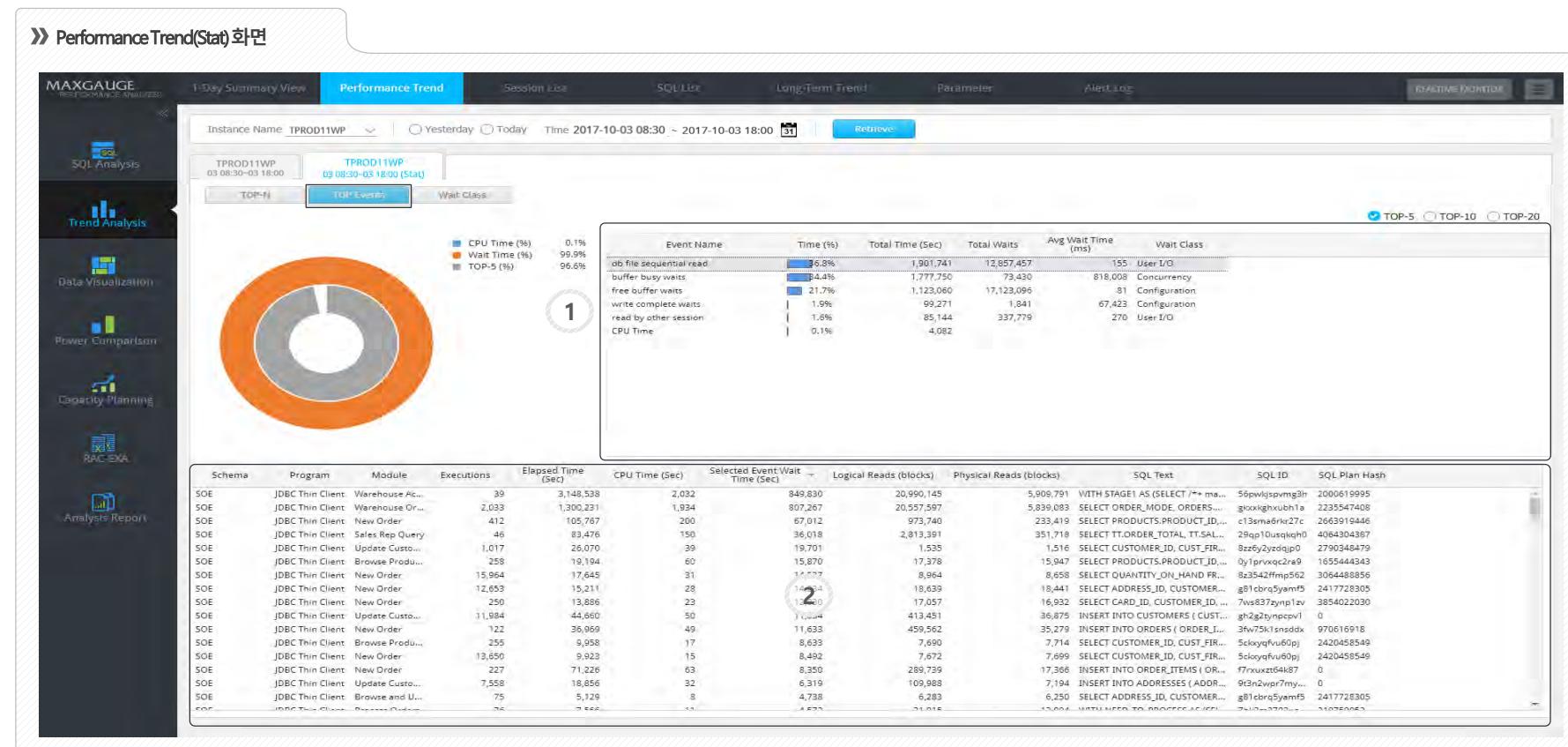
“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Trend 비교 및 확인
- 상세 Trend 비교 확인
 - ↳ Stat Analysis 연계
- Top Event 확인
 - ↳ TOP Events ▶ Event Description 연계



1 Stat Analysis를 통해, 작업 이전(17.10.03) 시점의 TOP Events를 상세하게 확인할 수 있습니다.

» db file sequential read 대기 이벤트가 평균 0.155초 대기하며 비중이 가장 큰 것을 알 수 있습니다.

2 추출 된 TOP Events 중 대기 이벤트 선택 시, 2번 영역에 해당 대기 이벤트를 대기한 SQL 목록을 보여줍니다.

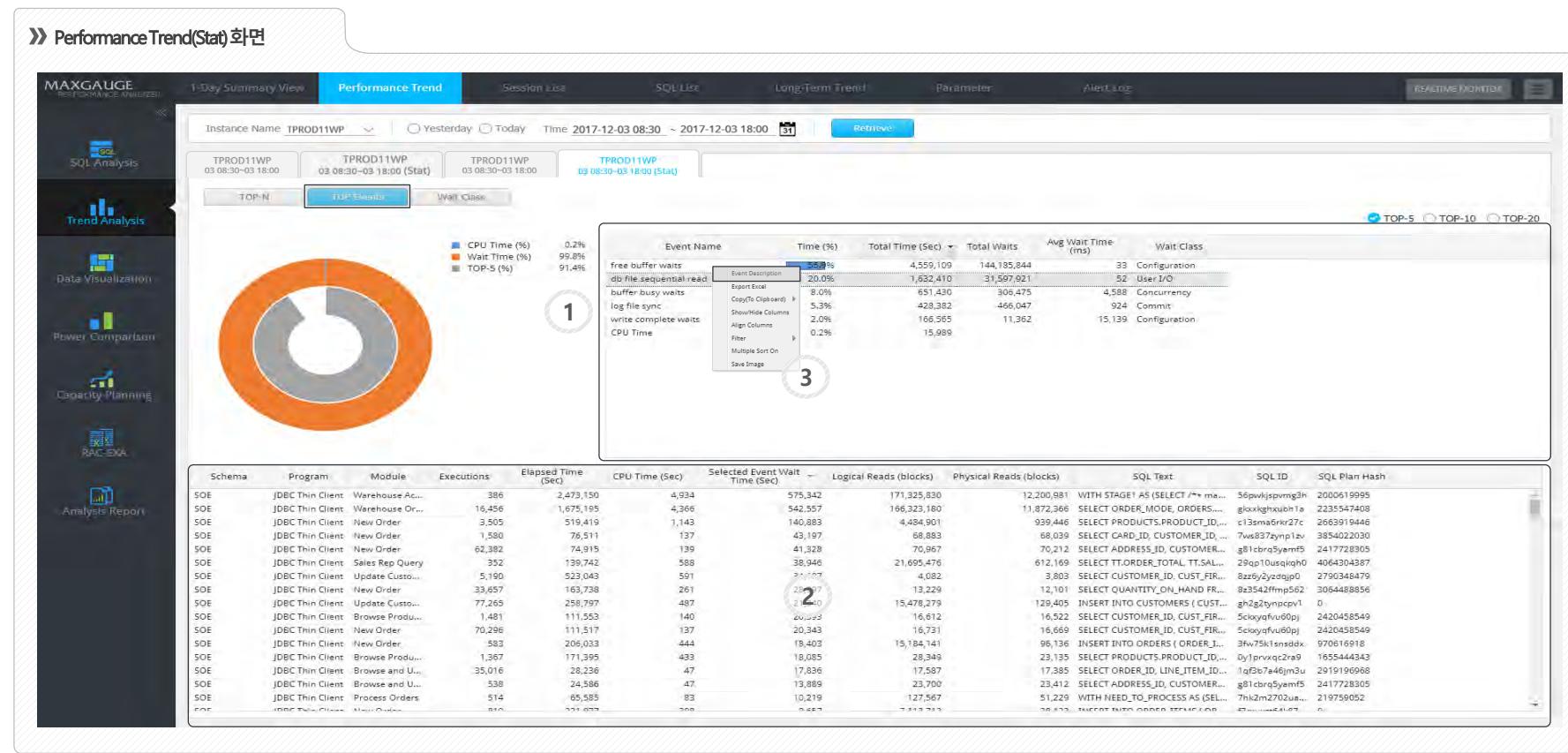
“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Trend 비교 및 확인
- 상세 Trend 비교 확인
 - ↳ Stat Analysis 연계
- Top Event 확인
 - ↳ TOP Events ▶ Event Description 연계



1 Stat Analysis를 통해, 작업 이후 (17.12.03) 시점의 TOP Events 를 상세하게 확인할 수 있습니다.

» db file sequential read 대기 이벤트가 Top-1에서 Top-2로 변경되며, 평균 대기 시간 또한 감소(0.155 -> 0.052 sec) 됨을 알 수 있습니다.

» 해당 I/O 오퍼레이션의 효율의 향상되어 CPU 점유율 향상에 영향을 미친 것으로 보여집니다.

2 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 Event Description으로 연계합니다.

“Storage 이전 작업 전후의 성능을 비교하고 싶어요 !”

STEP

1. 검색 조건 설정

2. 데이터 분석

- Trend 비교 및 확인
- 상세 Trend 비교 확인
 - ↳ Stat Analysis 연계
- Top Event 확인
 - ↳ TOP Events ▶ Event Description 연계

» Performance Trend(Stat) 화면

The screenshot shows the MAXGAUGE Performance Trend(Stat) interface. The main window displays a circular progress bar and a table of events. A detailed modal window is open for the 'db file sequential read' event. The modal has a tree view under 'Event Description' and a large text area with explanatory notes. The notes describe the event as a large I/O operation involving multiple disk reads, and mention specific cases like index range scans and full table scans.

Event Description

- db file sequential read

대기이벤트의 대기파라미터는 다음과 같다.
 P1 = File#
 P2 = Block#
 P3 = 블록수 (대부분 1이다. 오라클 7버전에서는 temp 세그먼트를 읽을 때 1이상인 경우가 있다)
 오라클을 사용하면서 발생되는 I/O 관련 이벤트인 db file sequential read는 오라클 DB에서 Disk I/O 대기 이벤트는 항상 발생하는 것으로 비정상적으로 높은 대기가 발생한다면 Disk I/O 처리부분 및 설정부분을 점검을 하여야 한다.

db file sequential read 발생 사유

- Single Block I/O
 - Index 결유한 Scan
 - Index Range Scan
 - Index Unique Scan
 - Index Full Scan
- * Chained Row / Migrated Row
- Multi Block I/O 도중에도 발생 가능

Index range scan이나 index unique scan도 또한 비효율적으로 많은 I/O가 발생된다면, DB Server의 성능저하를 가져올 수 있다.

첫번째, Index Full Scan은 index의 모든 Block을 Sequential하게 Read하게 되는데
 Index Full Scan later Table Access By Index Rowid 를 처리가 되는 SQL은 I/O 발생률이
 과다하게 발생될 우려가 있으므로 Index Full Scan이 발생된다면 Random I/O가 발생되는지
 I/O 발생률이 얼마나 되는지 등을 면밀히 체크를 해야 할 것이다.

두번째, Chained row는 Block size 보다 큰 Insert가 수행되면 하나의 row가 두 개의 blocks에
 걸쳐 일련되게 되는 row 의미하고 Chained row는 같은 rowid를 갖는 row를 대상으로 발생되고
 Update는 같은 row 기준으로 처리되는 경우 해당 row를 저장할 수 없을 경우 다른 block으로 이동하게
 될때 발생된다. Chained row와 Migrated row는 Single block I/O나 Multi block I/O 발생 시에
 부가적인 single block I/O를 처리해야 하기 때문에 DB Server내에 chained row & migrated row가
 과다하게 발생되고 있다면 I/O 발생률이 과다하게 발생되게 되고 이로 인해 db file sequential read
 대기 이벤트가 과다하게 발생될 수 있고 DB Server의 성능저하를 가져올 수 있다.

세번째, Multi block I/O 도중에 발생 가능한 한 db file sequential read 대기 이벤트가 발생되는 원인 중
 가장 잘 관찰되는 것은 Chained row가 발생된 rows를 가진 block을 조회할 경우라고 할 수 있다.
 db file sequential read 대기 이벤트의 발생은 절상적인 I/O 처리를 하는 부분에서 발생될 수
 있으나 위와 같은 경우들이 과다하게 발생된다면 좀더 면밀히 점검을 해보아 한다.

마지막으로 Case에서도 db file sequential read 대기가 과다하게 발생될 수 있다.
 아래와 같은 경우 모니터링을 꾸준히 하여 DB Server 성능저하 Factor를 제거하여야 한다.

✓ Event Help Description 기능을 통해 “db file sequential read” 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

실전 분석 사례 Case 2.
(매주 같은 시간)
반복적으로 장애를 발생시키는
쿼리를 찾고 싶어요.

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

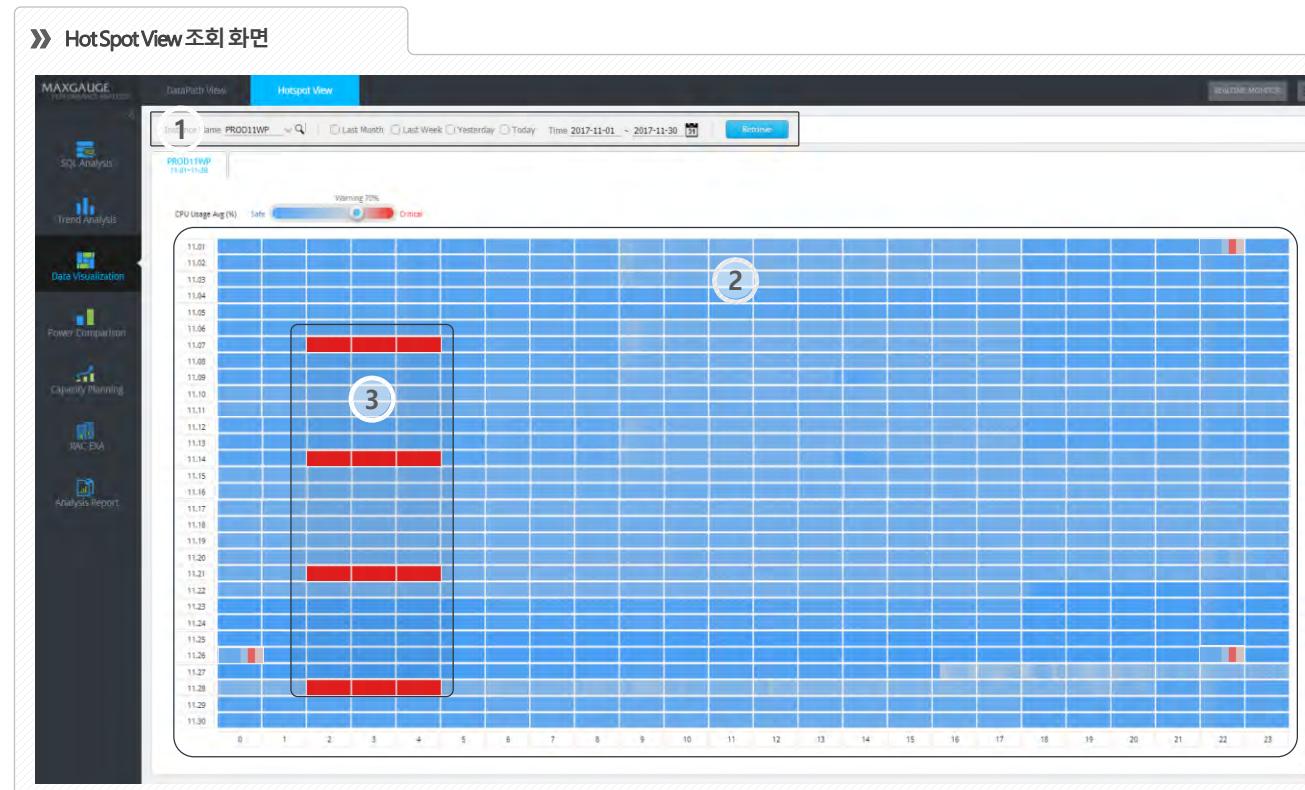
1. 장애 구간 파악

└ Data Visualization ▶ Hotspot View

2. 검색 조건 설정

3. 데이터 분석

- 매주 같은 요일 Trend 비교
- 상세 Trend 비교
 - └ Stat Analysis 연계
- 문제 SQL 상세 정보 확인
 - └ TOP-N ▶ SQL ▶ SQL Detail 연계



- ① 비교 대상 Instance (PROD11WP)에 대하여 한달 간의 CPU 사용률 분석을 위한 구간 (2017.11.01 ~ 2017.11.30) 선택합니다.
- ② 각각의 개별 Box는 1시간을 10분 단위로 6등분하여 1시간 내의 CPU(%)이 높은 구간을 확인할 수 있도록 Hotspot을 제공합니다.
- ③ Hotspot View 조회 결과, 매주 화요일 02:00~05:00 시간 대에 CPU 사용률이 높은 것을 확인할 수 있습니다.

✓ 시나리오

INSTANCE	: PROD11WP
분석 구간	: 2017년 11월
장애 발생 시점	: 매주 화요일 02:00 ~ 05:00 (2017.11.07, 2017.11.14, 2017.11.21, 2017.11.28)

✓ 목표

매주 같은 요일 시간 대 CPU 사용률에 영향을 미치는 장애 원인 파악.

STEP

1. 장애 구간 파악

└ Data Visualization ▶ Hotspot View

2. 검색 조건 설정

3. 데이터 분석

- 매주 같은 요일 Trend 비교

- 상세 Trend 비교

└ Stat Analysis 연계

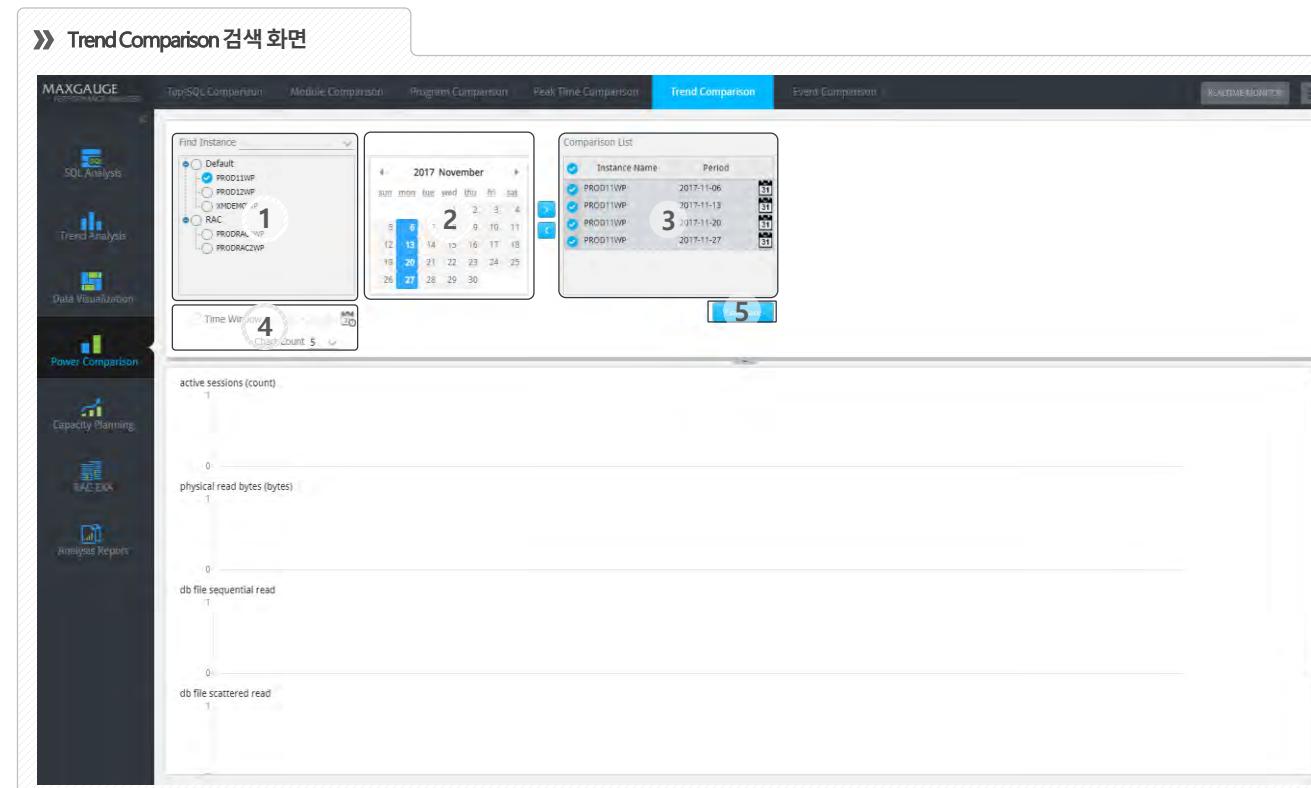
- TOP SQL 정보 확인

└ TOP-N ▶ SQL

- TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계

" 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! "



✓ 검색 조건

INSTANCE

: PROD11WP

장애 발생 시점

: 매주 화요일 02:00 ~ 05:00
 (2017.11.07, 2017.11.14,
 2017.11.21, 2017.11.28)

- 1 비교 대상 Instance (PROD11WP)를 선택합니다.
- 2 분석하고자 하는 날짜(17.11.07, 17.11.14, 17.11.21, 17.11.28)를 원하는 날짜로 선택 가능합니다.
- 3 분석 대상의 인스턴스 및 일자가 표시되는 부분으로, **최대 6일** 까지 설정 가능합니다.
- 4 분석 지표의 수(Default: 5개) 및 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택할 수 있습니다.
- 5 위의 설정한 조건으로 **Trend Comparison 실행합니다.**

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

↳ Data Visualization ▶ Hotspot View

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

↳ Stat Analysis 연계

▪ TOP SQL 정보 확인

↳ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

↳ TOP Event ▶ SQL Detail 연계



- ✓ 장애가 발생한 시점인 11월 매주 화요일 (17.11.07, 17.11.14, 17.11.21, 17.11.28) 지표의 Trend 추이 변화를 확인합니다.
- ✓ CPU 사용률 및 Active Session 수 추이를 각 날자 별로 비교해 보면 02:00 ~ 05:00 사이에 4일 모두 비슷한 추이를 볼 수 있습니다.

Trend Comparison의 활용

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

└ Data Visualization ▶ Hotspot View

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

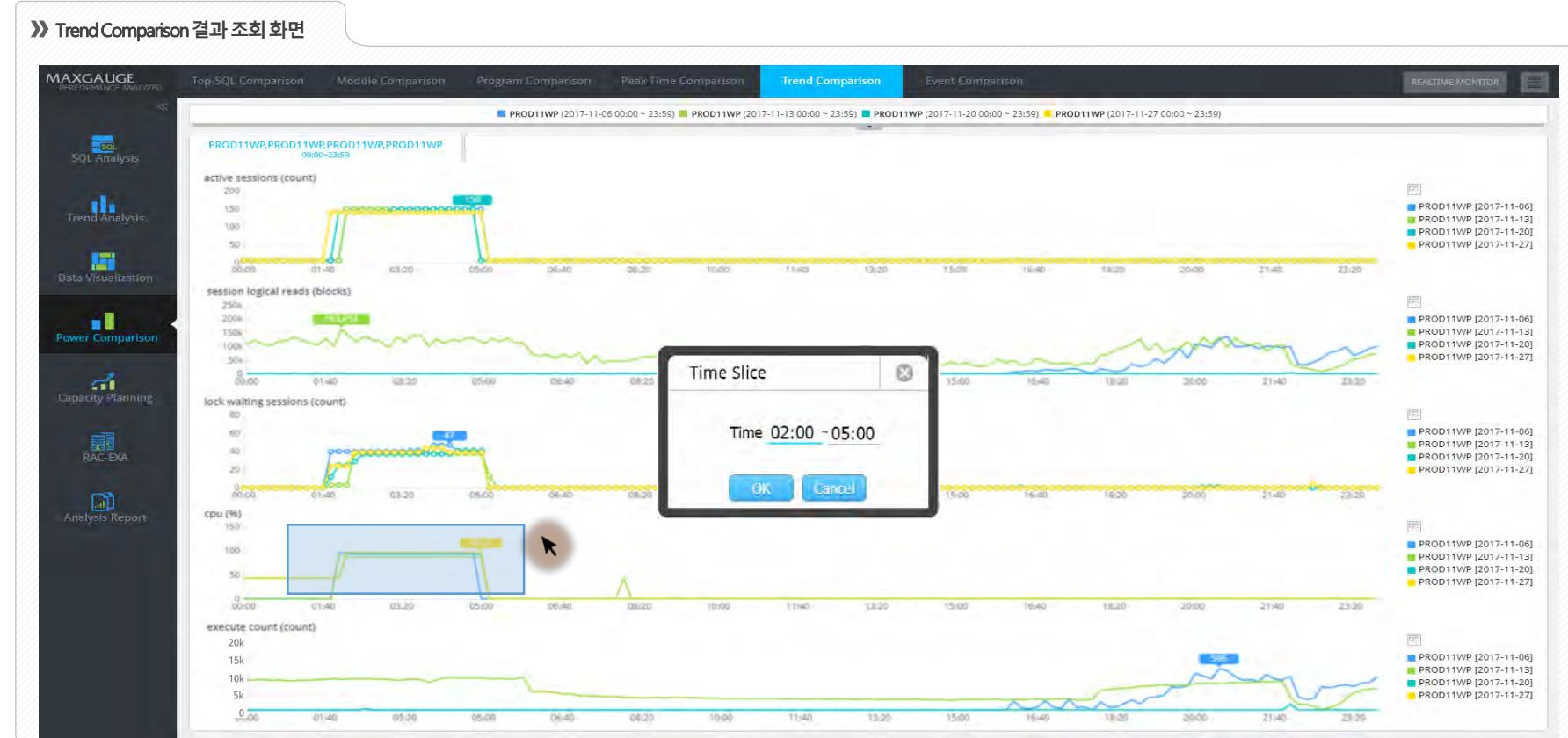
└ Stat Analysis 연계

▪ TOP SQL 정보 확인

└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계



✓ 장애 발생 구간(02:00 ~ 05:00)에 대한 상세 분석을 위해, 분석 구간을 마우스로 Drag 후 시간 설정하여 "OK" 버튼을 클릭합니다.

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

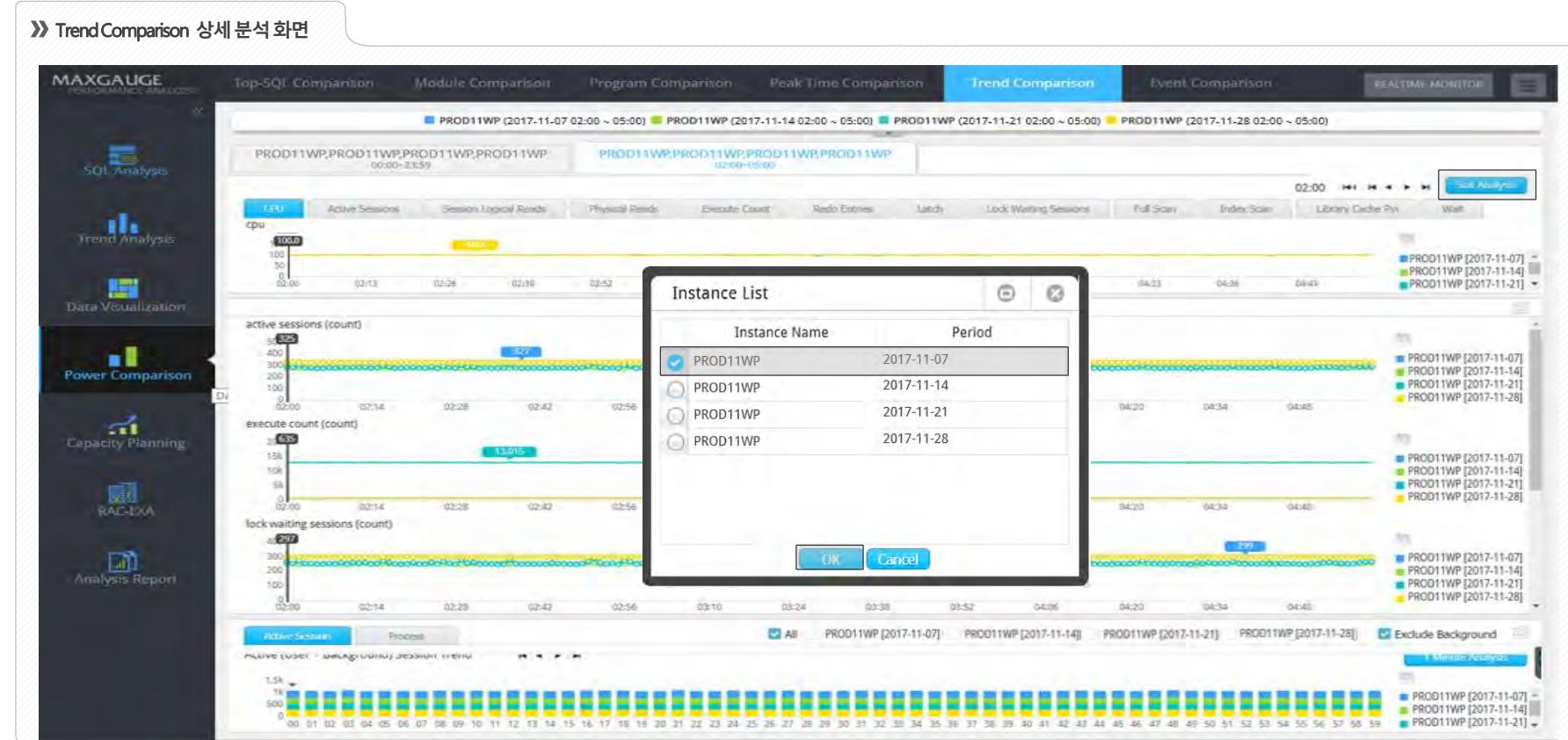
└ Stat Analysis 연계

▪ TOP SQL 정보 확인

└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계



✓ 장애 발생 구간(02:00 ~ 05:00)에 대한 상세 분석을 위해 Stat Analysis 를 클릭하여 인스턴스 명, 날짜를 선택합니다.

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

└ Stat Analysis 연계

▪ TOP SQL 정보 확인

└ TOP-N ▷ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▷ SQL Detail 연계

» Performance Trend(Stat) 화면

The screenshot shows the MAXGAUGE Performance Trend(Stat) interface. The main area displays a table of top SQL queries with columns for SQL ID, Schema, Program, Module, SQL Plan Hash, Elapsed Time (%), CPU Time (%), Logical Reads (%), Physical Reads (%), and Executions. Two rows are highlighted in red: 'gzhkw1qu6fwxm' and 'gzhkw1qu6fwxm'. Below this, a detailed view for the first query is shown with sections for Stat Name (AVG, SUM), Wait Time (buffer busy waits, cursor: pin S, etc.), and Execution Plan (SQL code: INSERT INTO LOGON (LOGON_ID, CUSTOMER_ID, LOGON_DATE) VALUES...). A callout '1' points to the top SQL table, and '2' points to the detailed view.

1 Stat Analysis를 통해, 분석 구간 내의 (2017.11.07 02:00 ~ 05:00)의 TOP-N 정보 관련 SQL 목록을 보여줍니다.

» Elapsed Time 및 Executions 기준으로 TOP 1-3 SQL문이 동일한 SQL문임을 알 수 있습니다.

2 추출 된 TOP SQL 중 선택 시, 2번 영역에 해당 SQL에 대한 STAT 정보 등 상세한 정보를 확인할 수 있습니다.

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

└ Stat Analysis 연계

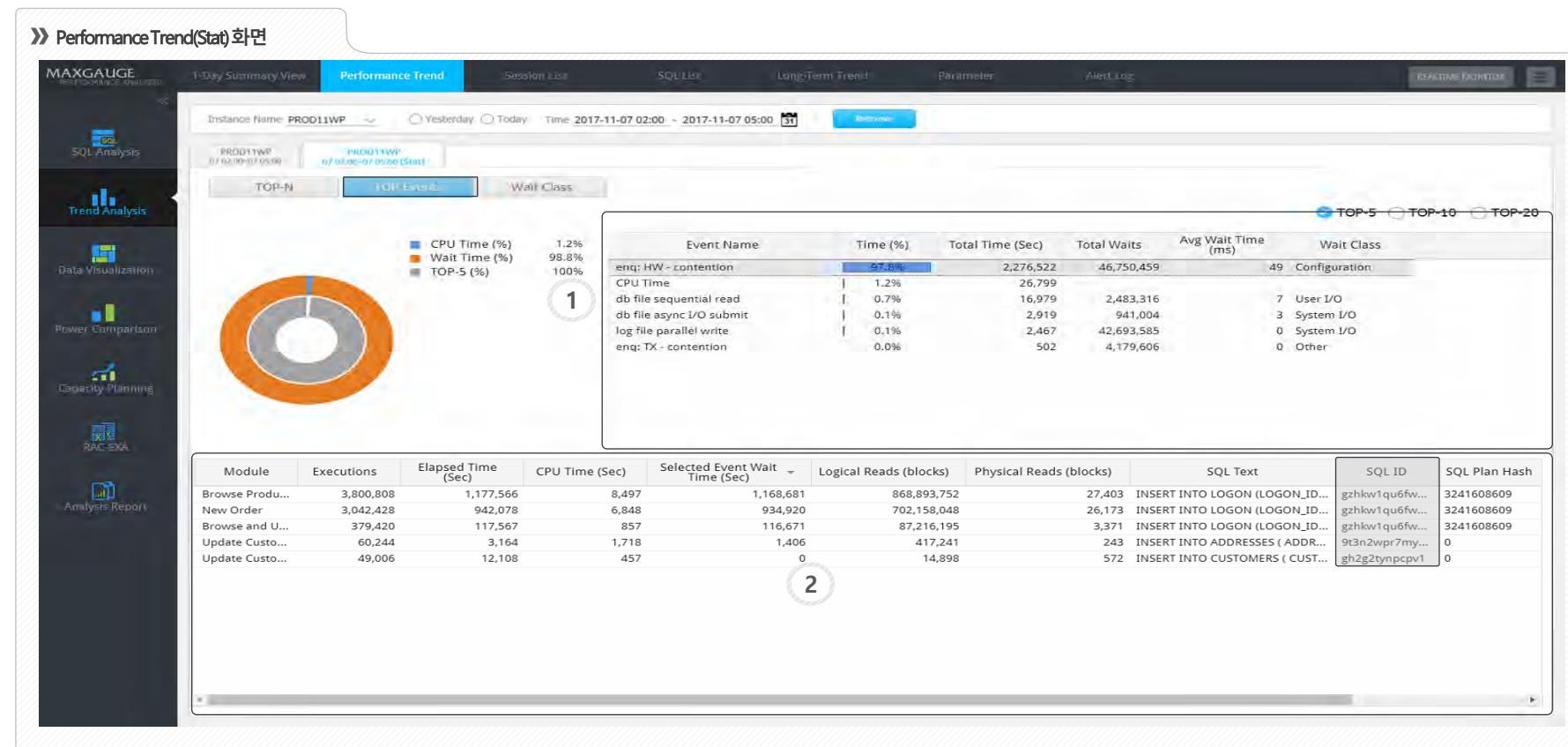
▪ TOP SQL 정보 확인

└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계

" 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! "



1 분석 구간 내의 (2017.11.07 02:00 ~ 05:00)의 TOP Events 를 상세하게 확인할 수 있습니다.

» Wait Time 비중이 높으며, 그 중 대부분 enq: HW – contention 대기 이벤트를 대기하고 있습니다.

2 추출 된 TOP Events 중 대기 이벤트 선택 시, 2번 영역에 해당 대기 이벤트를 대기한 SQL 목록을 보여줍니다.

» 또한, 해당 이벤트를 대기하는 SQL이 TOP SQL과 동일한 것을 알 수 있습니다.

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

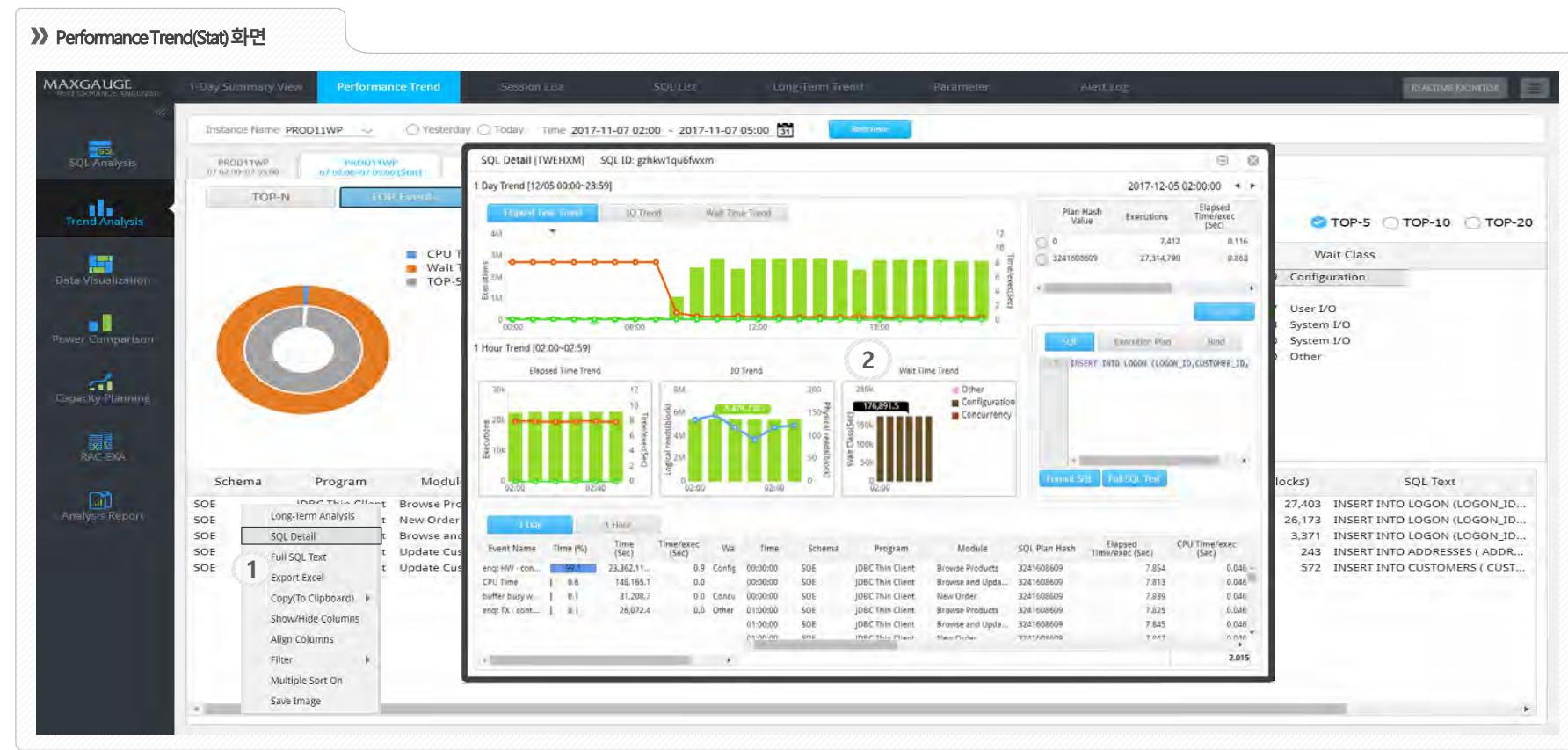
└ Stat Analysis 연계

▪ TOP SQL 정보 확인

└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계



1 선택한 SQL의 자세한 정보를 확인하기 위해 마우스 오른쪽 클릭 후, SQL Detail을 선택합니다.

2 SQL Detail을 통해 선택한 SQL의 상세한 정보를 확인할 수 있습니다.

» 해당 SQL의 시간 별 수행 정보를 통해 성능 지연의 원인을 파악할 수 있습니다.

“ 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! ”

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

└ Stat Analysis 연계

▪ TOP SQL 정보 확인

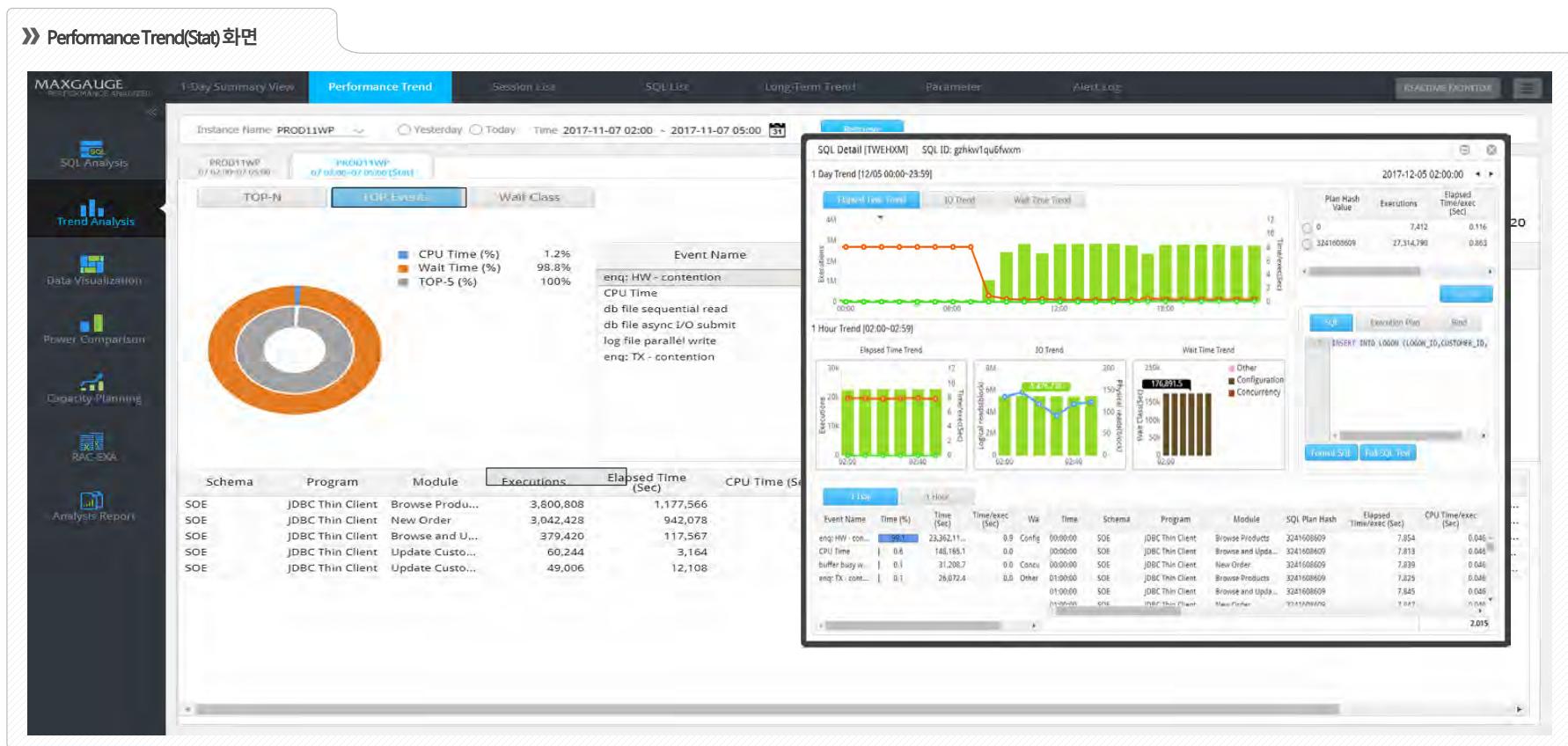
└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계

2

1



1 선택한 SQL의 자세한 정보를 확인하기 위해 마우스 오른쪽 클릭 후, SQL Detail을 선택합니다.

2 SQL Detail을 통해 선택한 SQL의 상세한 정보를 확인할 수 있습니다.

» 해당 SQL의 시간 별 수행 정보를 통해 성능 지연의 원인을 파악할 수 있습니다.

STEP

1. 장애 구간 파악

2. 검색 조건 설정

3. 데이터 분석

▪ 매주 같은 요일 Trend 비교

▪ 상세 Trend 비교

└ Stat Analysis 연계

▪ TOP SQL 정보 확인

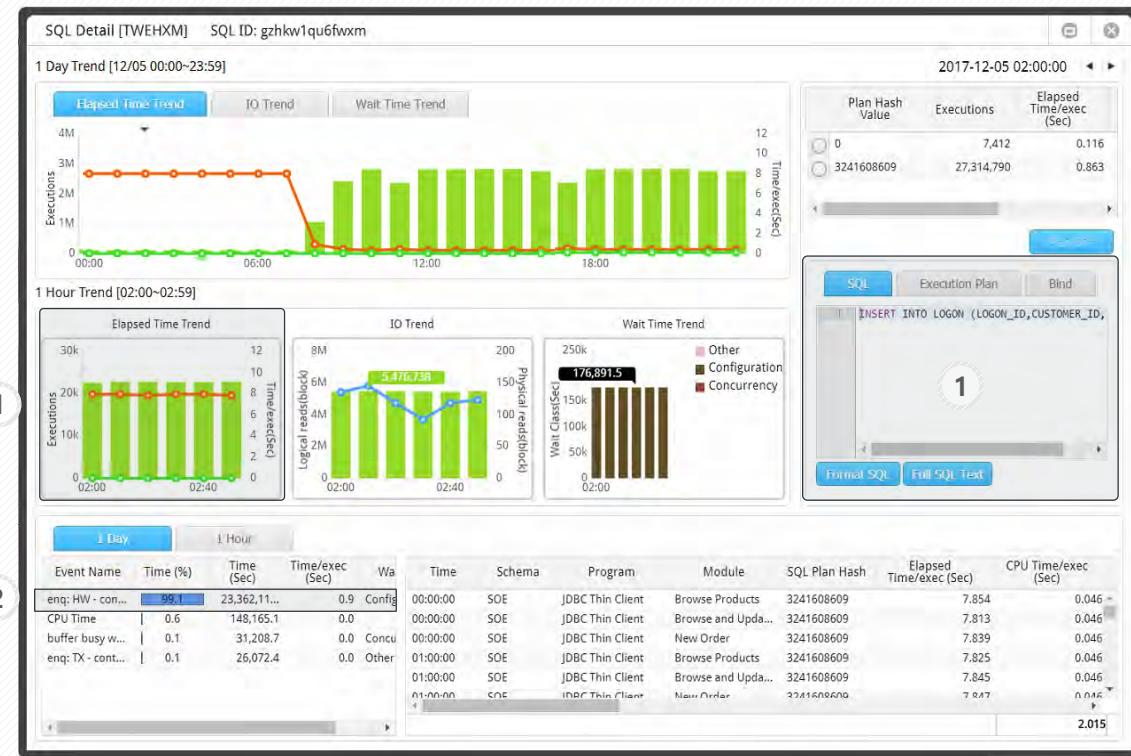
└ TOP-N ▶ SQL

▪ TOP Event 및 문제 SQL 확인

└ TOP Event ▶ SQL Detail 연계

" 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! "

» Performance Trend(Stat) 화면



1 장애 현상을 유발하는 SQL은 **INSERT 문**으로, 장애 구간에 시간 당 약 2천만번 이상 수행되는 것을 확인할 수 있습니다.

2 동시에 여러 세션에서 대량의 INSERT문이 수행되면서, **enq:HW- contention 대기** 이벤트를 대기하게 됩니다.

» 따라서, **EXTENT SIZE**를 확인하여, **적절한 크기로 조정**하면 HWM 경합 문제를 줄여 해당 현상을 방지할 수 있습니다.

MAXGAUGE

Practical Guide

Capacity Planning [Performance Analyzer]

Contents

Capacity planning?

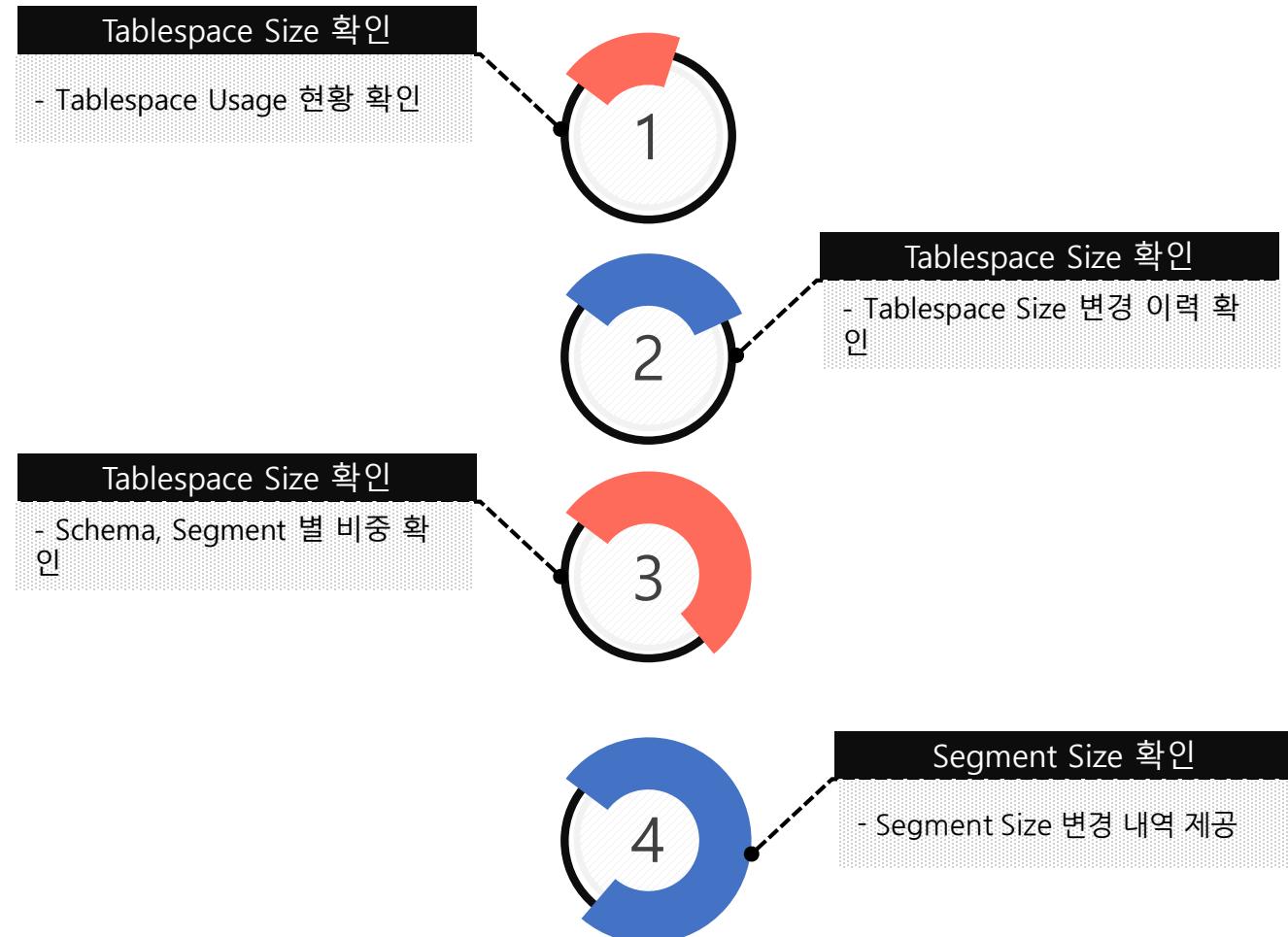
Capacity planning 의 활용

Case 1. TABLESPACE ALARM이 발생 했는데 원인을 알고 싶어요

Case 2. 특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요

Capacity Planning?

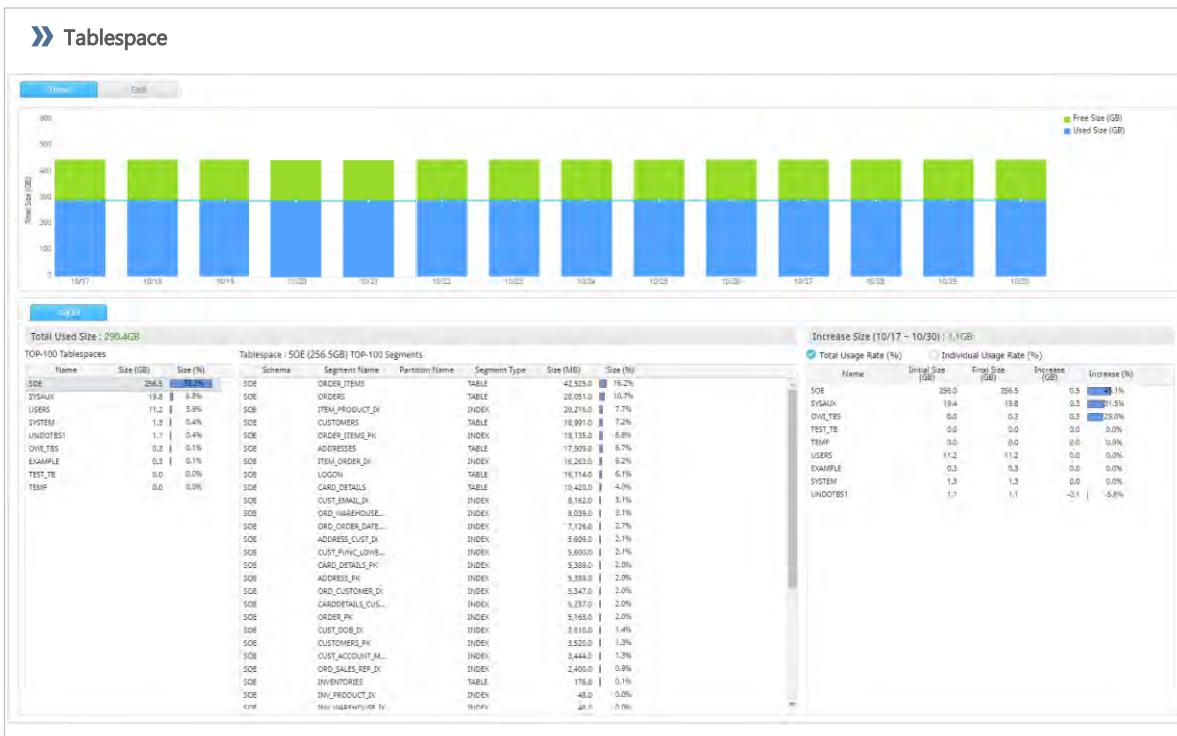
Capacity Planning의 주요 기능은 무엇인가요?



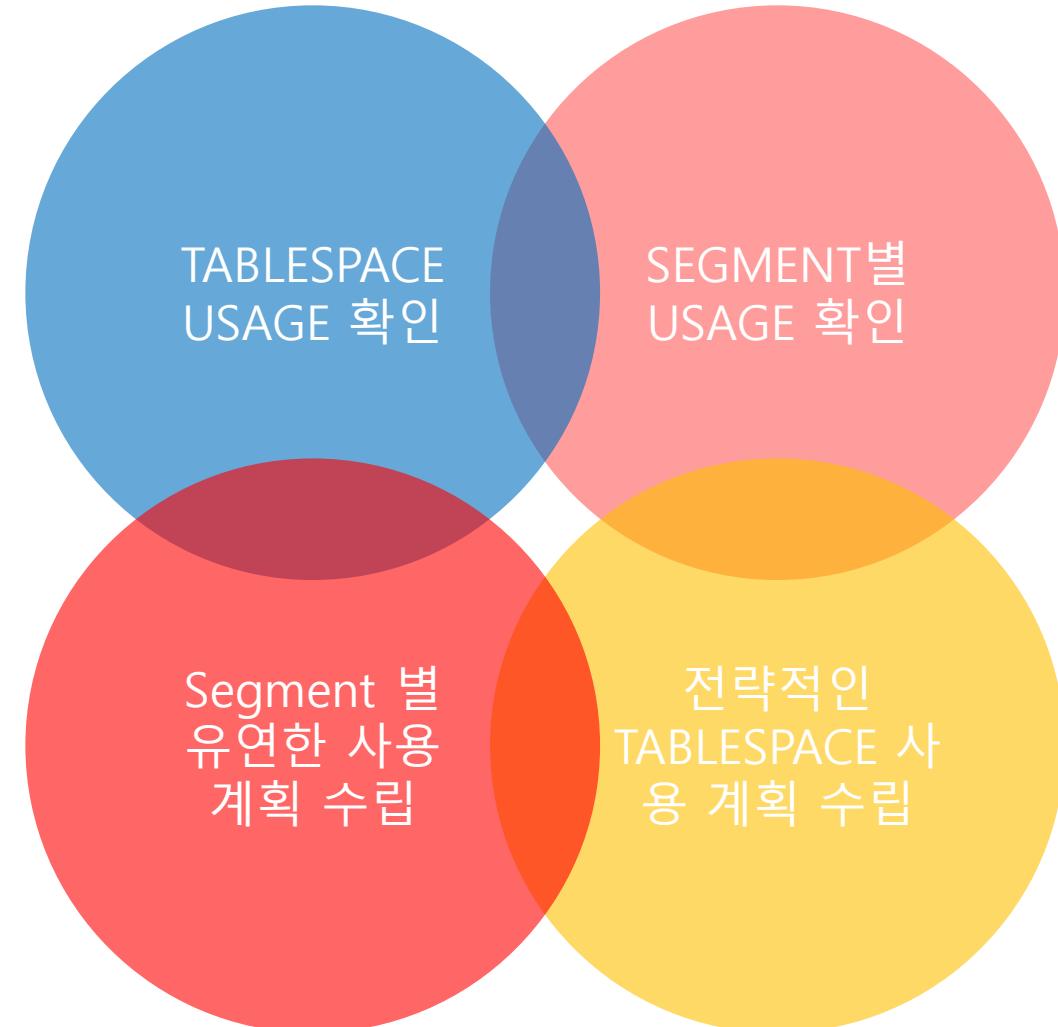
Capacity Planning?

Capacity Planning

- 테이블스페이스의 사용량 및 여유 공간에 대한 추이를 제공한다.
- 사용량 증가 비율이 높은 테이블 스페이스 내의 Top-N 세그먼트 정보를 제공한다.
- 세그먼트에 대한 일별 크기 및 증가 크기를 제공한다.
- 테이블스페이스와 세그먼트 정보를 일자 별 Trend와 Grid로 제공하기 때문에 효율적인 용량관리가 가능하다.
- 해당 기능은 Performance Analyzer > Capacity Planning > Tablespace 경로를 통해 사용할 수 있다.



안정적인 TABLESPACE 관리가 가능합니다!



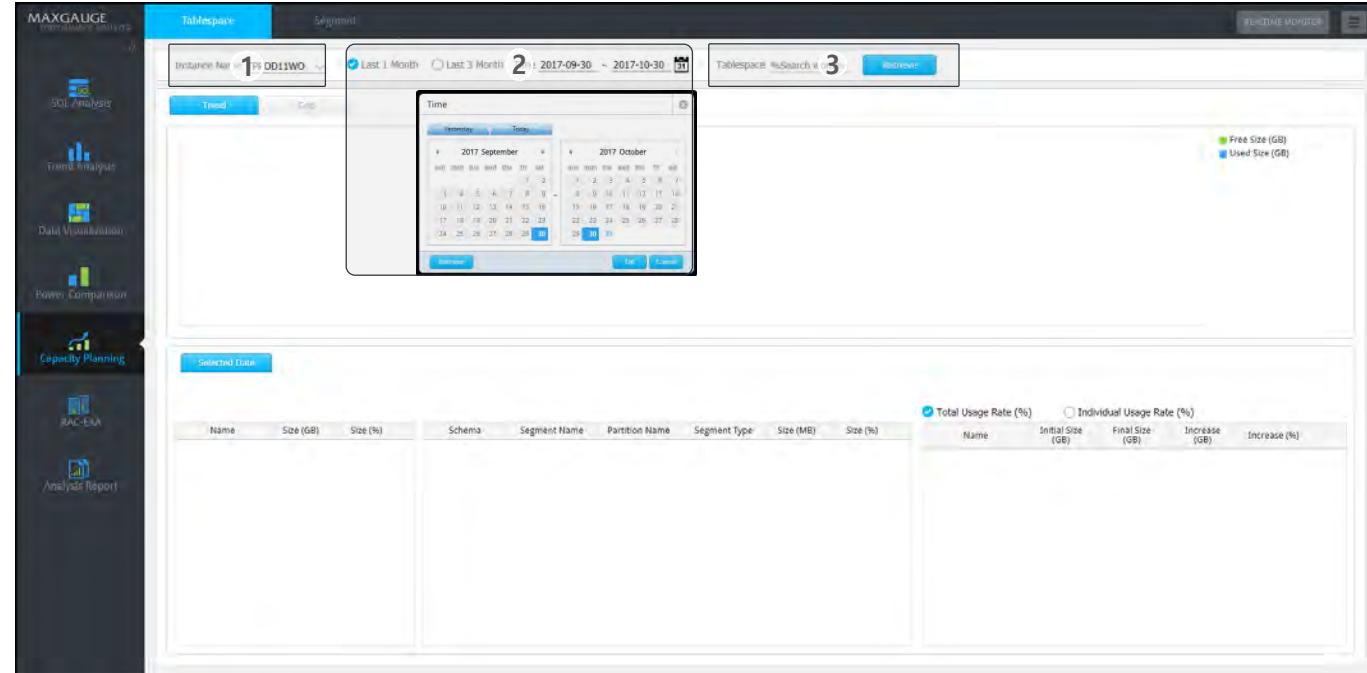
Capacity Planning의 활용

Capacity Planning의 사용 방법 (Tablespace)

Capacity Planning (Tablespace-Trend)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Tablespace-Trend 검색 화면



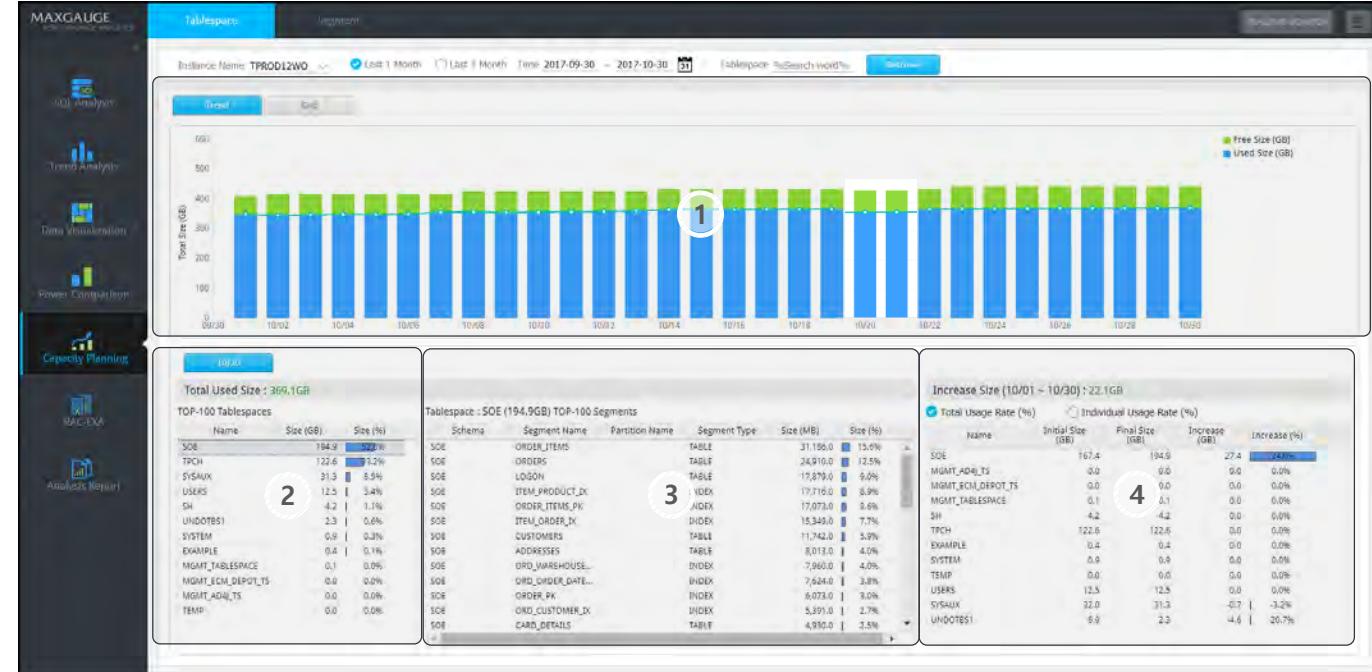
- 1 대상 Instance를 선택합니다.
- 2 Last 1 Month, Last 3 Month로 기간을 선택 할 수 있으며, 달력 UI를 이용하여 일(Day) 단위로 기간을 선택 할 수 있습니다.
- 3 찾고자 하는 Tablespace의 Full Name 또는 일부를 조건에 넣을 수 있습니다.

Capacity Planning의 사용 방법 (Tablespace)

Capacity Planning (Tablespace-Trend)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Tablespace-Trend 조회 화면



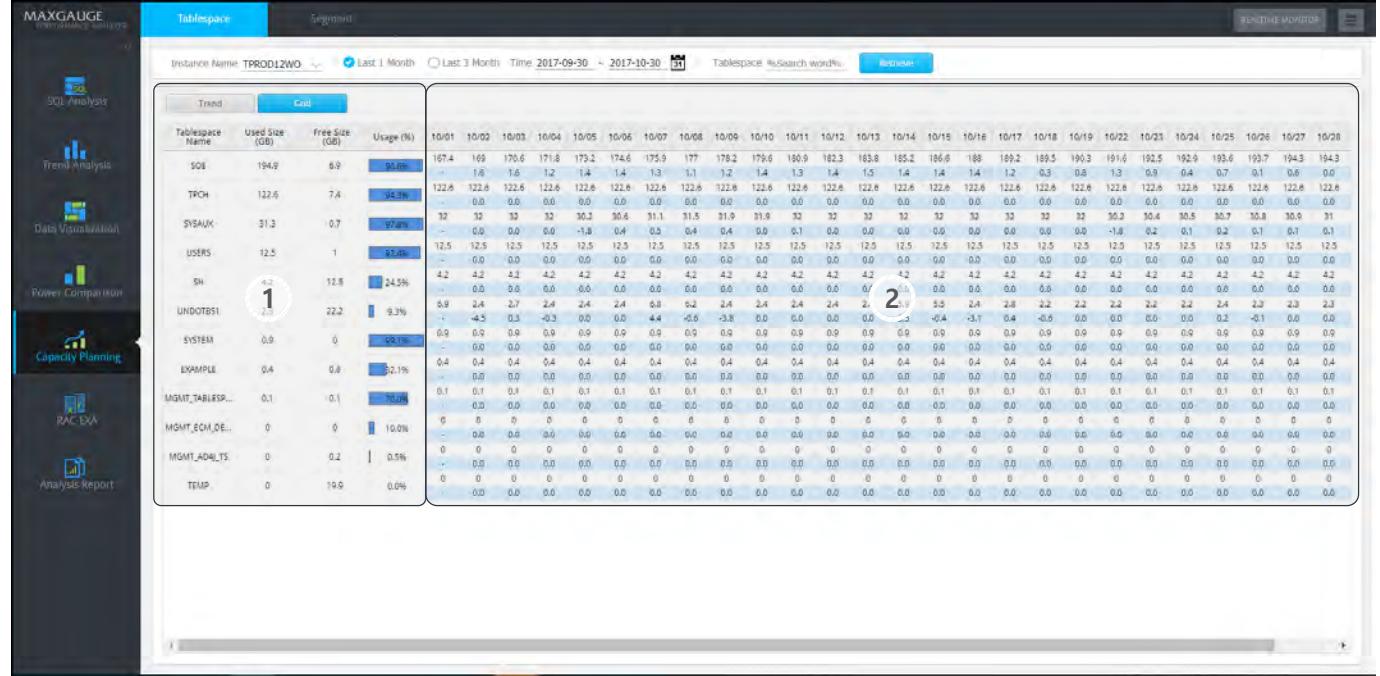
- 1 지정한 기간의 모든 Tablespace의 Free Size(SUM)와 Used Size(SUM)을 일자별 그래프로 제공합니다.
 - “█” 부분은 Free Size , “█” 부분은 Used Size를 의미합니다.
- 2 지정한 기간의 마지막 일자의 Top-100 Tablespace 정보를 Size기준으로 제공 합니다.
 - Size(GB) : 해당 Tablespace의 사용량
 - Size(%) : 전체 Tablespace 사용량 중 해당 Tablespace가 차지하는 비율
- 3 Top-100 Tablespace에서 선택한 Tablespace에 속해 있는 Segment를 Size기준으로 Top-100 Segment 정보를 제공합니다.
- 4 Tablespace의 증가율 정보를 제공합니다.
 - Total Usage Rate : 사용량이 가장 크게 증가한 순으로 정렬하여 정보를 제공합니다.
 - Individual Usage Rate : 사용량 비율이 가장 크게 증가한 순으로 정렬하여 정보를 제공합니다.

Capacity Planning의 사용 방법 (Tablespace)

Capacity Planning (Tablespace-Grid)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Tablespace-Grid 조회



1 지정한 기간 마지막 일자의 Tablespace 상세 정보를 제공 합니다.

- Used Size(GB) : 마지막 날짜의 Tablespace 사용 공간입니다.
- Free Size(GB) : 마지막 날짜의 Tablespace 여유 공간입니다.
- Usage(%) : Tablespace의 사용률 ($\text{FreeSize}/\text{UsedSize} * 100$) 입니다.

2 Tablespace의 일자별 Size 변화 추이를 제공합니다.

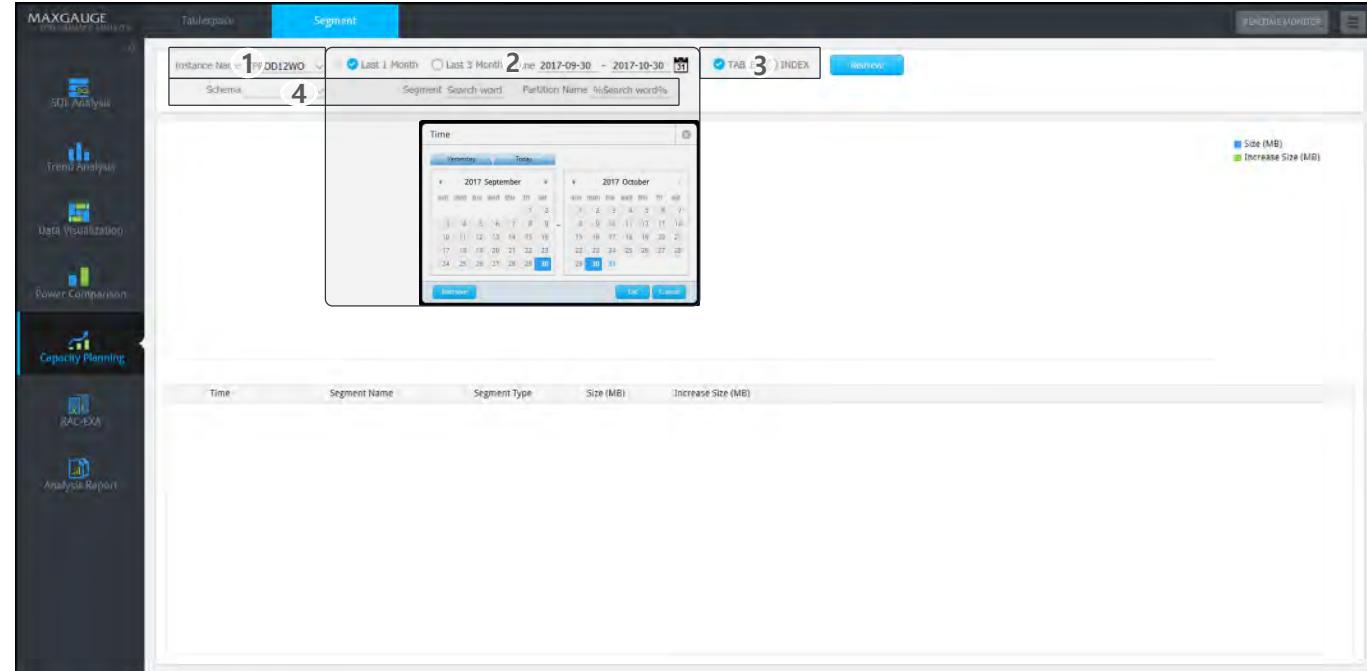
- “” 부분은 일자별 증가 추이를 의미합니다.
- “” 부분은 일자별 Tablespace Size를 의미합니다.

Capacity Planning의 사용 방법 (Segment)

Capacity Planning (Segment)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Segment 검색 화면



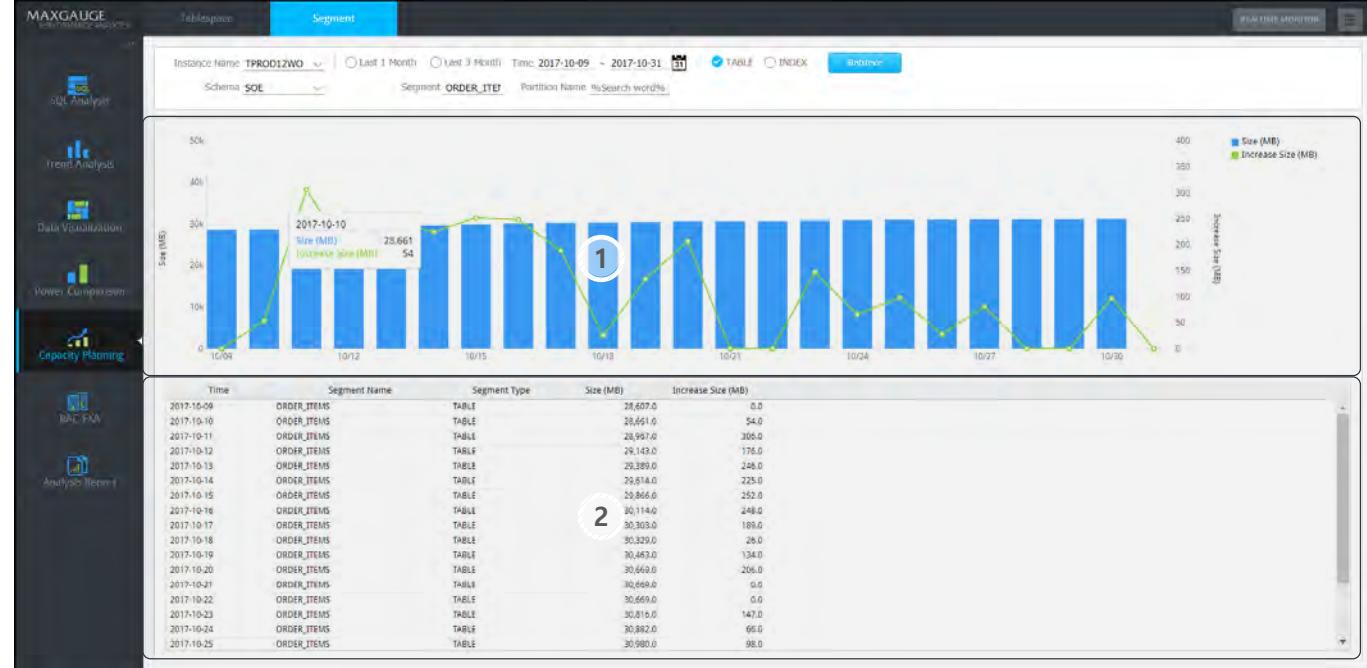
- 1 대상 **Instance**를 선택합니다.
- 2 **Last 1 Month, Last 3 Month**로 기간을 선택 할 수 있으며, 달력 UI를 이용하여 **일(Day)** 단위로 기간을 선택 할 수 있습니다.
- 3 Segment 유형을 **TABLE,INDEX** 중 선택 할 수 있습니다.
- 4 **Schema**와 **Segment**는 필수 적으로 선택 및 입력해야 하며, Partition Name은 특정 Partition을 찾고자 할때 입력 하면 됩니다.

Capacity Planning의 사용 방법 (Segment)

Capacity Planning (Segment)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Segment 조회 화면



1 선택한 Segment에 대한 일별 크기 변화를 그래프로 제공합니다.

- "0 → 0" 부분은 Increase size(MB)를 의미 합니다.
- "■" 부분은 Size(MB)를 의미합니다.

2 지정된 기간에 대해서 일자별로 Size(MB),Increase Size(MB) 정보를 제공합니다.

- Time : 선택한 기간의 일(Day)자를 의미 합니다.
- Segment Name : 선택한 Segment의 Name을 의미 합니다.
- Segment Type : 선택한 Segment의 Type을 의미합니다.
- Size(MB) : 일별 Segment의 Size를 의미 합니다.
- Increase Size(MB) : 일별 증가한 Segment의 Size를 의미 합니다.

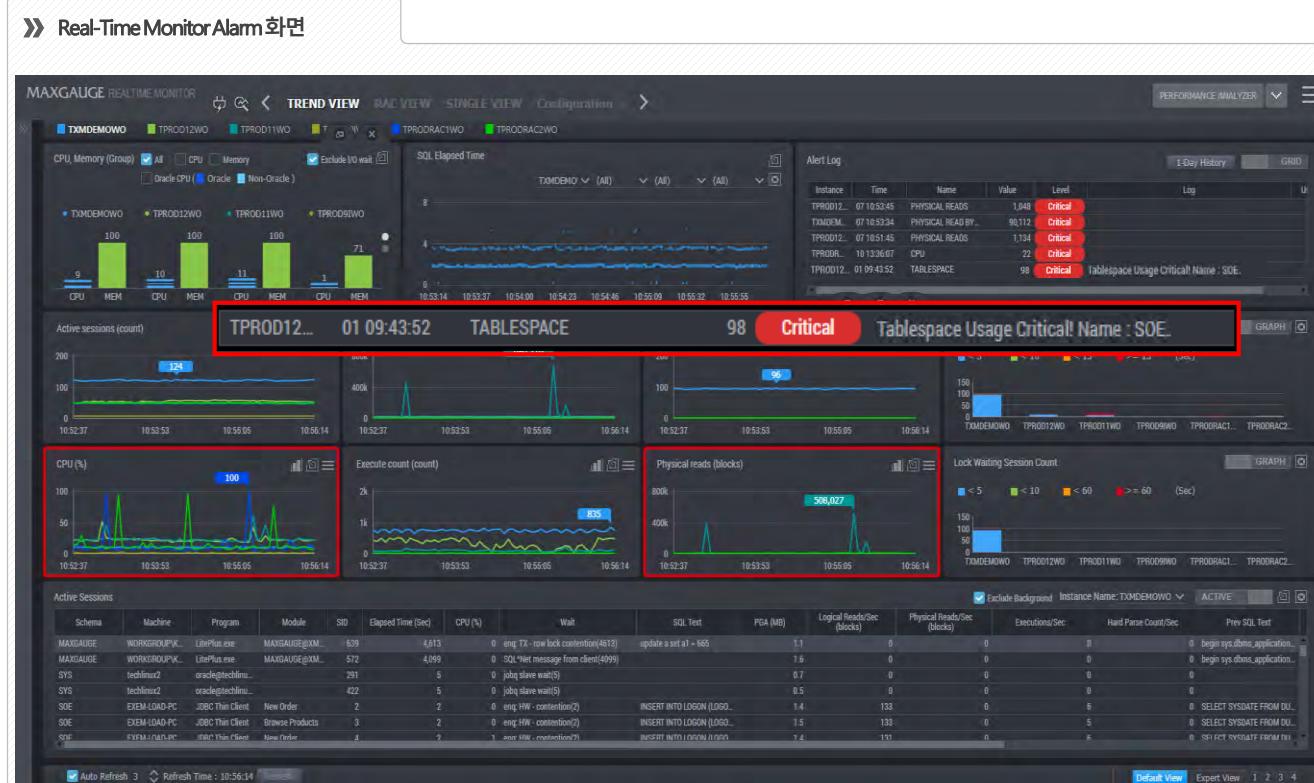
실전 분석 사례 Case 1.
**TABLESPACE ALARM이 발생 했는데 원인을
알고 싶어요.**

STEP

1. 상황 발생 및 검색 조건 설정

- 상황발생
 - 검색 조건 설정
2. 데이터 분석
- 출력 데이터 확인
 - Top-100 Segments 확인

"TABLESPACE ALARM이 발생했는데 원인을 알고 싶어요"



✓ 상황 발생

INSTANCE : TPROD12WO
 ALARM : Tablespace Usage CRITICAL
 TABLESPACE : SOE
 ALARM DATE : 2017년 10월 16일

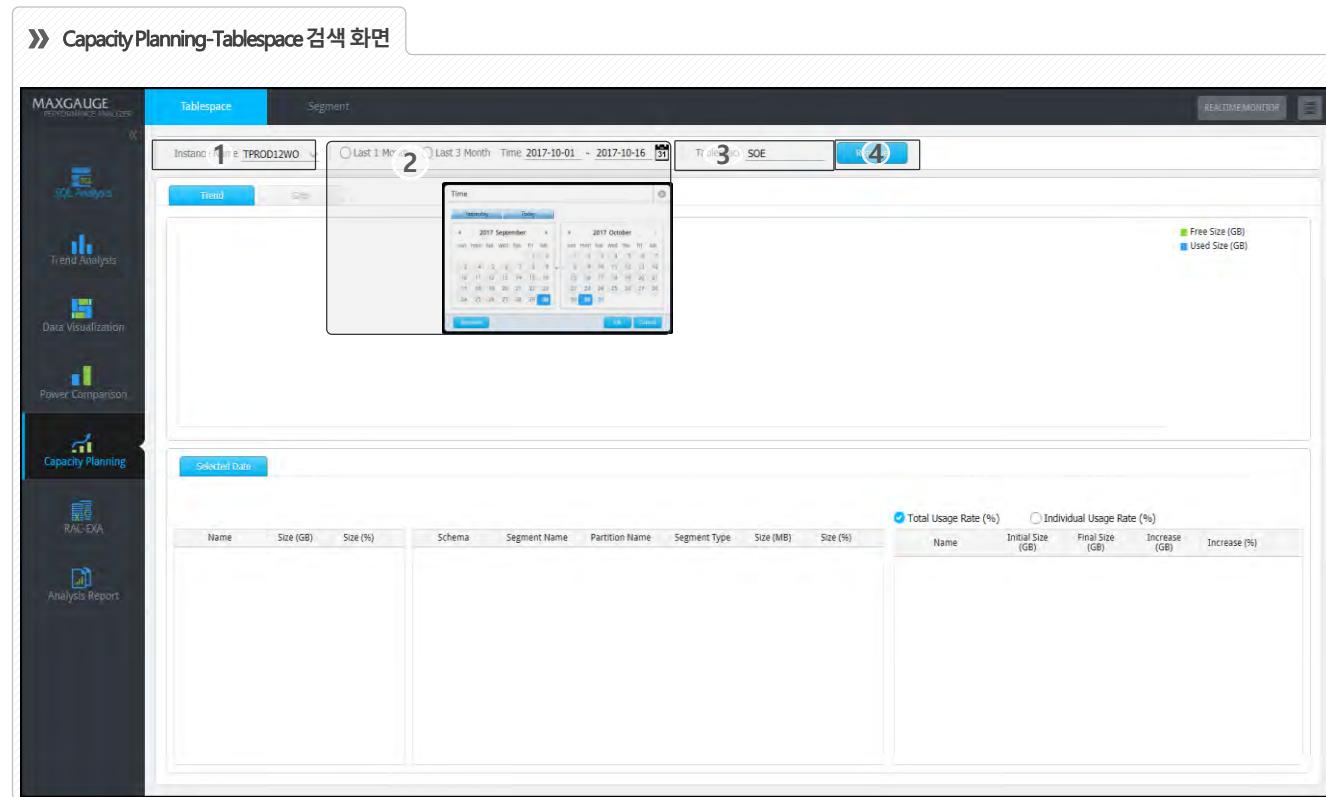
- ✓ CRITICAL ALARM이 발생한 INSTANCE NAME, TABLESPACE NAME을 확인 합니다.
- ✓ 발생한 ALARM을 확인해 보면 INSTANCE NAME : "TPROD12WO" / TABLESPACE NAME : "SOE"를 확인 할 수 있습니다.

“TABLESPACE ALARM이 발생했는데 원인을 알고 싶어요”

STEP

1. 상황 발생 및 검색 조건 설정

- 상황발생
 - 검색 조건 설정
2. 데이터 분석
- 출력 데이터 확인
 - Top-100 Segments 확인



- 1 대상 Instance (**TPROD12WO**)를 선택합니다.
- 2 Last 1 Month, Last 3 Month, 일(Day) 단위로 기간을 선택 할 수 있습니다
- 3 Critical Alarm의 대상인 **SOE**(Tablespace Name)를 Tablespace란에 입력 합니다.
- 4 위의 설정한 조건으로 **Tablespace**를 조회합니다.

✓ 시나리오

INSTANCE	: TPROD12WO
ALARM	: Tablespace Usage CRITICAL
TABLESPACE	: SOE
ALARM DATE	: 2017년 10월 16일
DATE	: 2017년 10월 01일 ~ 10월 16일

✓ 목표

CRITICAL ALARM이 발생한 TABLESPACE의 증가 추이를 파악하고, 해당 기간에 SIZE가 가장 많이 증가한 시점과 Segment를 찾기

“TABLESPACE ALARM이 발생했는데 원인을 알고 싶어요”

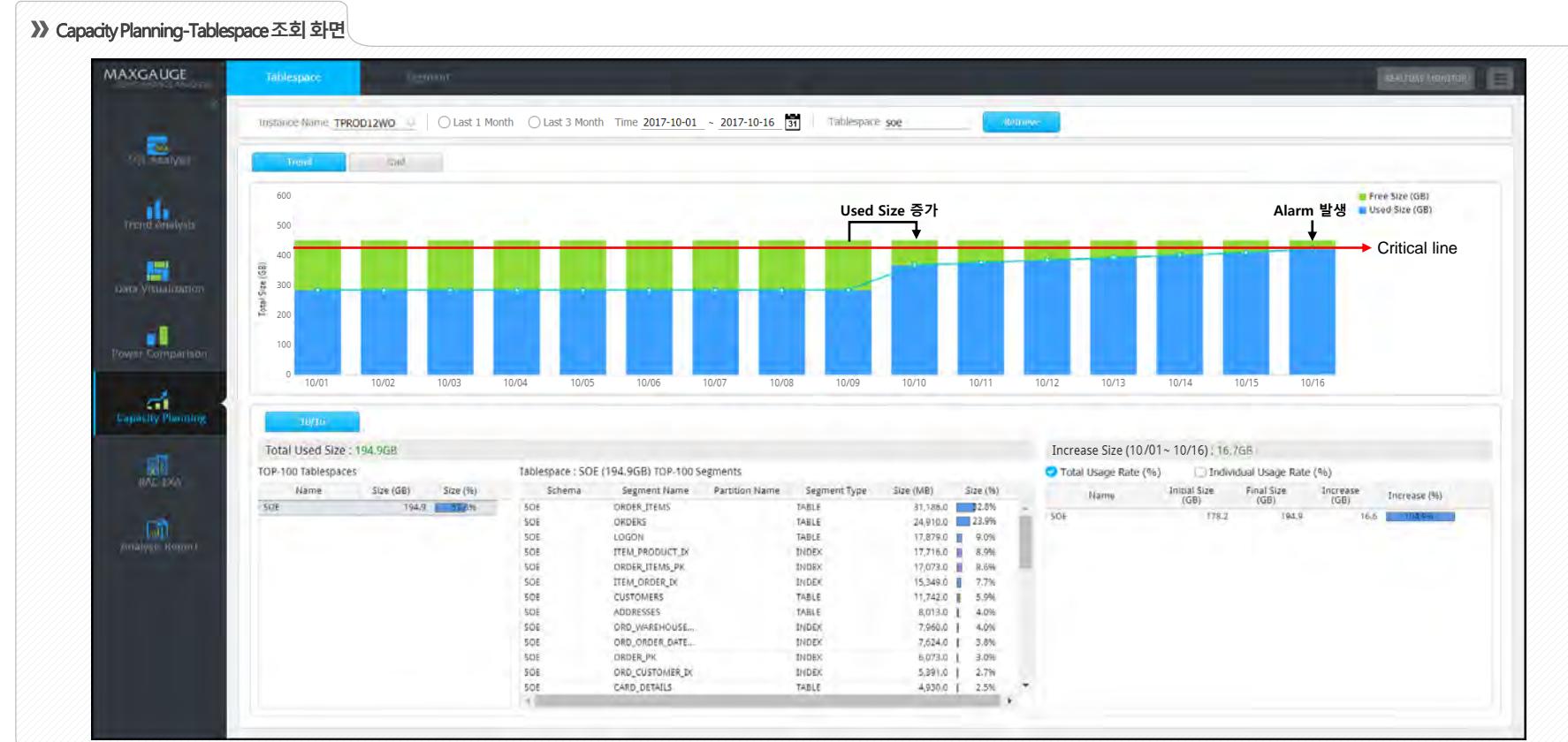
STEP

1. 상황 발생 및 검색 조건 설정

- 상황발생
- 검색 조건 설정

2. 데이터 분석

- 출력 데이터 확인
- Top-100 Segments 확인



- ✓ 막대 그래프를 통하여 일자 별 Size 증가 추이를 확인 합니다.
- ✓ “SOE” Tablespace 사용량이 10월 9일 이후로 점차 증가하여 **10월 16일에 “Tablespace Usage Critical” Alarm**이 발생한 것을 확인할 수 있습니다.
- ✓ **Used Size**가 급격히 증가했던 날짜는 **10월 10일**이므로, 10월 9일과 사용량을 비교하여 원인을 파악할 필요가 있습니다.
- ✓ Top-100 Segments에는 선택한 날짜의 **Segment Size** 정보를 상세히 제공 하므로 10월 9일과 **10월 10일**의 해당 Segment를 비교하여 파악할 필요가 있습니다.

“TABLESPACE ALARM이 발생했는데 원인을 알고 싶어요”

STEP

1. 상황 발생 및 검색 조건 설정

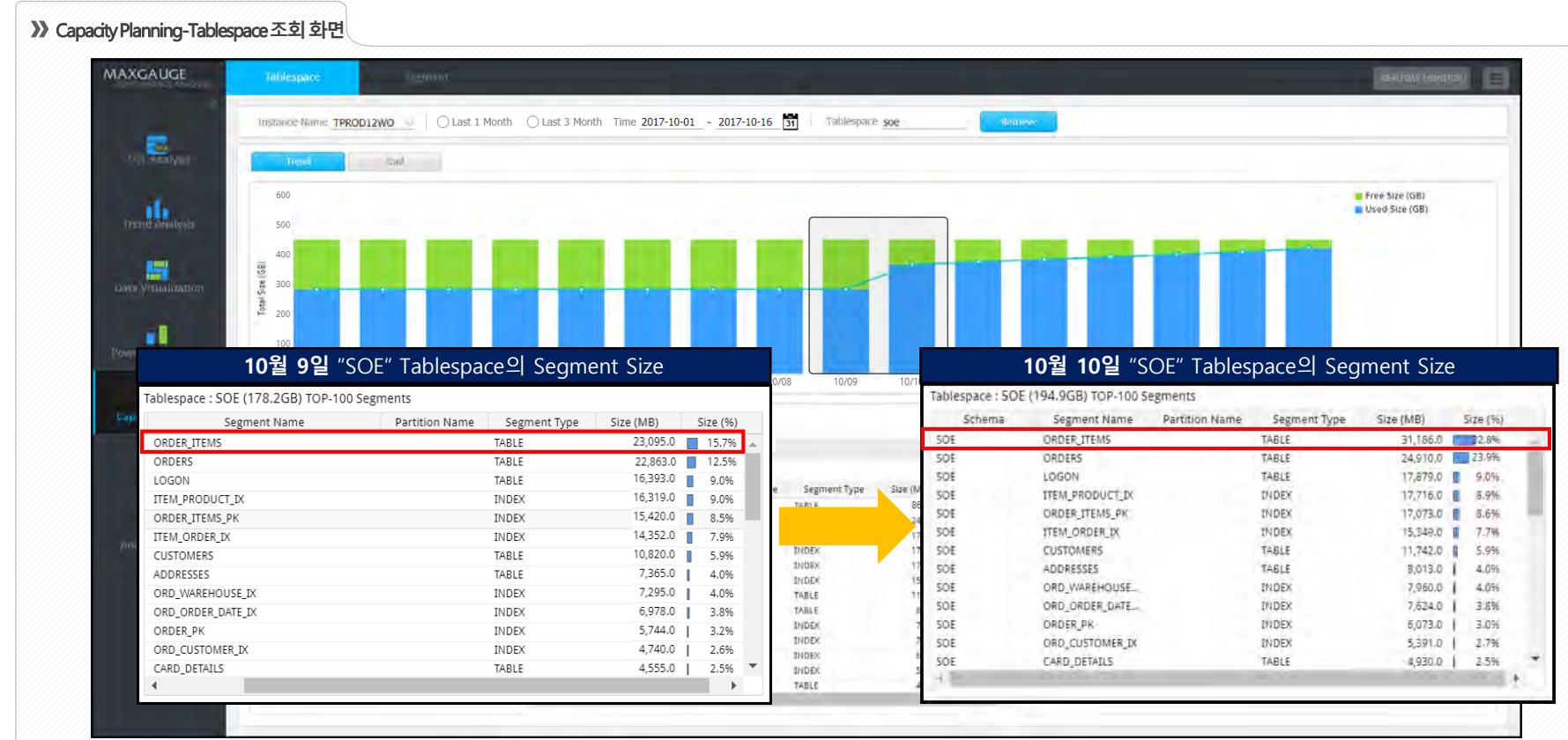
▪ 상황발생

▪ 검색 조건 설정

2. 데이터 분석

▪ 출력 데이터 확인

▪ Top-100 Segments 확인



- ✓ 중앙하단에 위치한 TOP-100 Segments를 살펴보면 Size가 증가율이 큰 순서대로 정렬 되어 있는 것을 볼 수 있습니다.
- ✓ 10월 9일과 10월 10일의 Top-100 Segments의 Size를 비교하면, “ORDER_ITEMS” TABLE이 23,095(MB) → 31,186(MB)로 증가했음을 알 수 있습니다.
- ✓ Size(%)도 마찬가지로 10월 9일과 10월 10일의 비중도는 15.7% → 22.8%로 증가했음을 알 수 있습니다.
- ✓ Size 증가에 가장 많은 영향을 준 Segment인 “ORDER_ITEMS”와 “SOE” Schema를 확인 할 수 있으며, 해당 Segment에 대한 증가 이력을 살펴 볼 필요가 있습니다.

실전 분석 사례 Case 2.
**특정 Segment Size가 크게 증가한 시점에
자주 수행된 SQL을 알고 싶어요.**

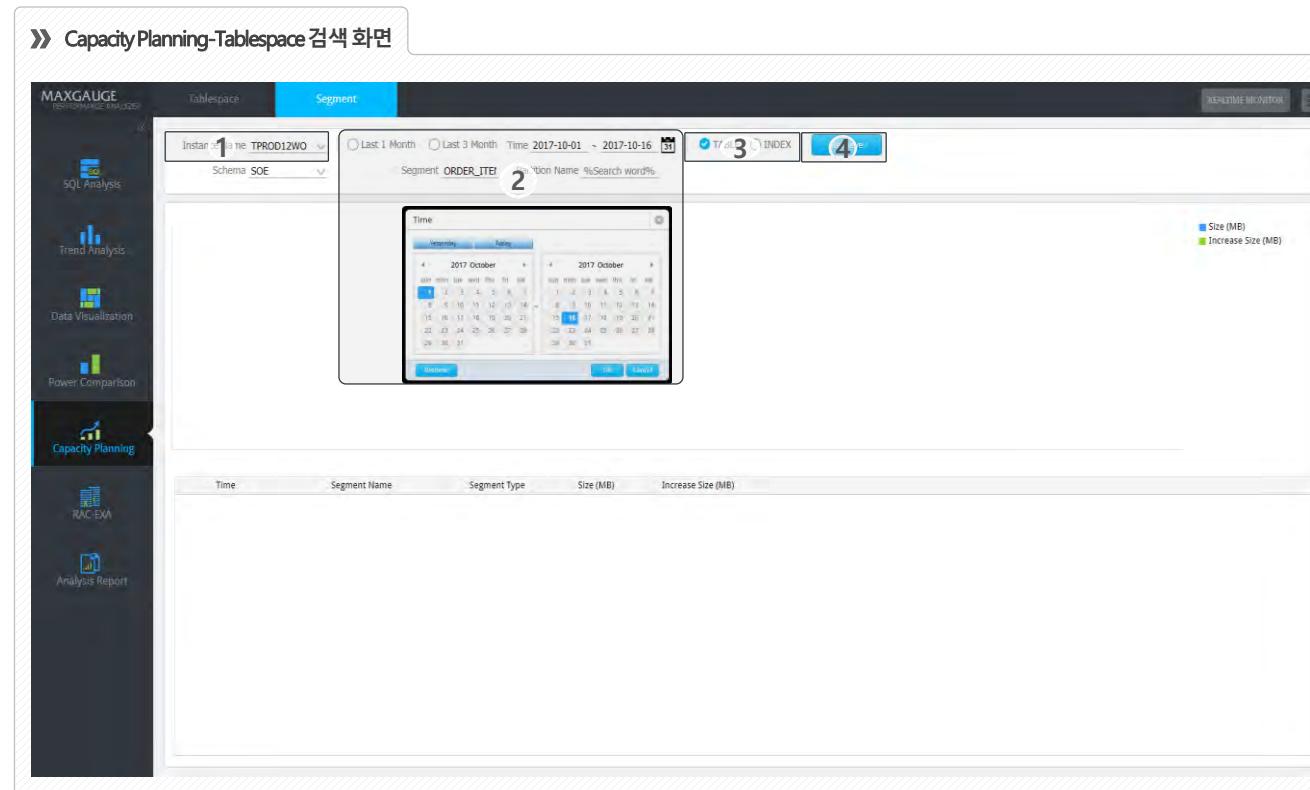
“특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요”

STEP

1. 상황 발생 및 검색 조건 설정

2. 데이터 분석

- Segment 조회
- 출력 데이터 확인
- SQL 정보 확인



- 1 대상 Instance (TPROD12WO)를 선택합니다.
- 2 Last 1 Month, Last 3 Month, 일(Day) 단위로 기간을 선택 할 수 있습니다
- 3 Segment Type을 선택 합니다.
- 4 Schema, Segment는 필 수 입력해야 하므로 입력 후 Segment 정보를 조회 합니다.

✓ 시나리오

INSTANCE : TPROD12WO
TABLESPACE : SOE
Date : 2017년 10월 01
~ 10월 16일

✓ 목표

Case1에서 Tablespace 증가에 큰 영향을 준 Segments를 찾았다.
이제 Segment Size 증가 이력을 살펴 보면서 급격히 증가한 시점을 찾고, 영향도가 높은 SQL을 추적하자.

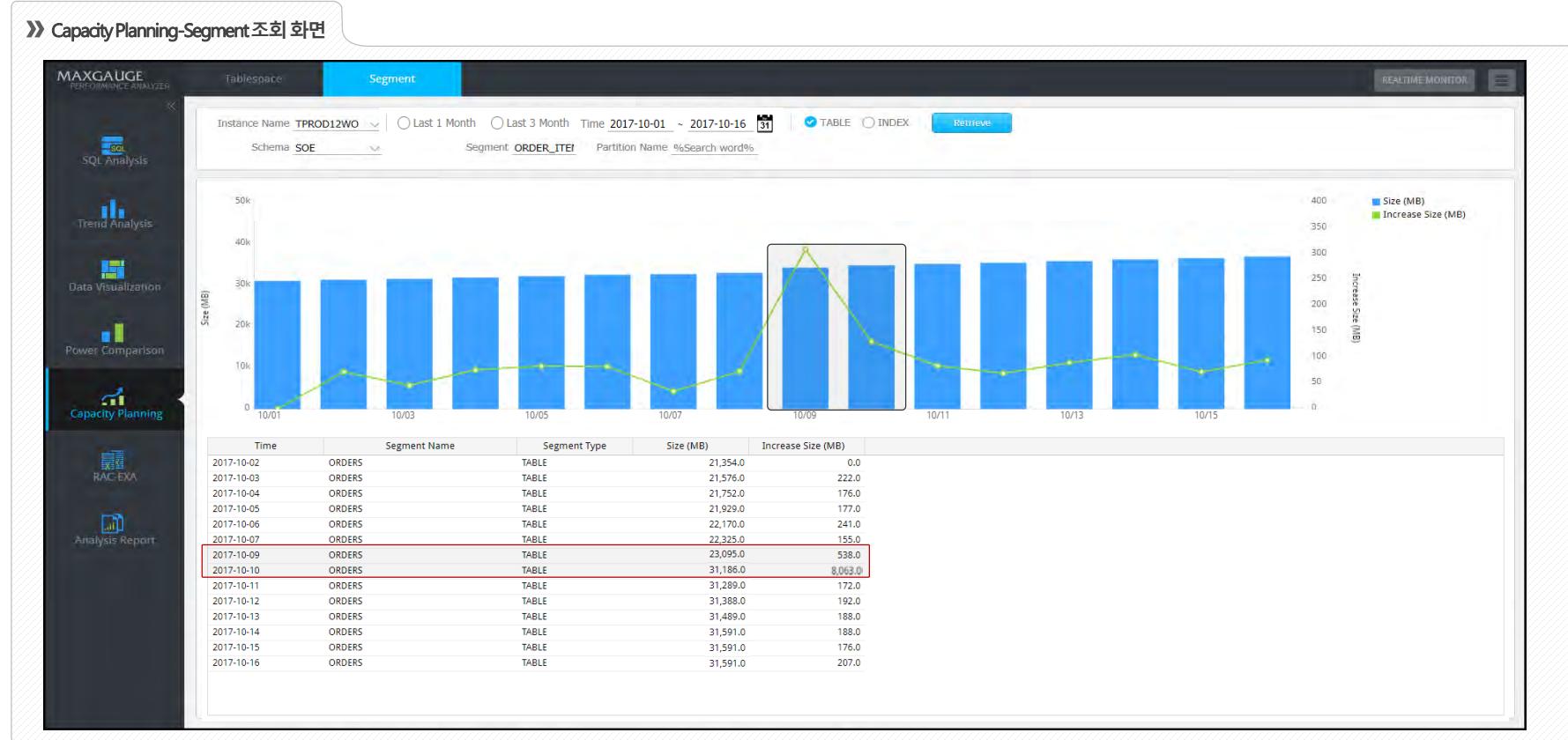
“특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요”

STEP

1. 상황 발생 및 검색 조건 설정

2. 데이터 분석

- Segment 조회
- 출력 데이터 확인
- SQL 정보 확인



- ✓ 막대 그래프를 통하여 일자별 Size 증가 추이를 확인 합니다.
- ✓ Increase Size를 살펴 보면 점차 Size가 증가하다가 10월 9일과 10월 10일 사이에 **8,063(MB)**가 급격히 증가한 것을 확인 할 수 있습니다.
- ✓ 10월 09일과 10월 10일 사이에 Size가 급격히 증가하는데 영향을 끼친 “SQL”이 무엇인지 확인 해 볼 필요가 있습니다.

“특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요”

STEP

1. 상황 발생 및 검색 조건 설정

2. 데이터 분석

- Segment 조회
- 출력 데이터 확인
- SQL 정보 확인

» Trend Analysis – SQL List 조회 화면

- ✓ Segment 탭을 통해 파악한 시점을 선택하고, SQL_TEXT에 Segment name을 넣고 조회 합니다.
- ✓ 10월 10일에 가장 수행 건수가 많은 SQL은 INSERT문이고, Execution은 **10분당 평균 12,000건** 정도로 매우 높습니다.
- ✓ 10월 10일에 **10분당 많은 INSERT문**이 수행 되었고, 해당 SQL로 인해 10월 9일 ~ 10월 10일 사이에 Size가 급격히 증가 했음을 알 수 있습니다.

MAXGAUGE



Practical Guide

Alert Script [Maxgauge Agent Sect]

Contents

Alert Script ?

Alert Script 의 활용

Case 1. Alert Script

평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요.

Case 2. Alert Script

업무시간에 Backup이 수행되는 것을 방지하기 위해
Alert을 등록하고 싶어요.

Script Alert ?

Script Alert 는 언제 쓰나요?



Alert
Script

- » 사용자가 별도로 사용하는 Script 를 통해 Alert을 등록할 수 있는 **Alert Script**

Script Alert 의 활용

RTS User Script Alert 의 사용 방법

RTS User Script Alert

- 동작 방법

» 화면구성

```
1. Listener Check Interval 설정
listener_check_interval=10
User Script Alert [scriptname],[interval],[normal value],[script full path]
scriptl=inval_obj,60,0,/maxgauge/TWEH12R2WP/util/rc.sh
#host[pc1-inval_obj],60,0,/maxgauge/TWEH12R2WP/util/rc.sh
#scriptl[inval_obj,60,0,/maxgauge/TWEH12R2WP/util/rc.sh]
```

사용자가 별도로 사용하는 Script를 통해 Alert을 등록할 수 있는 기능입니다.

1 각 옵션의 의미

- **script name** : script 이름 지정
- **interval** : script 수행 주기 지정
- **normal value** : 의미 없는 값이므로 수정하지 않습니다.
- **script full path** : 수행할 script가 존재하는 위치

2 등록한 Shell 의 결과 값 (마지막 1 byte) 이

- **0 : Normal**
- **1 : Warning**
- **2 : Critical**

으로 Real Time Monitor (Alert log) 프레임에 보여집니다.

실전 활용 사례 Case 1.

평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요.

“평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!”

Alert Script

- **Script 등록**
- **Script 모니터링**
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

```
[weheheh:/maxgauge/TWEH12R2WP/util]$ls -lrt
total 224
drwxr-xr-x. 3 maxgauge oinstall 4096 Nov 18 2016 MaxGauge
-rwx-----. 1 maxgauge oinstall 1267 Nov 18 2016 MXG_DiskInfo.vbs
drwxr-xr-x. 6 maxgauge oinstall 4096 Nov 18 2016 scripts
-rwx-----. 1 maxgauge oinstall 153 Nov 18 2016 get_base.sql
-rwx-----. 1 maxgauge oinstall 192 Nov 18 2016 get_base.sh
-rwx-----. 1 maxgauge oinstall 152 Nov 18 2016 get_adm.sql
-rwx-----. 1 maxgauge oinstall 188 Nov 18 2016 get_adm.sh
drwxr-xr-x. 2 maxgauge oinstall 4096 Apr 27 2017 bin
drwxr-xr-x. 2 maxgauge oinstall 4096 Sep 14 15:43 etc
rw-r--r--. 1 maxgauge oinstall 1124 Nov 13 17:08 ts.sh
-rw-r--r--. 1 maxgauge oinstall 1276 Nov 13 17:09 ts.sql
-rwxr-xr-x. 1 maxgauge oinstall 1004 Nov 14 14:14 resource_limit.sql
-rw-r--r--. 1 maxgauge oinstall 403 Nov 14 16:20 tmp11.txt.old
-rw-r--r--. 1 maxgauge oinstall 402 Nov 14 16:20 tmp22.txt.old
-rw-r--r--. 1 maxgauge oinstall 7 Nov 14 18:37 rc1.sh
-rwxr-xr-x. 1 maxgauge oinstall 1348 Nov 14 19:31 rc.sh
-rw-r--r--. 1 maxgauge oinstall 402 Nov 15 08:52 tmp22.txt
-rw-r--r--. 1 maxgauge oinstall 484 Nov 15 08:52 tmp11.txt
-rw-r--r--. 1 maxgauge oinstall 149056 Nov 15 08:52 shlog.log
[weheheh:/maxgauge/TWEH12R2WP/util]$
```

✓ 시나리오

INSTANCE	: TWEH12R2WP
결과 확인 방법	: OS (Linux) ls -lrt 명령어

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

1

Script 파일의 실행권한을 확인합니다.

“평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!”

Alert Script

- **Script 등록**
 - **Script 모니터링**
 - RTM Alert log > User Script
 - Repository Table Data

» Alert User Script 추가 화면

- 1 Script 파일의 실행권한을 확인합니다.
 - 2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : rts.conf 내용 확인

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

“평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!”

Alert Script

- Script 등록
- Script 모니터링
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

SINGLE VIEW >

User Script

Instance : TWEH12R2WP DateTime : 2017-11-15 09:13:00 Status : Critical

1 #ALERT RC_C 09:13:05 processes 19.67%

PERFORMANCE ANALYZER

Alert Log 1-Day History GRID

id	Name	Value	Level	Log	User Script
1:00	SERVER A...		Critical		#ALERT RC_C...
1:00	SERVER A...		Critical		#ALERT RC_C...
1:00	SERVER A...		Critical		#ALERT RC_C...

Long Active Session Count GRAPH

Lock Waiting Session Count GRAPH

Background Instance Name: TWEH12R2WP ACTIVE

- 1 Script 파일의 실행권한을 확인합니다.
- 2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.
- 3 RTM 화면에서 알람 발생 여부를 확인합니다.

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : RTM>Alert log>User Script

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

“평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!”

Alert Script

- Script 등록
- Script 모니터링
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

partition_key	db_id	time	type	name	value	lvl	sms_flag	description
1	171115004	4 2017-11-15 09:34:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:34:00 processes 19.67%
2	171115004	4 2017-11-15 09:33:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:33:00 processes 19.67%
3	171115004	4 2017-11-15 09:32:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:32:00 processes 19.67%
4	171115004	4 2017-11-15 09:31:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:31:00 processes 19.67%
5	171115004	4 2017-11-15 09:30:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:30:00 processes 19.67%
6	171115004	4 2017-11-15 09:29:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:29:00 processes 19.67%
7	171115004	4 2017-11-15 09:28:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:28:00 processes 19.67%
8	171115004	4 2017-11-15 09:27:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:27:00 processes 19.67%
9	171115004	4 2017-11-15 09:26:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:26:00 processes 19.67%
10	171115004	4 2017-11-15 09:25:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:25:00 processes 19.67%
11	171115004	4 2017-11-15 09:24:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:24:00 processes 19.67%
12	171115004	4 2017-11-15 09:23:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:23:00 processes 19.67%
13	171115004	4 2017-11-15 09:22:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:22:00 processes 19.67%
14	171115004	4 2017-11-15 09:21:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:21:00 processes 19.67%
15	171115004	4 2017-11-15 09:20:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:20:00 processes 19.67%
16	171115004	4 2017-11-15 09:19:00	2 SERVER ALERT	SCRIPTALERT SCRIPTALERT inval obj	0	2 0		#ALERT RC C 09:19:00 processes 19.67%

- 1 Script 파일의 실행권한을 확인합니다.
- 2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.
- 3 RTM 화면에서 알람 발생 여부를 확인합니다.
- 4 Repository > ora_alarm_history Table Data를 확인합니다.

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : ora_alarm_history Table

✓ 목표

사용자가 평소 사용하는 script를 이용하여
알람을 등록할 수 있다.

실전 활용 사례 Case 2.

업무시간에 Backup 이 수행되는 경우를
방지하기 위해, Script를 이용하여 Backup
Alert을 등록하고 싶어요.

“업무 시간에 Backup 수행 되는 것을 방지하기 위해 Alert 등록하고 싶어요!”

Alert Script

- **Script 등록**
- **Script 모니터링**
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

```
-rwxr--r--. 1 maxgauge oinstall 2355 Nov 23 10:25 bak.sh
-rwx-----. 1 maxgauge oinstall 776 Nov 23 10:26 bak5.sql
-rw-r--r--. 1 maxgauge oinstall 1012 Nov 23 10:33 xmtest.sh
-rw-r-----. 1 maxgauge oinstall 394 Nov 23 11:20 weh_temp1.txt
-rw-r-----. 1 maxgauge oinstall 218 Nov 23 11:20 weh_temp2.txt
-rw-r--r--. 1 maxgauge oinstall 1481349 Nov 23 11:20 bak_shlog.log
[weheheh:/maxgauge/TWEH12R2WP/util]$ ①
```

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : OS (Linux) ls -lrt 명령어

✓ 목표

사용자가 평소 사용하는 script를 이용하여
알람을 등록할 수 있다.

1

Script 파일의 실행권한을 확인합니다.

“업무 시간에 Backup 수행 되는 것을 방지하기 위해 Alert 등록하고 싶어요!”

Alert Script

- Script 등록
- Script 모니터링
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

```
db_id = 4
# Daemon Server Host
daemon_port=5083
# Listener IP Address and port
wr_host=10.10.30.62
wr_port=7001
# DB Read Timeout (milliseconds)
stat_interval=1
# DB Write Send Interval (milliseconds)
send_sysstat_interval=1
# DB Session Data Interval (milliseconds)
session_list_interval=1
# DB Session Dump Interval 0:OFF (milliseconds)
send_session_list_interval=1
# Lock List Interval (milliseconds)
send_lock_info_interval=1
# Top Processes Interval 0:OFF (milliseconds)
top_process_interval=3
# Xview Limit (XView:maximum, XView Off, Not Loading)
xview_limit=1000
# SqlBase XView Gathering Count (maximum=20)
sqllog_gathering_count=20
# Listener Check Interval (maximum=10000, minimum=1000, 10.0.1.252:1521)
listener=10.10.30.250:1521
# LISTENER CHECK INTERVAL (milliseconds)
listener_check_interval=10
# User Program Status (maximum=1, minimum=0, default=0, Full (push))
script1=backup_status,60,0,/maxgauge/TWEH12R2WP/util/bak.sh
# script interval (1s, 0.5s, 0.25s, 0.125s, 0.0625s, 0.03125s)
```

②

1 Script 파일의 실행권한을 확인합니다.

2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : rts.conf 내용 확인

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

“업무 시간에 Backup 수행 되는 것을 방지하기 위해 Alert 등록하고 싶어요!”

Alert Script

- Script 등록
- Script 모니터링
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가 화면

The screenshot shows the 'AlertUserScript' configuration window. The script is named 'ACTIVE'. It has one parameter: 'instance: TWEH12R2WP'. The 'date/time' is set to '2017-11-23 11:26:00' and the 'status' is 'Critical'. The script body contains the command: '1 ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.dbf'. To the right of the configuration window, there is a 'Alert Log' table and two monitoring graphs: 'Long Active Session Count' and 'Lock Waiting Session Count'. The 'Alert Log' table shows two entries: '1:27:00 SERVER A... Critical ACTIVE 4 /or...' and '1:26:00 SERVER A... Critical ACTIVE 4 /or...'. The 'Long Active Session Count' graph shows a single session for 'TWEH12R2WP' with a value of 1. The 'Lock Waiting Session Count' graph also shows a single session for 'TWEH12R2WP' with a value of 1.

- 1 Script 파일의 실행권한을 확인합니다.
- 2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.
- 3 RTM 화면에서 알람 발생 여부를 확인합니다. (업무시간 : 09:00:00 ~ 18:00:00, ACTIVE 상태인 Data file number 및 name 정보 출력)

✓ 시나리오

INSTANCE : TWEH12R2WP
결과 확인 방법 : RTM>Alert log>User Script

✓ 목표

사용자가 평소 사용하는 script를 이용하여
알람을 등록할 수 있다.

“업무 시간에 Backup 수행 되는 것을 방지하기 위해 Alert 등록하고 싶어요!”

Alert Script

- Script 등록
- **Script 모니터링**
- RTM Alert log > User Script
- Repository Table Data

» AlertUserScript 추가화면

id	alarm_key	db_id	time	type	alert_type	name	value	lvl	sms_flag	description
1	171123004	4	2017-11-23 11:40:00	2	SERVER ALERT	SCRIPTALERT	backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
2	171123004	4	2017-11-23 11:59:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
3	171123004	4	2017-11-23 11:38:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
4	171123004	4	2017-11-23 11:37:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
5	171123004	4	2017-11-23 11:36:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
6	171123004	4	2017-11-23 11:35:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
7	171123004	4	2017-11-23 11:34:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
8	171123004	4	2017-11-23 11:33:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
9	171123004	4	2017-11-23 11:32:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
10	171123004	4	2017-11-23 11:31:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
11	171123004	4	2017-11-23 11:30:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
12	171123004	4	2017-11-23 11:29:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
13	171123004	4	2017-11-23 11:28:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
14	171123004	4	2017-11-23 11:27:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.
15	171123004	4	2017-11-23 11:26:00	2	SERVER ALERT	SCRIPTALERT	SCRIPTALERT backup status	0	2 0	ACTIVE 4 /oradata/weheheh/users01.dbf ACTIVE 3 /oradata/weheheh/undotbs01.

- 1 Script 파일의 실행권한을 확인합니다.
- 2 rts.conf 파일에 사용할 script 정보를 등록시킵니다.
- 3 RTM 화면에서 알람 발생 여부를 확인합니다. (업무시간 : 09:00:00 ~ 18:00:00, ACTIVE 상태인 Data file number 및 name 정보 출력)
- 4 Repository > ora_alarm_history Table Data를 확인합니다.

✓ 시나리오

INSTANCE

: TWEH12R2WP

결과 확인 방법

: ora_alarm_history Table

✓ 목표

사용자가 평소 사용하는 script를 이용하여
알람을 등록할 수 있다.

MAXGAUGE



Practical Guide

Script Manager [Real Time Monitor]

Contents

Script Manager?

Script Manager 의 활용

Case 1. User Script

Process 와 Session 의 사용량(Usage %)을 알고 싶어요.

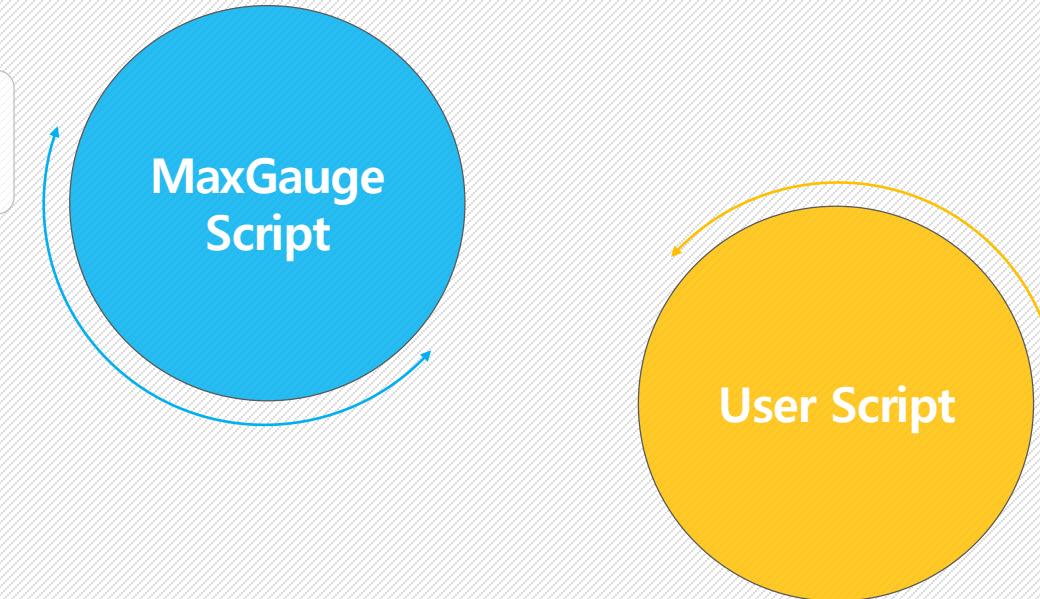
Case 2. MaxGauge Script

Real-Time Monitor에서 Tablespace Alert 이 발생하였습니다.
현재 시점의, Tablespace 사용정보를 확인하고 싶습니다.

Script Manager?

Script Manager 는 언제 쓰나요?

DBA에게 유용한 다양한 **MaxGauge Script** 제공
파라미터정보, 테이블 스페이스 정보 등
시스템관리에 필요한 정보조회



» 사용자들이 자주 사용하는 Script 를 직접 등록하여 사용할 수 있는 **User Script**

Script Manager 의 활용

Script Manager 의 사용 방법

Script Manager

- 화면구성
- User Script 등록
- User Script 삭제
- User Script 수정

» 화면구성



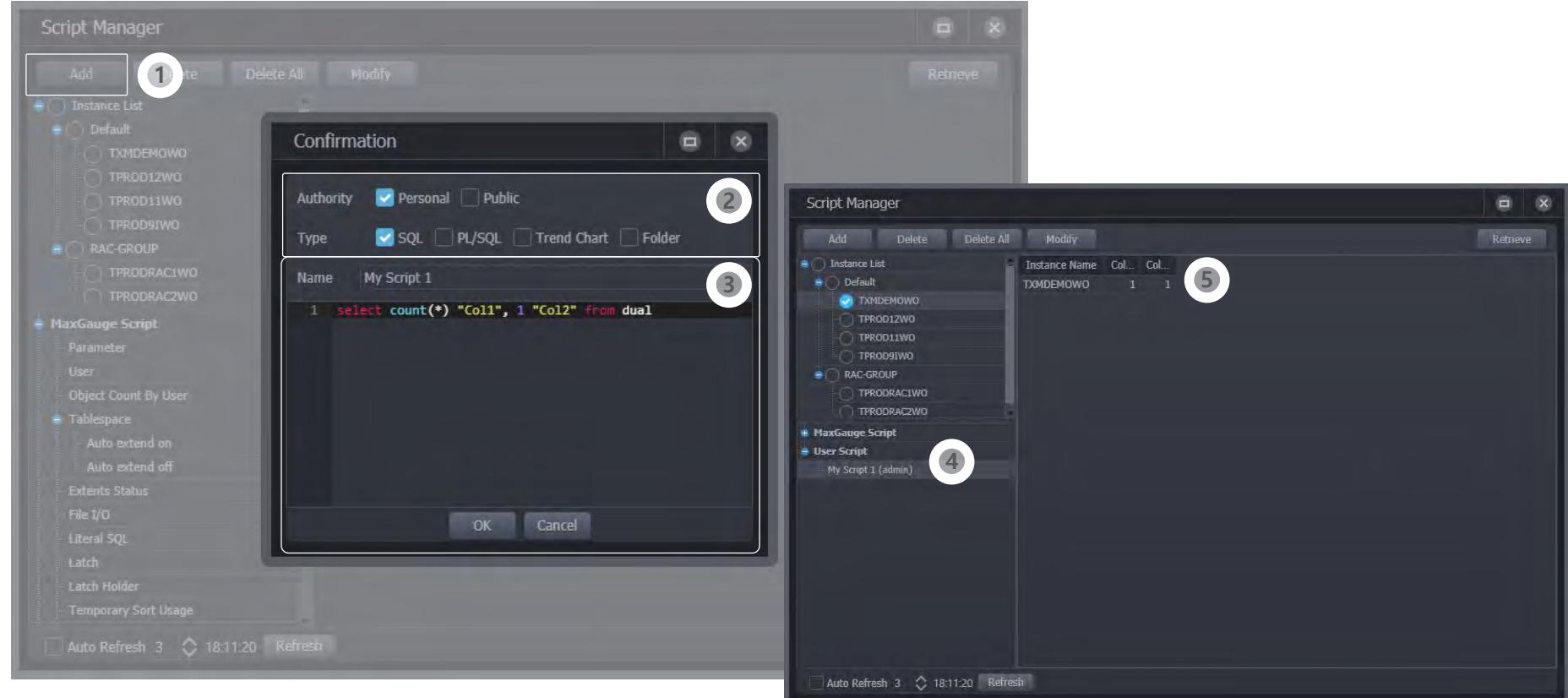
- 1 User Script 등록, 삭제, 수정하는 기능입니다.
- 2 Instance List 인스턴스 선택화면입니다.
- 3 MaxGauge Script(기본제공) & User Script(사용자 등록) 목록 표시화면입니다.
- 4 수행결과 표시 화면

Script Manager 의 사용 방법

Script Manager

- 화면구성
- User Script 등록
- User Script 삭제
- User Script 수정

» UserScript등록



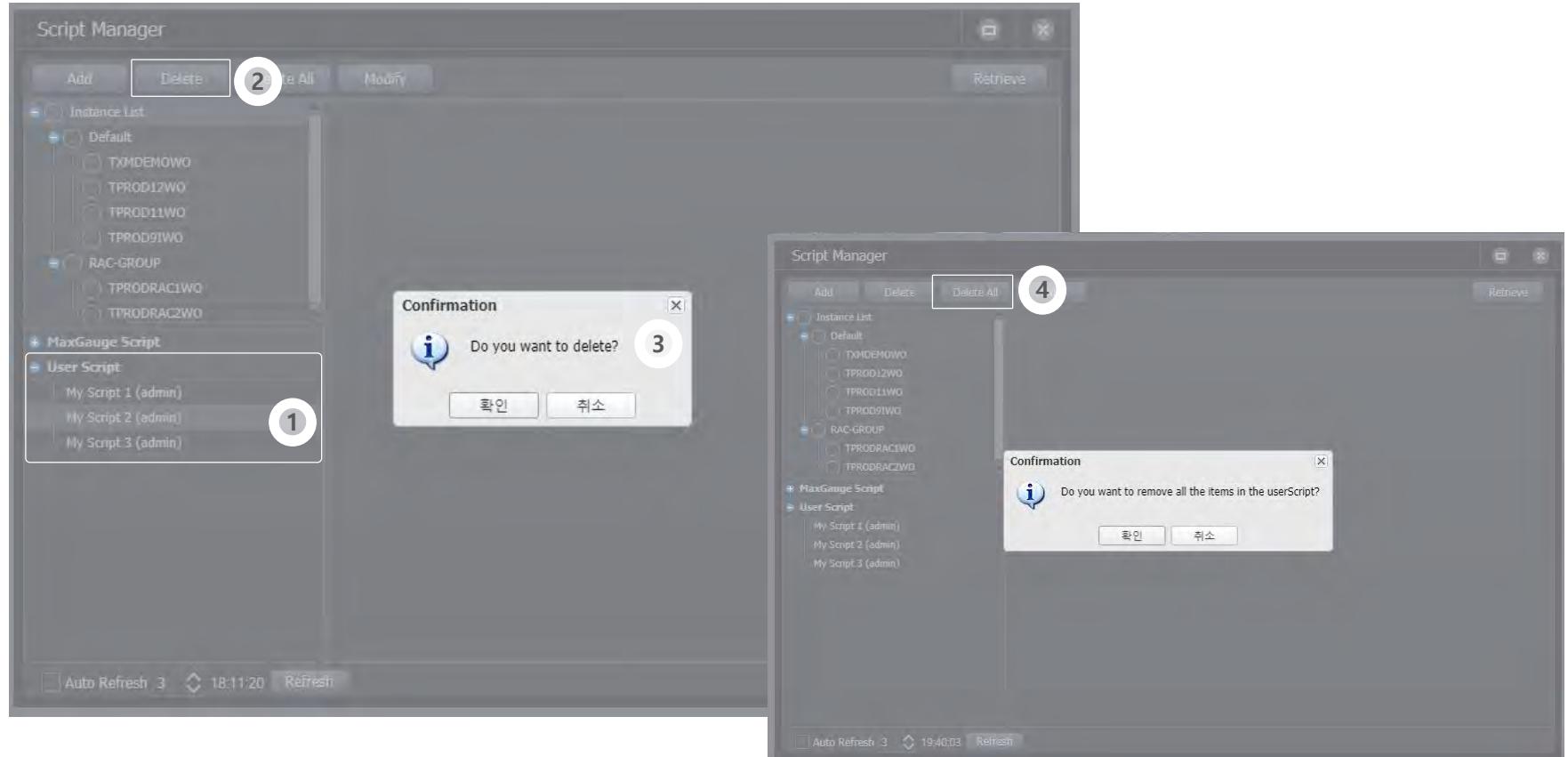
- 1 Add 버튼을 클릭합니다.
- 2 Authority 및 Type 을 선택합니다.
- 3 Name 및 Script 입력 후 OK 버튼을 클릭합니다. (등록 완료)
- 4 User Script 목록을 확인하여 정상 등록 여부 확인 합니다.
- 5 Instance List에서 Instance 를 선택 한 후 User Script를 더블 클릭하거나, 수행 할 User Script를 선택한 후 Retrieve 버튼을 클릭하여 스크립트를 수행결과를 확인합니다.

Script Manager 의 사용 방법

Script Manager

- 화면구성
- User Script 등록
- User Script 삭제
- User Script 수정

» UserScript삭제



- 1 삭제 대상 User Script 를 선택합니다.
- 2 Delete 버튼을 클릭 합니다.
- 3 확인 또는 취소 버튼을 클릭합니다.
- 4 Delete 버튼을 클릭 후 확인 및 취소 버튼을 클릭하여 User Script 를 전체 삭제 할 수 있습니다.

Script Manager 의 사용 방법

Script Manager

- 화면구성
- User Script 등록
- User Script 삭제
- User Script 수정

» UserScript수정



- 1 수정할 User Script 를 선택합니다.
- 2 Modify 버튼을 클릭합니다.
- 3 내용 수정 후 OK 버튼을 클릭합니다.

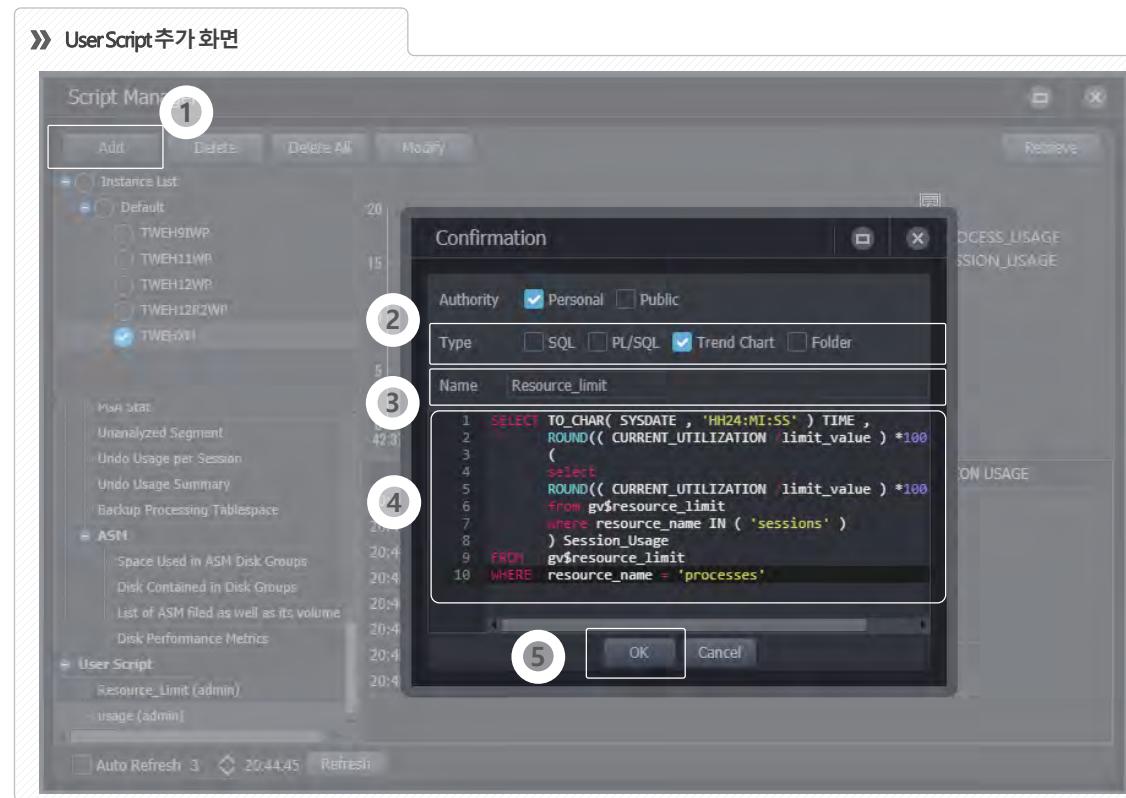
실전 활용 사례 Case 1.

**Process 와 Session 의 사용량(Usage %)을
알고 싶어요.**

“Process 와 Session 의 사용량%(Usage)을 알고 싶어요!”

User Script

- Script 등록
- Script 모니터링
- Trend Chart 유형
- SQL 유형



- 1 Add 버튼을 클릭하여, Script 등록 창을 실행 시킵니다.
- 2 Type에서 Trend Chart 를 'Check' 합니다.
- 3 Name 영역에, 'Title name ' 을 입력 합니다.
- 4 Script 내용을 등록 합니다.
- 5 OK 버튼을 선택하여, Script 내용을 저장 합니다.

✓ 시나리오

INSTANCE : TWEHXM
결과 확인 방법 : Trend Chart (그래프)

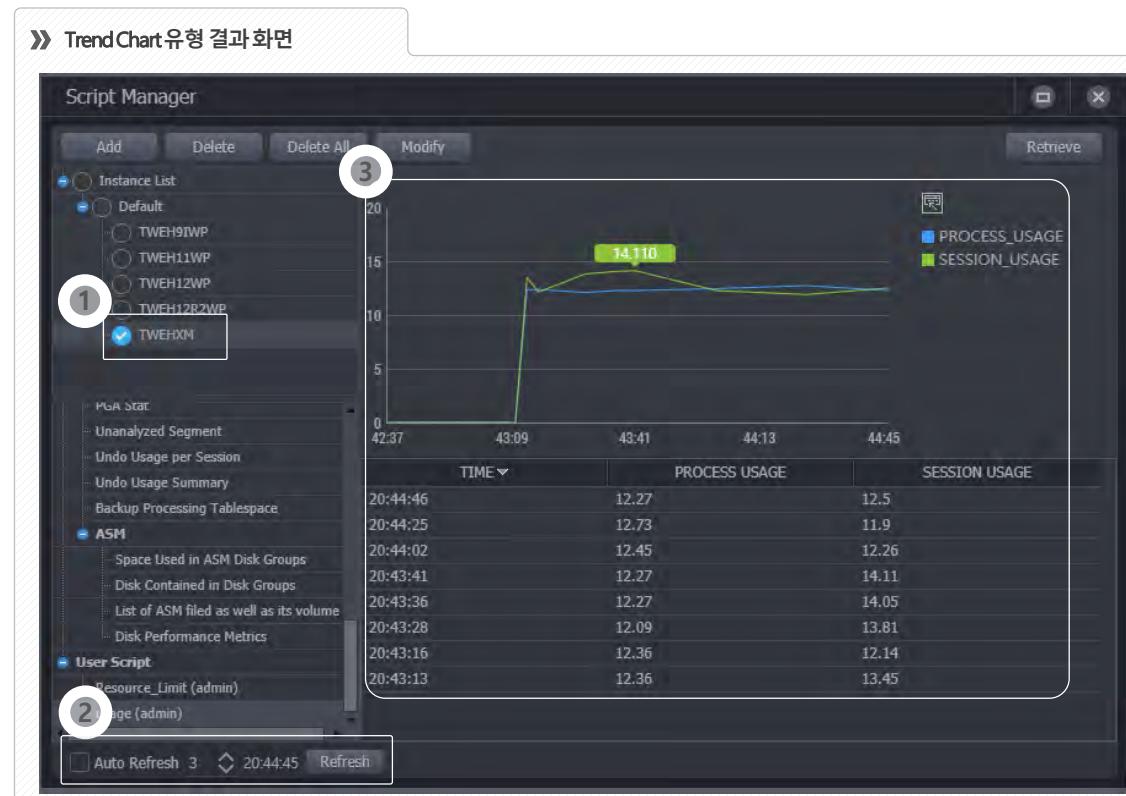
✓ 목표

Process 의 사용량과,
Session 의 사용량 확인 하기

“Process 와 Session 의 사용량%(Usage)을 알고 싶어요!”

User Script

- Script 등록
- Script 모니터링
- Trend Chart 유형
- SQL 유형



- 1 Instance List 영역에서 'TWEHXM' Instance 를 'Check' 합니다.
- 2 Auto Refresh (Default 3) sec 를 선택하거나, Guide Refresh 를 선택 합니다.
- 3 Trend Chart 에 수행 결과가 순차적으로 출력 됩니다.
- Trend Chart 유형의 경우, Instance 2 개 이상 선택한 경우에는 Grid 만 제공 합니다.

✓ 시나리오

INSTANCE : TWEHXM
결과 확인 방법 : Trend Chart (그래프)

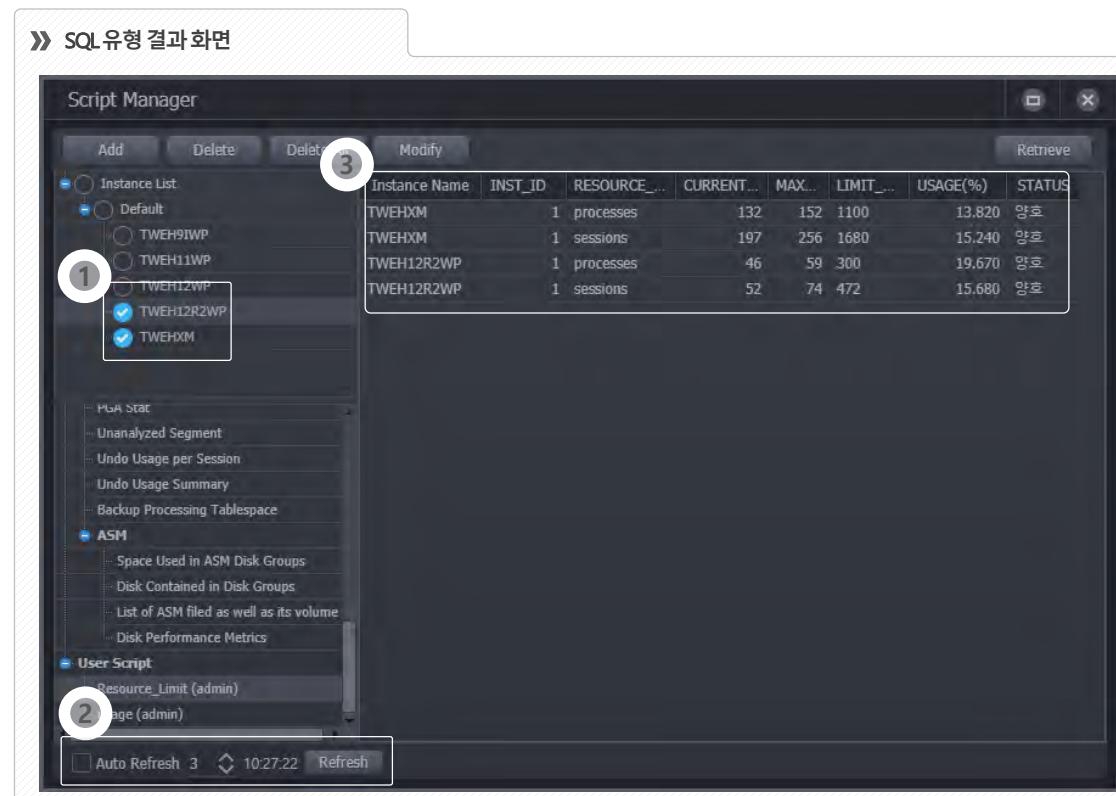
✓ 목표

Process 의 사용량과,
Session 의 사용량 확인 하기

“Process 와 Session 의 사용량%(Usage)을 알고 싶어요!”

User Script

- Script 등록
 - **Script 모니터링**
 - Trend Chart 유형
 - SQL 유형



- 1 Instance List 영역에서 'TWEHXM' , 'TWEH12R2WP' Instance 2개를 'Check' 합니다.
 - 2 Auto Refresh (Default 3) sec 를 선택하거나, Guide Refresh 를 선택 합니다.
 - 3 Instance 별 수행 결과가 순차적으로 출력 됩니다.

✓ 시나리오

INSTANCE

: TWEHXM, TWEH12R2WP

결과 확인 방법

: Trend Chart 유형

: SQL 유형

✓ 목표

Process 의 사용량과,
Session 의 사용량 확인 하기

Script Manager 활용 Case 2.

**RTM에서 Tablespace Alert 발생 시
현재 시점의 Tablespace 사용정보를 확인하고 싶습니다.**

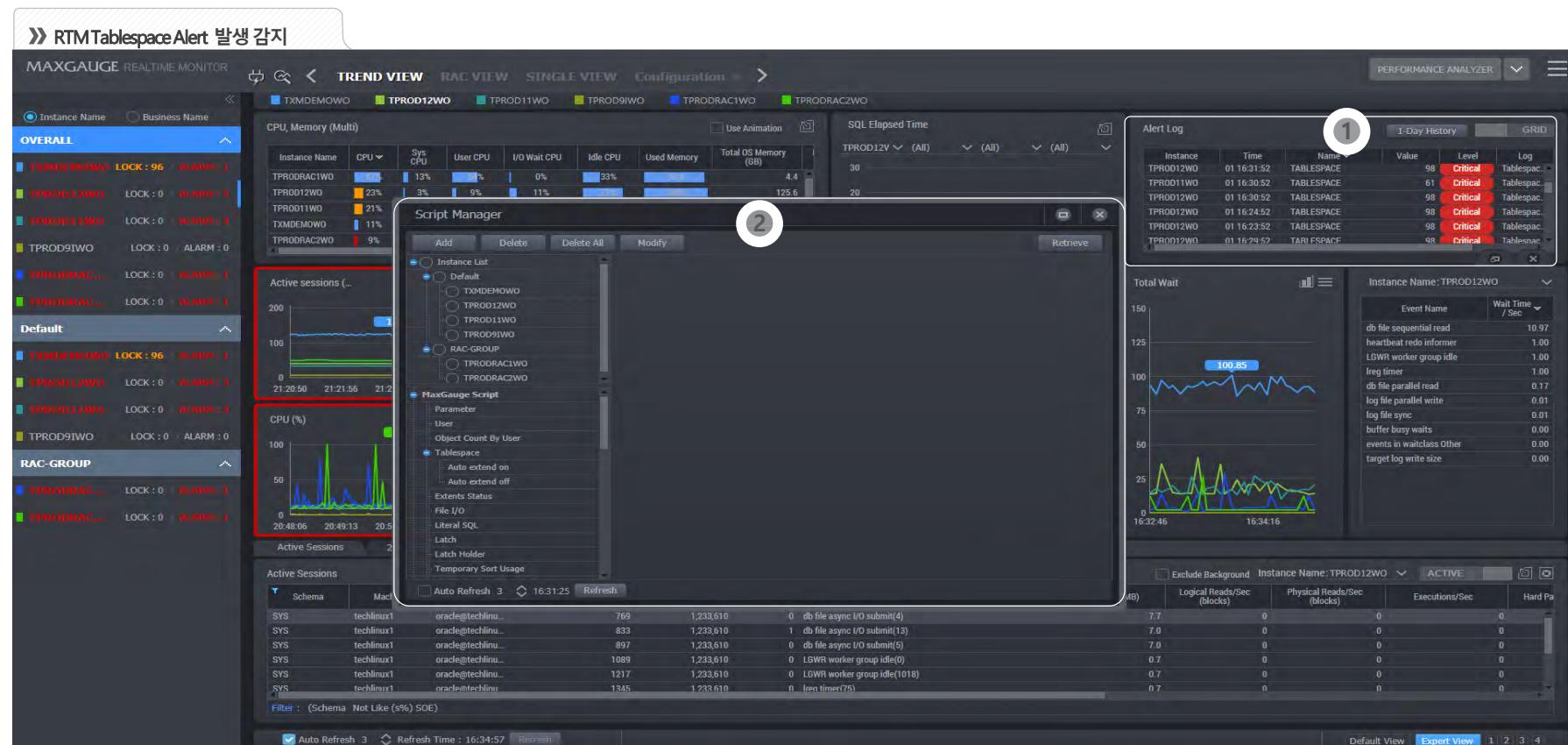
Tablespace Alert 발생 시 현재 시점의 Tablespace 사용정보를 확인하고 싶습니다.

STEP

1. RTM에서 Tablespace Alert 감지

2. Script Manger Tablespace 정보 확인

- 해당 인스턴스 선택
- MaxGauge Script 선택
- Auto extend on 더블클릭
- Tablespace 사용정보 확인



1 Real- Time Monitor에서 Tablespace Alert이 감지 되었습니다.

2 메뉴 -> Script Manager를 실행합니다.

Tablespace Alert 발생 시 현재 시점의 Tablespace 사용정보를 확인하고 싶습니다.

STEP

1. RTM에서 Tablespace Alert 감지

2. Script Manger에서 Tablespace

정보 확인

- 해당 인스턴스 선택
- MaxGauge Script 선택
- Auto extend on 더블클릭
- Tablespace 사용정보 확인

» Script Manager Tablespace 정보 확인

Script Manager

Instance Name	TableSpace	File Name	Total Space(MB)	Total Blocks	Free Space(MB)	Free Blocks	# of Frags	Biggest Contiguous Blocks	Smallest Contiguous Blocks	Usage
TPROD12WO	EXAMPLE	/ora_data/ora12c/ORA12102/exa...	32,767	4,194,302	32,368	4,143,150	3	104,960	48	1.2%
	MGMT_AD41_TS	/ora_data/ora12c/ORA12102/mg...	32,768	4,194,302	32,368	4,143,150	3	104,960	48	1.2%
	MGMT_ECM_D...	/ora_data/ora12c/ORA12102/mg...	32,768	4,194,302	32,767	4,194,174	1	25,472	25,472	0.0%
	MGMT_TABLE...	/ora_data/ora12c/ORA12102/mg...	32,768	4,194,302	32,767	4,194,118	1	25,472	25,472	0.0%
	SH	/ora_data/ora12c/ORA12102/sh...	33,554,431	4,294,967...	33,550,164	4,294,421...	9	262,144	5,760	0.0%
	SOE	/ora_data/ora12c/ORA12102/soe...	33,554,431	4,294,967...	33,354,247	4,269,343...	4	262,144	66,560	0.6%
	SYSAUX	/ora_data/ora12c/ORA12102/sy...	32,767	4,194,302	607	77,712	191	4,224	8	98.1%
	SYSTEM	/ora_data/ora12c/ORA12102/syst...	32,768	4,194,302	607	77,712	191	4,224	8	98.1%
	TPCH	/ora_data/ora12c/ORA12102/tph...	133,120	17,039,360	7,624	975,928	18	262,144	24	2.9%
	UNDOTBS1	/ora_data/ora12c/ORA12102/und...	32,767	4,194,302	30,365	3,886,806	2,841	507,904	24	2.9%
			32,768	4,194,302	30,366	3,886,806	2,841	507,904	24	94.3%

Auto Refresh 3 20:14:47 Refresh

1 대상 Instance 를 선택합니다.

2 MaxGauge Script 중 Tablespace – auto extend on 더블클릭 합니다.

3 Tablespace 정보 확인합니다.

Thank You