MAXGAUGE
Practical Guide

PRODUCT
DOCUMENTATION

Contents

Contents

Contents

Contents

Practical Guide

Alert Configuration

Contents

Alert Configuration

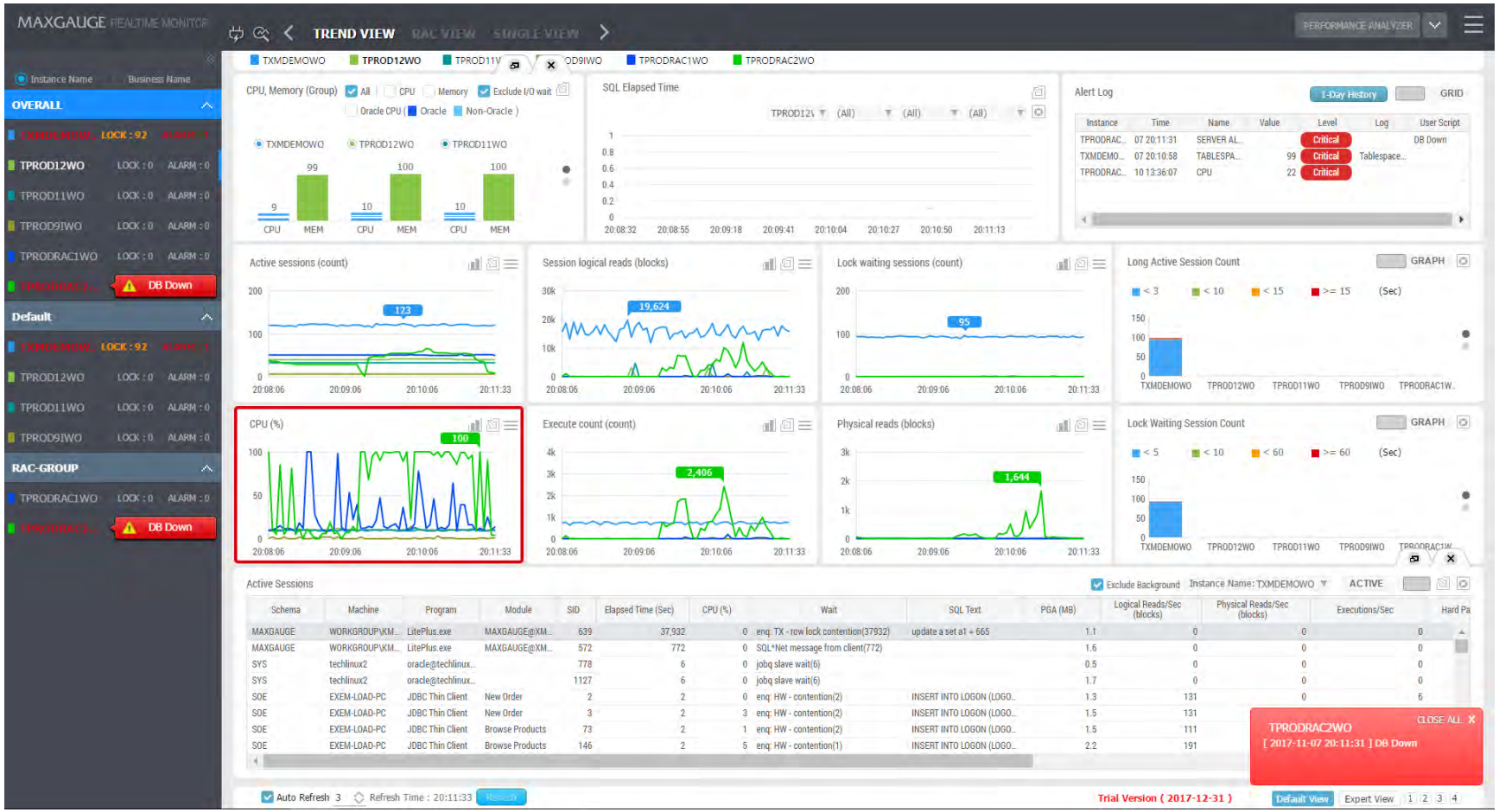
모든 성능 지표에 대한 **Alert** 설정 기능

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

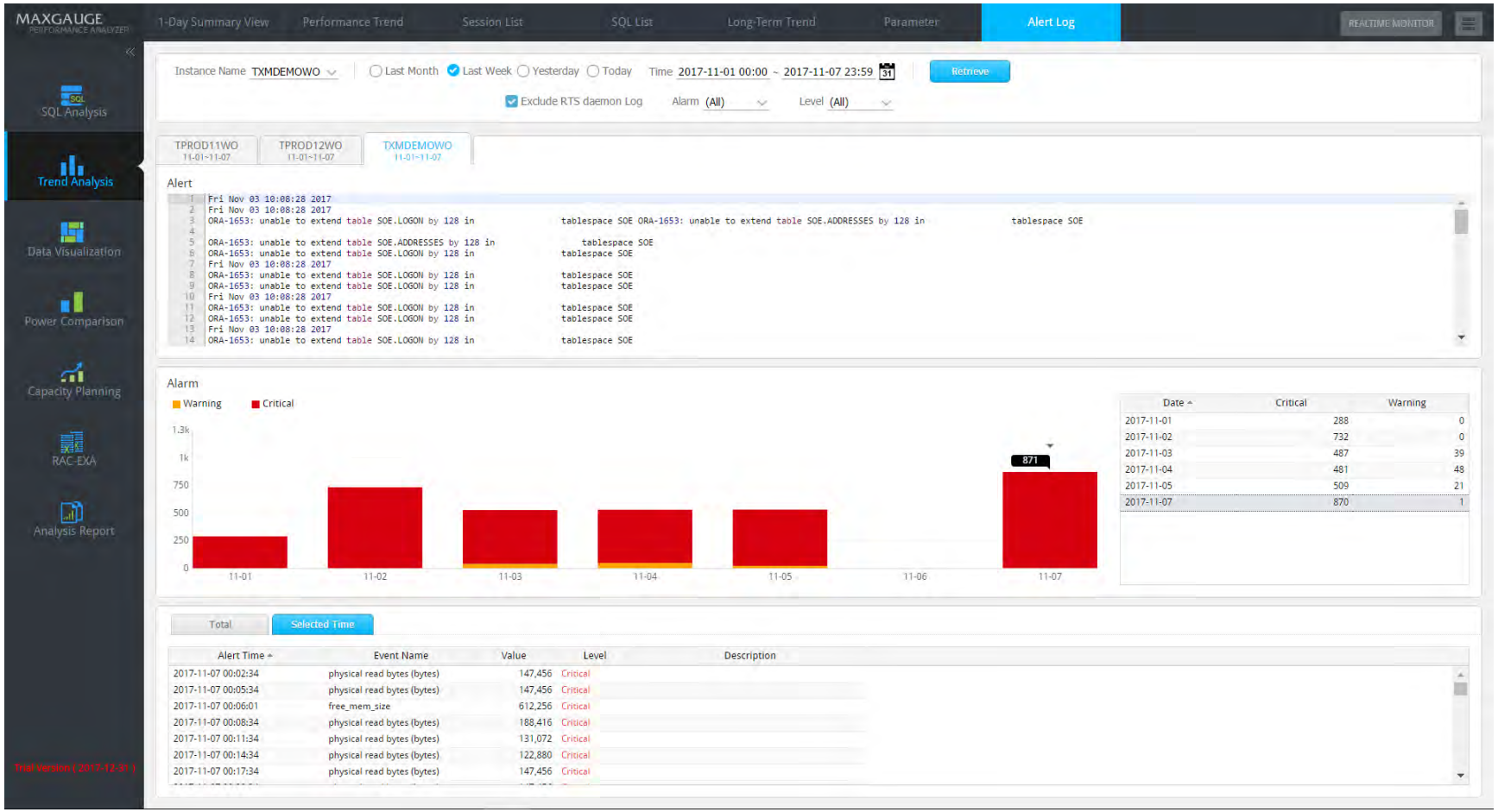
Config에서 설정 된 Alert 에 대하여 **실시간 감지**

- ✓ 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 Chart 에서는 화면 색상이 변경 됩니다. (Warning - 노란색 / Cortical - 빨간색)



발생 되었던 Alert 에 대하여 **사후 분석**

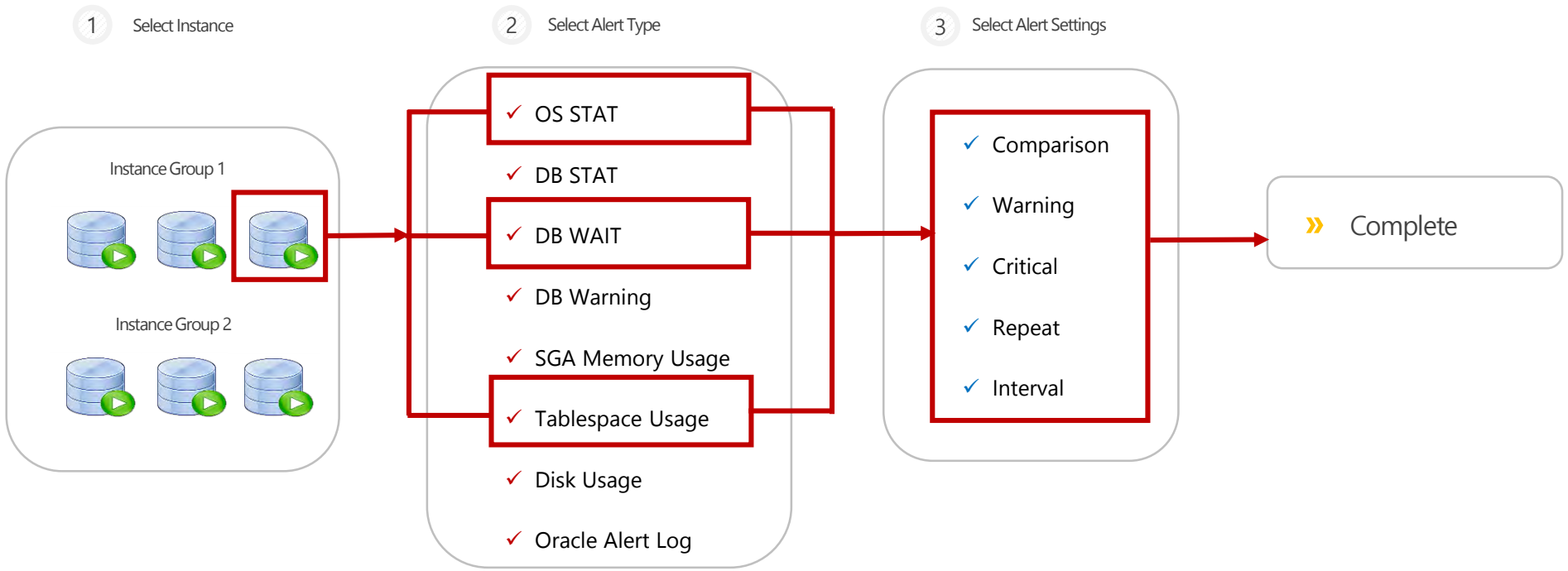
- ✓ Oracle Alert Log 발생 이력을 제공해 줍니다.
- ✓ Configuration 화면에서 설정한 지표 별 임계 값 (threshold value)을 조건에 만족하는 횟수를 시간 별 추이 그래프 및 그리드로 제공합니다.



Alert 사용자 가이드

Alert 설정 가이드

» 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 Type 구분

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

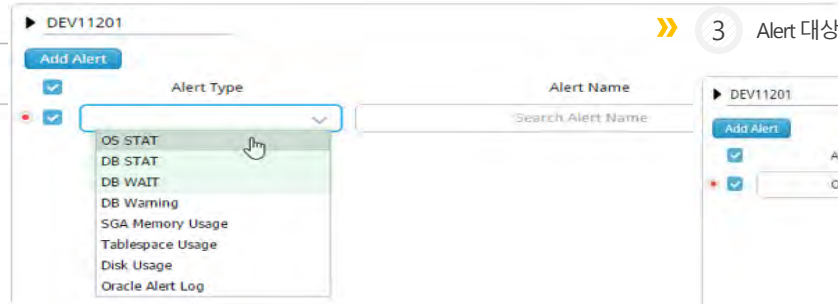
Alert 설정 가이드

Instance Alert 설정

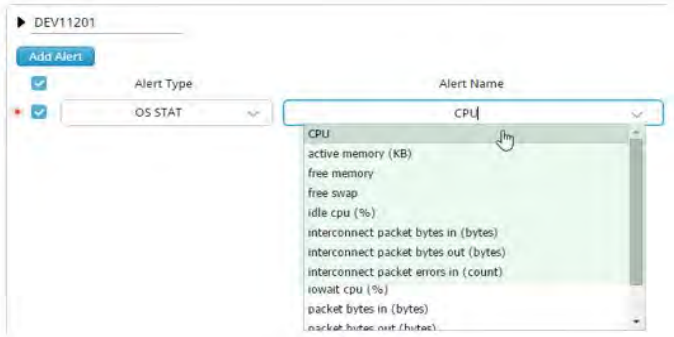
OS STAT / DB STAT / DB Wait

1 Alert 대상 Instance 를 선택한다

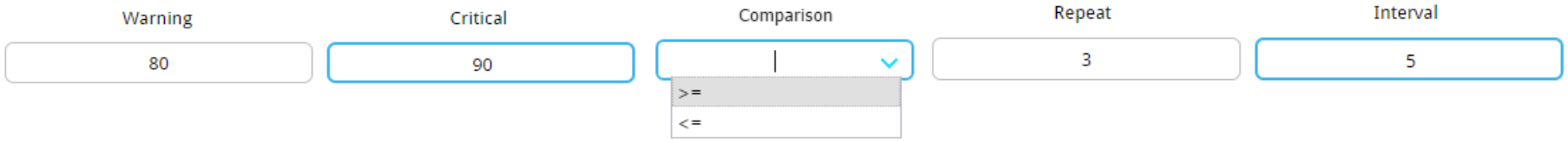
2 Alert Type 의 OS STAT/DB STAT/DB Wait 중 필요 Type 선택한다.



3 Alert 대상을 선택한다. (Alert Name)



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

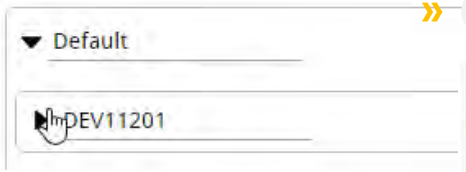


5 Save All 버튼 클릭

Alert 설정 가이드

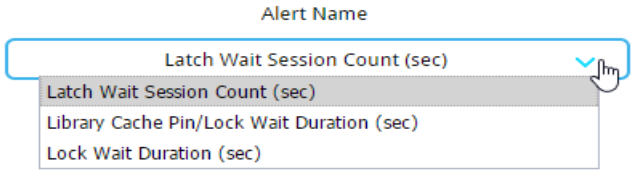
Instance Alert 설정
DB Warning

1 Alert 대상 Instance 를 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

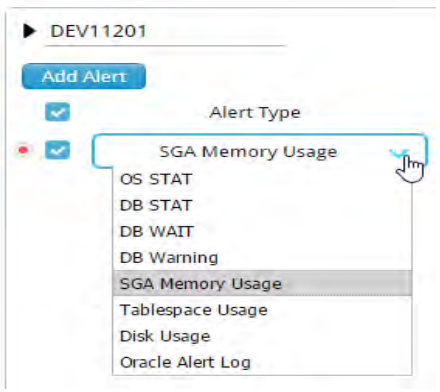
항목	설명
Alert Type	● DB Warning
Alert Name	● Lock Wait Duration (sec) : Lock의 지속시간을 초단위로 설정. ● Latch Wait Session Count (sec) : Latch 이벤트를 대기하는 세션들의 개수. ● Library Cache Pin / Lock (sec) : Library Cache Pin/Lock의 두 이벤트를 합한 시간.

Alert 설정 가이드

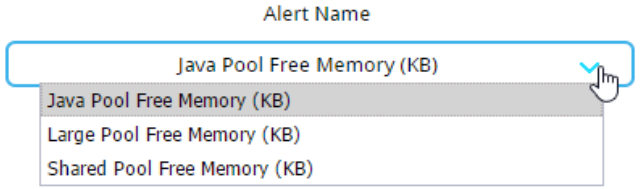
Instance Alert 설정
SGA Memory Usage

1 Alert 대상 Instance 를 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.

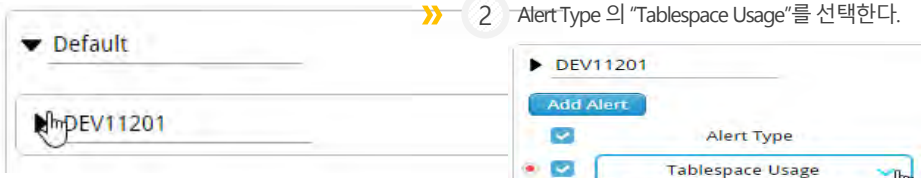
5 Save All 버튼 클릭

항목	설명
Alert Type	● SGA Memory Usage
Alert Name	● 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

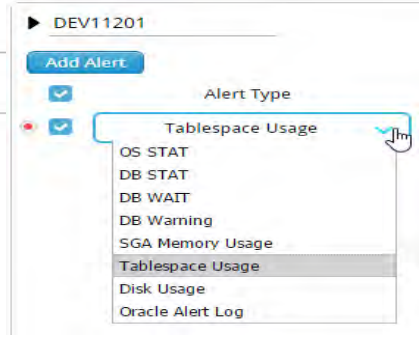
Alert 설정 가이드

Instance Alert 설정
Tablespace Usage

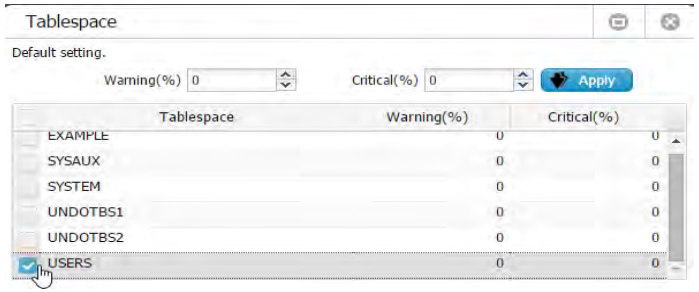
1 Alert 대상 Instance 를 선택한다



2 Alert Type 의 "Tablespace Usage"를 선택한다.



3 Alert 대상을 선택한다. (Alert Name) – 팝업 창



4 Alert Settings – 임계 값을 설정 해 준다.

5 OK 버튼 클릭한다.

6 발생 주기를 설정한다.

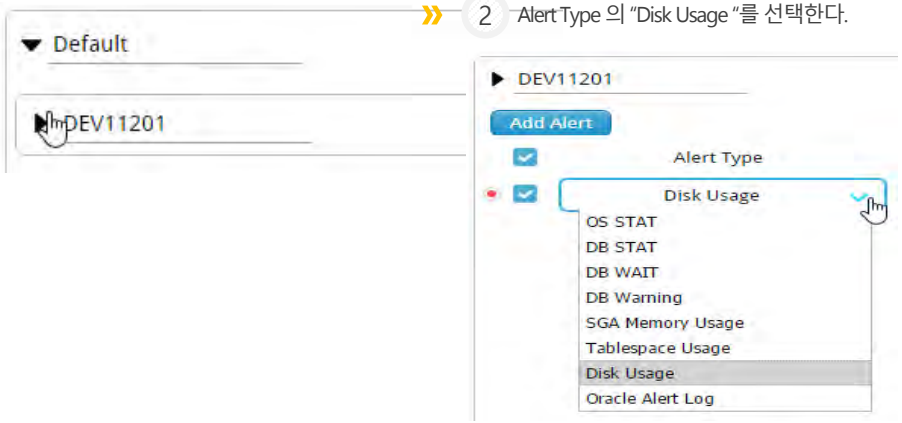
7 Save All 버튼 클릭한다.

Alert 설정 가이드

Instance Alert 설정

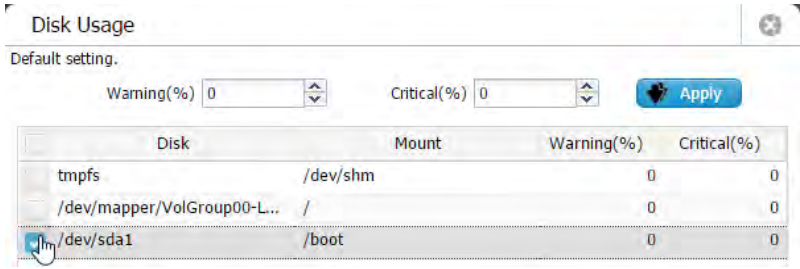
Disk Usage

1 Alert 대상 Instance 를 선택한다



2 Alert Type 의 "Disk Usage "를 선택한다.

3 Alert 대상을 선택한다. (Alert Name) – 팝업 창



4 Alert Settings – 임계 값을 설정 해 준다.

Disk	Mount	Warning(%)	Critical(%)
tmpfs	/dev/shm	0	91
/dev/mapper/VolGroup00-L...	/	0	0
/dev/sda1	/boot	90	95

5 OK 버튼 클릭한다.

6 발생 주기를 설정한다.

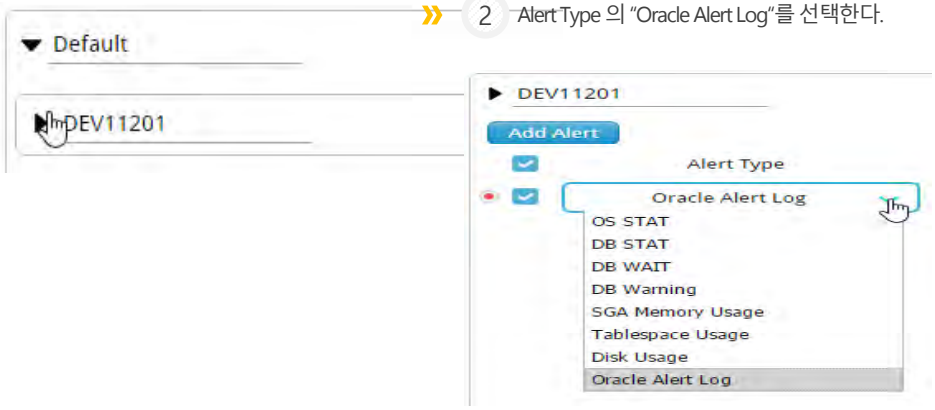
7 Save All 버튼 클릭한다.

Alert 설정 가이드

Instance Alert 설정

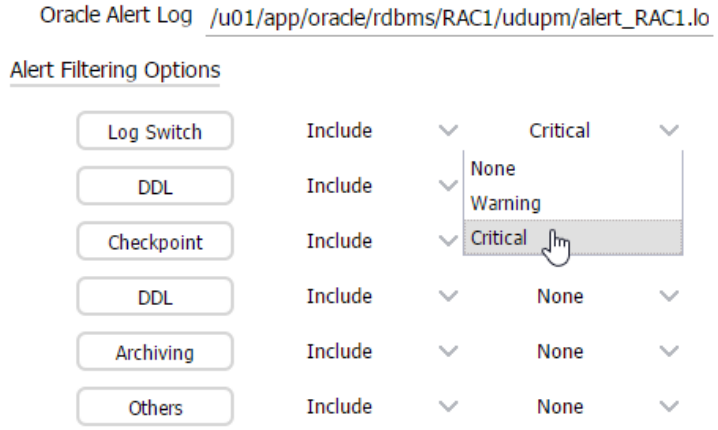
Oracle Alert Log

1 Alert 대상 Instance 를 선택한다

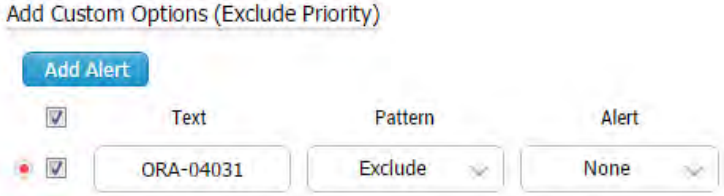


2 Alert Type 의 "Oracle Alert Log"를 선택한다.

3 Alert 대상을 선택한다. (Alert Name) – 팝업 창



4 Add Custom Option 을 설정한다. (선택사항)



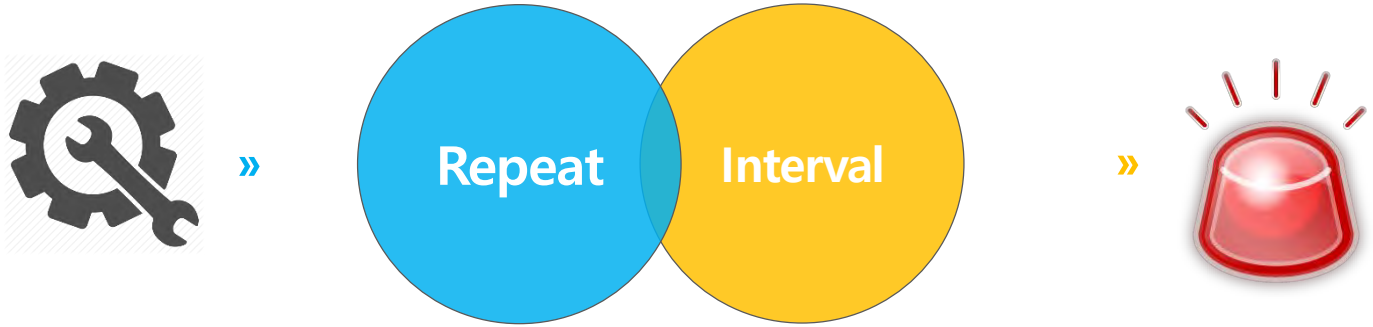
5 OK 버튼 클릭한다.

6 발생 주기를 설정한다.

7 Save All 버튼 클릭한다.

Repeat / Interval 에 따른 발송 규칙

» Alert 발생 관련 파라미터



Instance Alert 설정 » Alert 발생 관련 파라미터 » 일람 발생

- ✓ DG 는 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 / Interval 에 따른 발송 규칙

» Alert발생 규칙

1. Repeat count 가 0으로 초기화 되는 기준

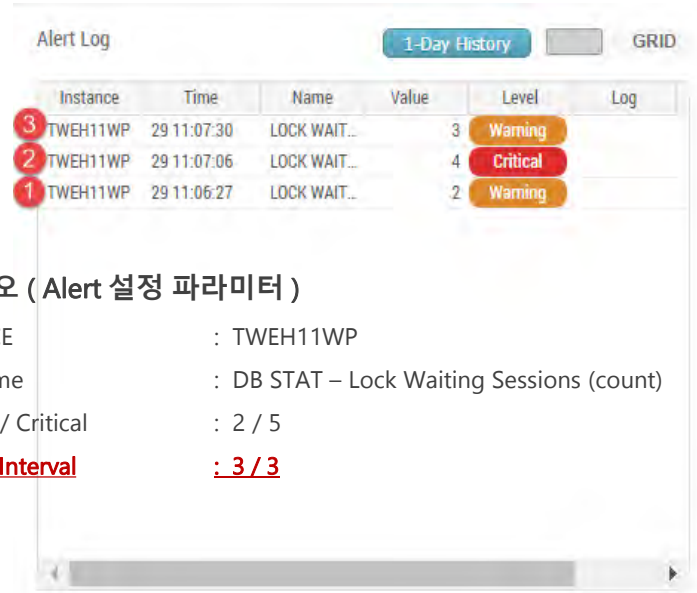
- Repeat count 가 지정한 수치에 도달하는 순간
- Alert(N/W/C)이 발생하는 순간 (W->C, C->W 는 제외)
- Normal이 체크되는 순간

※ W -> C, C -> W Level 이 변하는 경우 Repeat Count가 초기화 되지 않음

따라서, W,C 의 경계를 왔다 갔다 하는 경우 Alert Check 시점의 Level 로 Alert 이 발생함

2. 알람 발생 기준

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

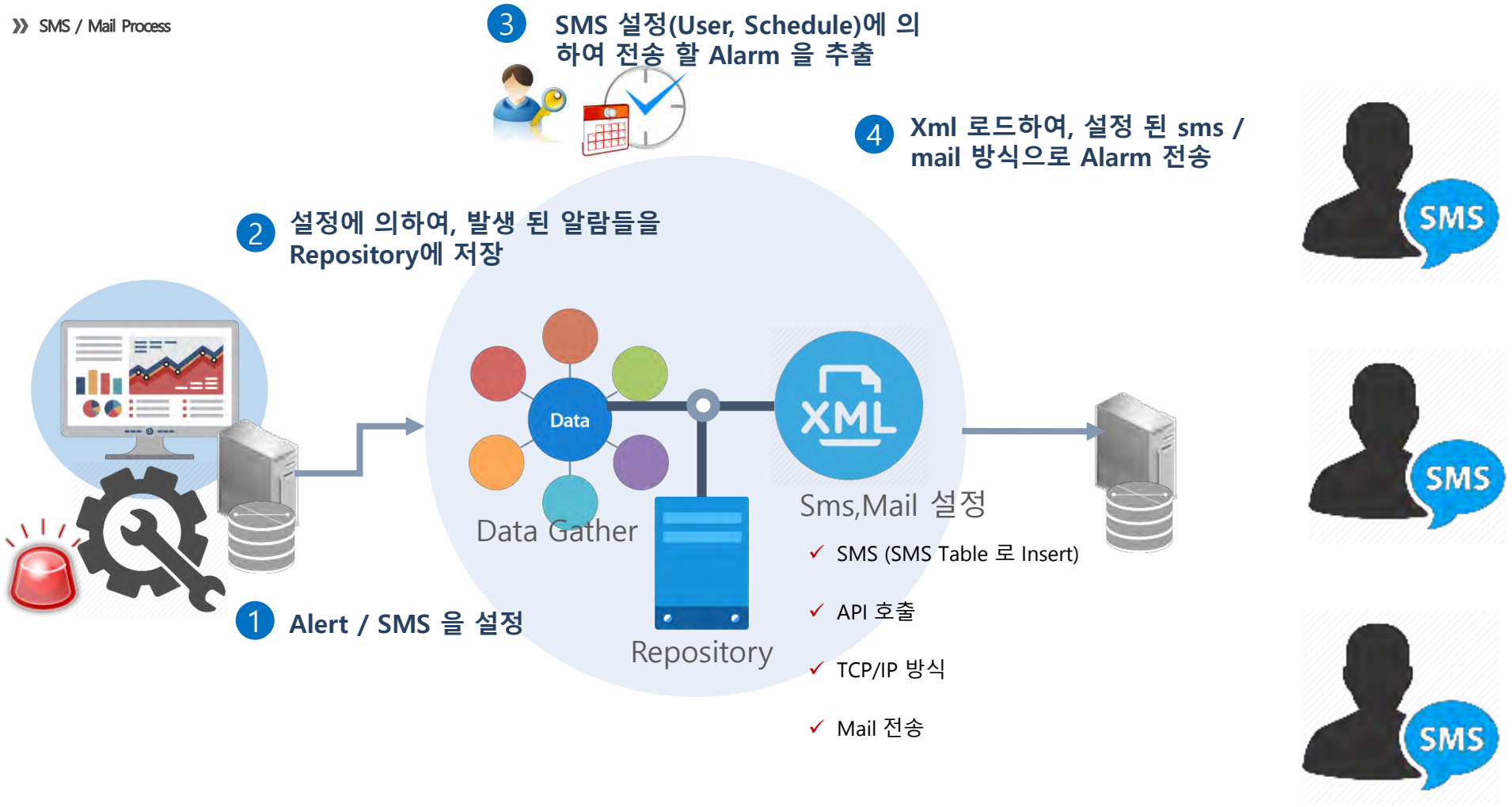


✓ **시나리오 (Alert 설정 파라미터)**

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

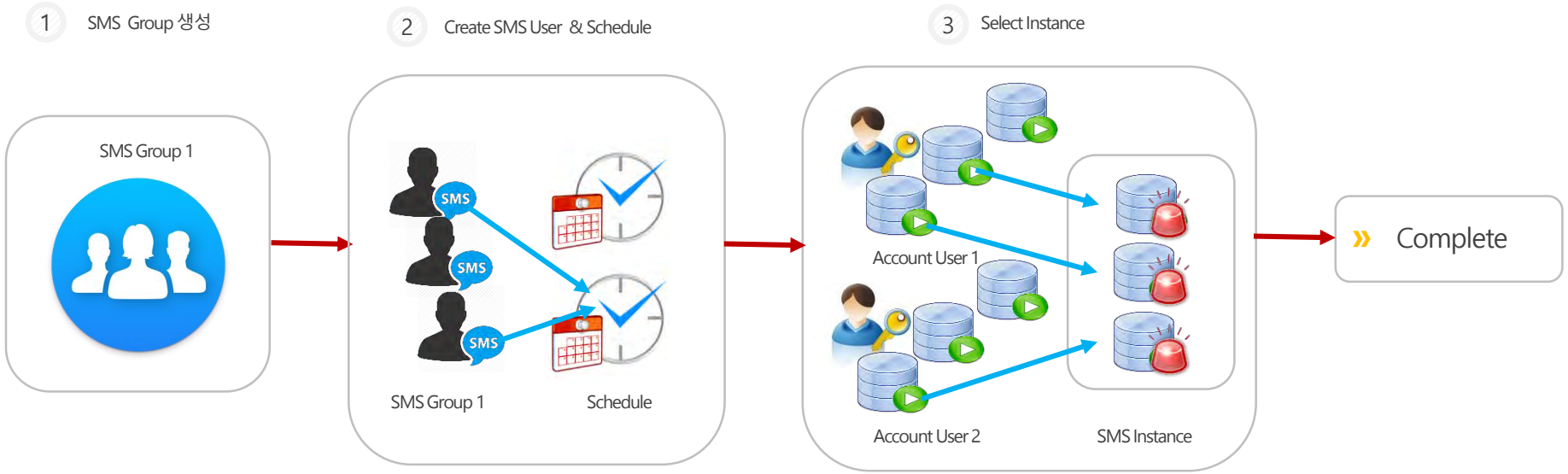
SMS / Mail 연동

» SMS / Mail Process

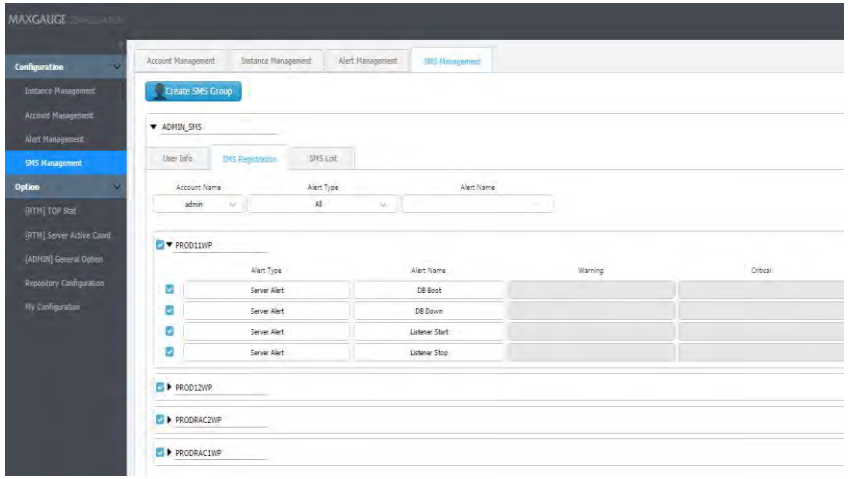


SMS / Mail 연동

» SMS / Mail 등록 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]

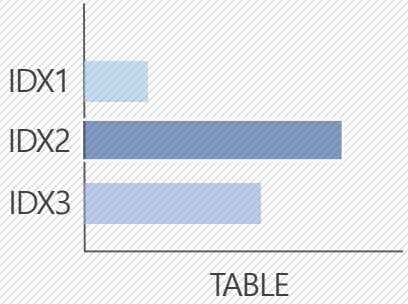
PRODUCT
DOCUMENTATION

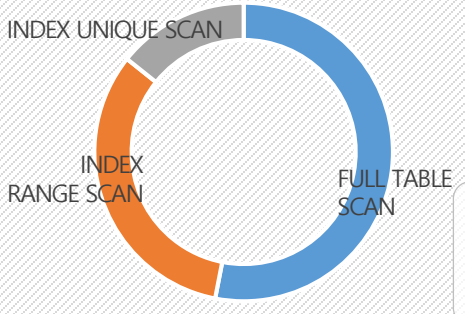
Contents

Access Statistics?

Access Statistics는 언제 쓰나요?

특정 테이블의
인덱스 사용 빈도가 궁금할 때. «

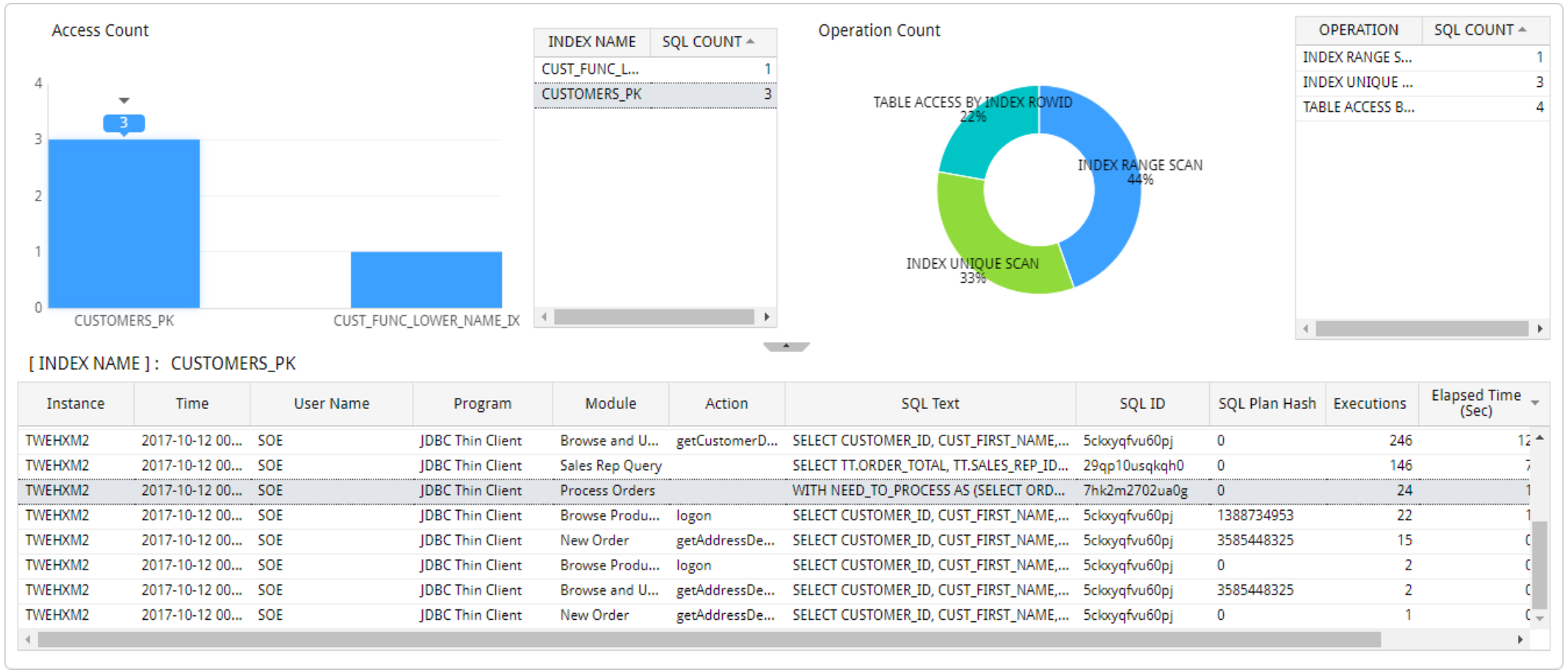




» **특정 테이블**의 오퍼레이션 별 수행 횟수를
바탕으로 튜닝 대상 선별.

Access Statistics

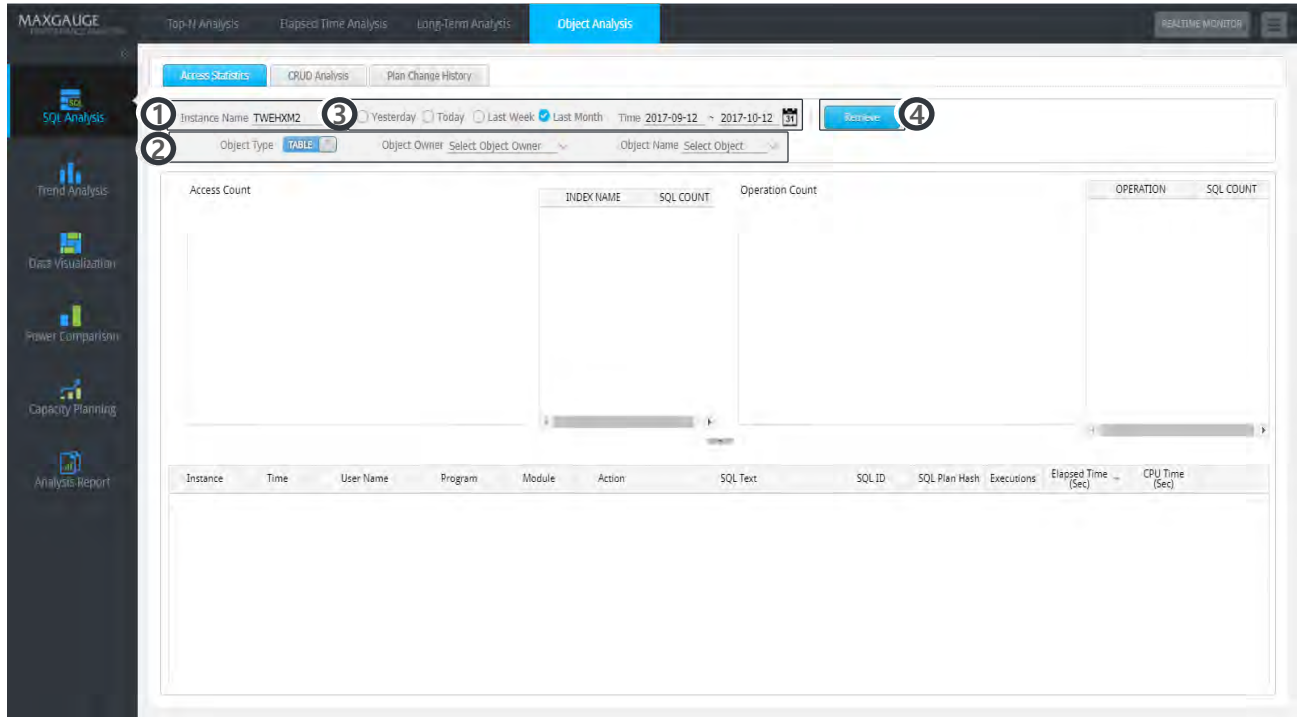
- 선택한 기간 동안 특정 오브젝트의 액세스 건 수를 확인할 수 있는 화면입니다.
- 인덱스 / 오퍼레이션 별 액세스 건 수를 그래프와 그리드로 표현해줍니다.
- 그래프(또는 그리드)를 클릭하면, 지정된 인덱스(오퍼레이션)를 사용하는 SQL 들이 하단 그리드에 나열되어 각 SQL의 세부정보를 확인할 수 있습니다.
- 해당 기능은 Performance Analyzer ▸ **SQL Analysis** ▸ **Object Analysis** 경로를 통해 사용할 수 있습니다.



Access Statistics의 활용

Access Statistics의 사용 방법

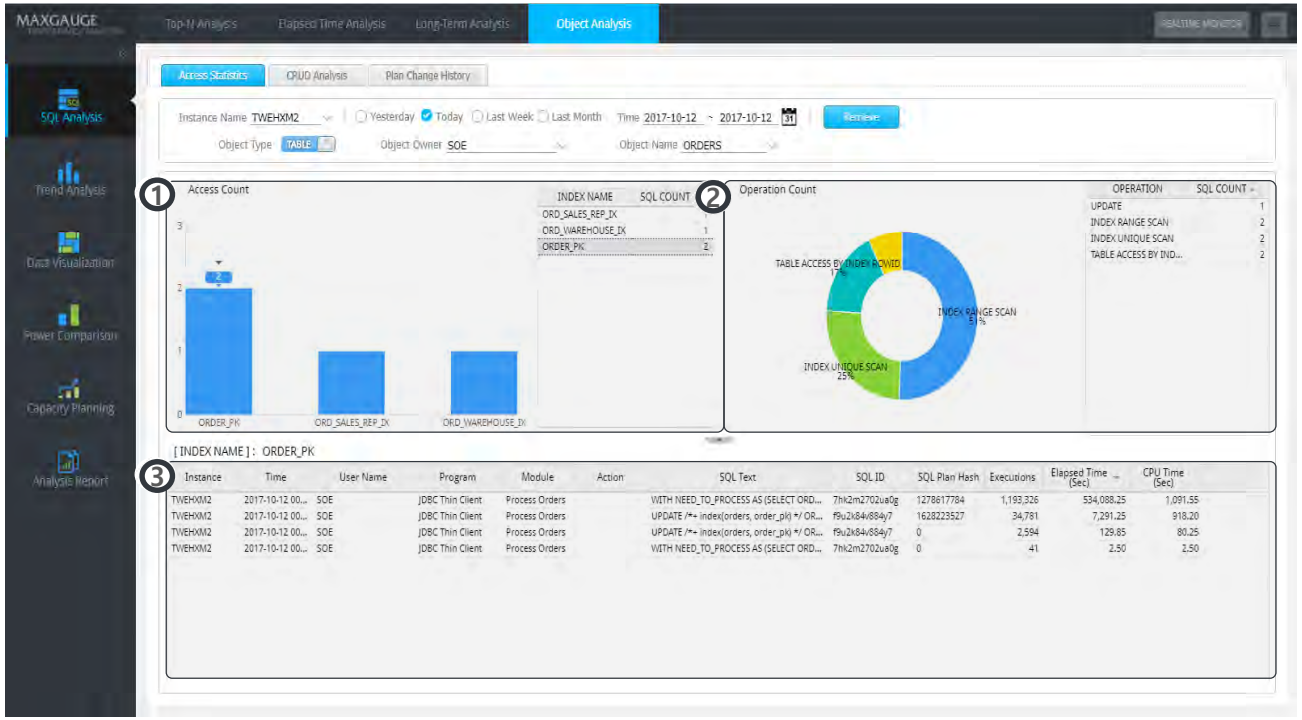
Access Statistics 검색 화면



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 실행**합니다.

Access Statistics의 사용 방법

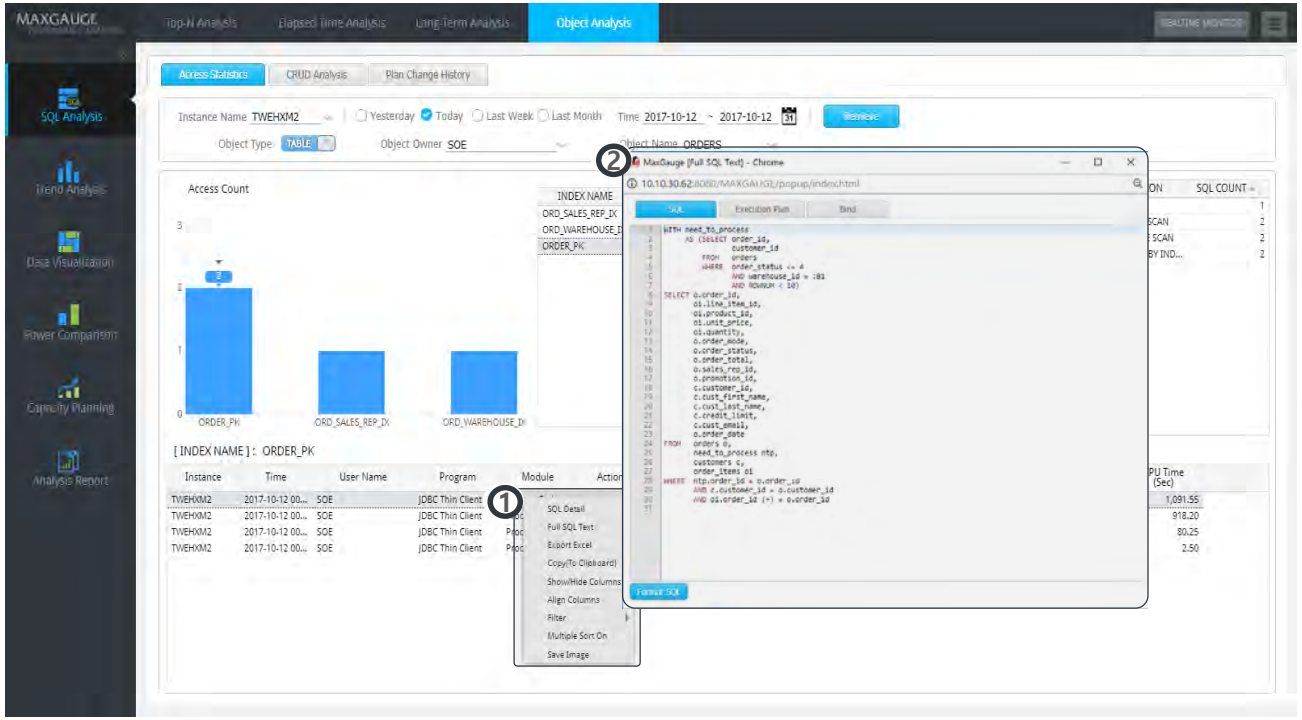
Access Statistics 검색 화면



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 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, 바인드변수 정보를 확인할 수 있습니다.

실전 분석 사례 Case 1.

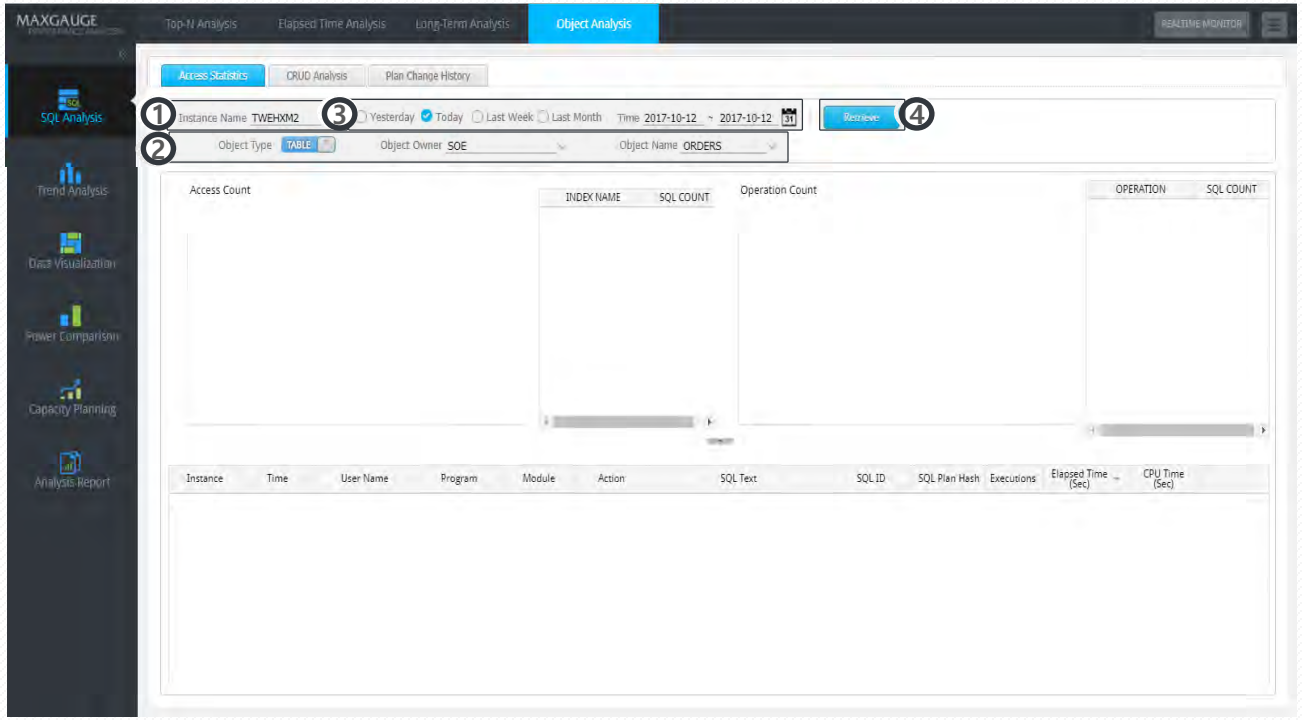
테이블 내 미사용 인덱스를 확인하고 싶어요

STEP

1. 검색 조건 설정
2. 데이터 분석
   - 검색 결과 확인
   - 메타정보와 비교하여 미사용 인덱스 검출

" 테이블 내 미사용 인덱스를 확인하고 싶어요 "

Access Statistics 검색 화면



✓ **시나리오**

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

✓ **목표**

최근 한달 지정된 테이블이 포함된 SQL 에서 어떤 인덱스를 사용했는지 확인하고, 미사용 인덱스를 제거하여 여유 공간을 확보한다.

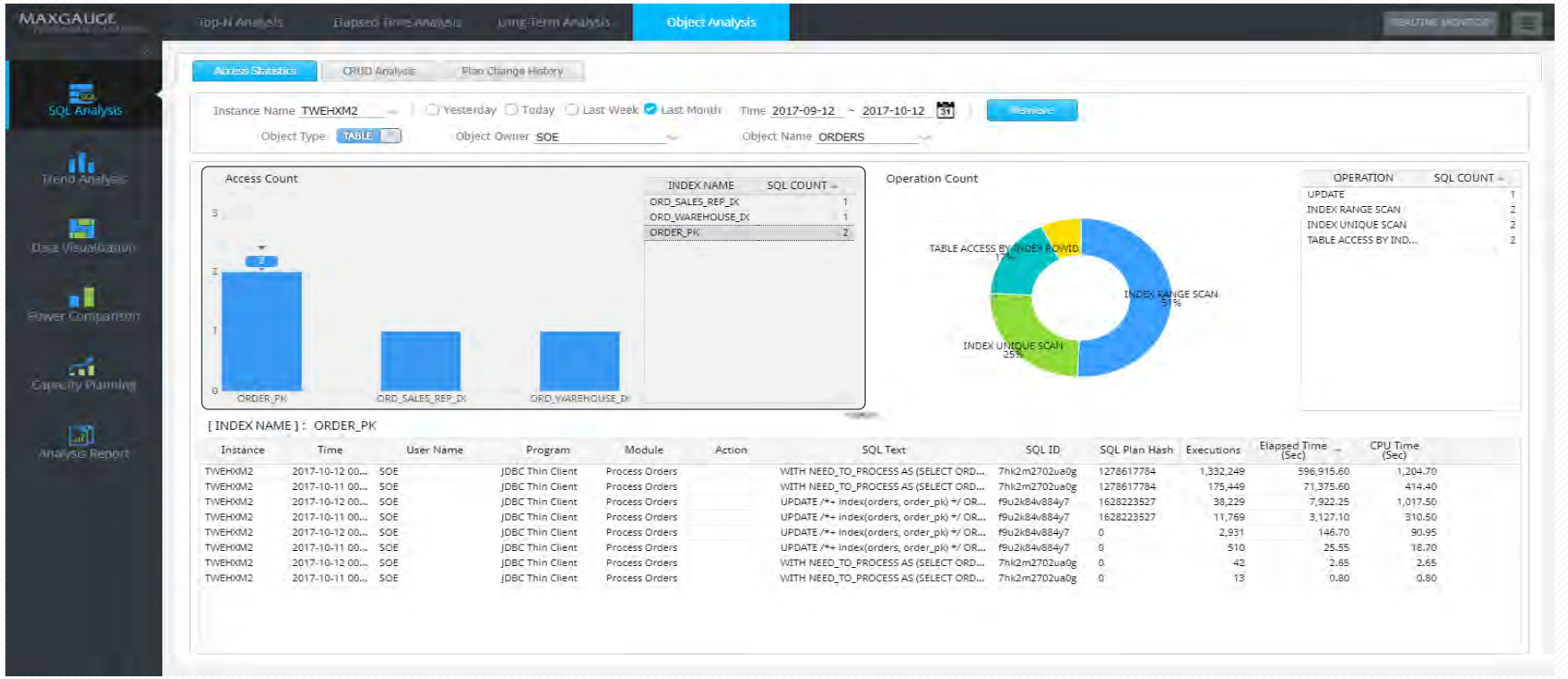
1. 비교 대상 Instance (**TWEHXM2**)를 선택합니다.
2. Object Type 을 TABLE 로 놓고, Object Owner 와 Object Name 을 선택합니다.
3. 검색 기간을 최근 한 달(**Last Month**)로 선택합니다.
4. 위의 설정 조건으로 **Access Statistics를 실행**합니다.

" 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



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

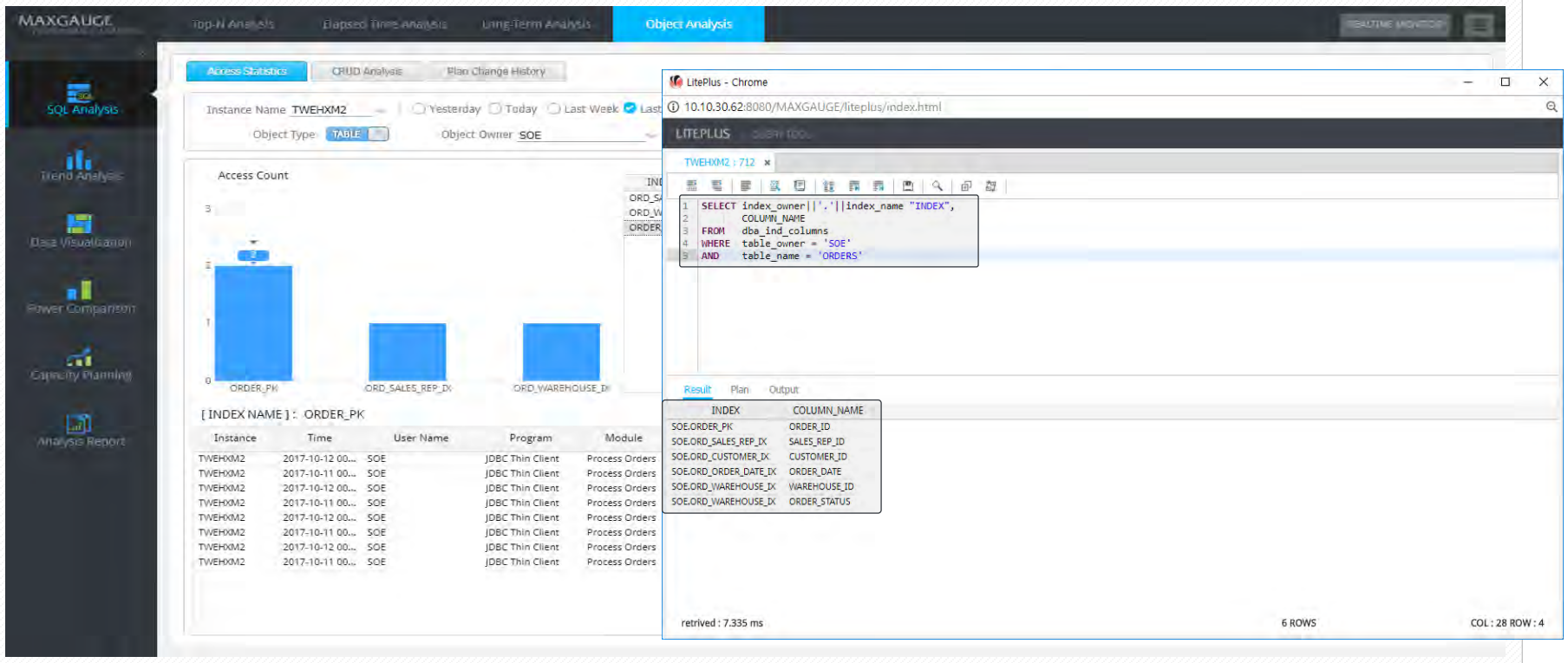
✓ 실제 테이블에 속한 인덱스와 비교하기 위해, 단축키(**Ctrl+L**) 을 눌러 제품에 내장된 Query Tool 을 호출합니다. (계속)

" 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



✓ 팝업 형태로 호출된 Query Tool에서 SQL 을 이용해 해당 테이블의 전체 인덱스 정보를 조회합니다.

✓ Query Tool에서 조회된 결과(5건)와 제품화면의 인덱스(3건)를 비교하여 미사용 인덱스를 간추립니다.

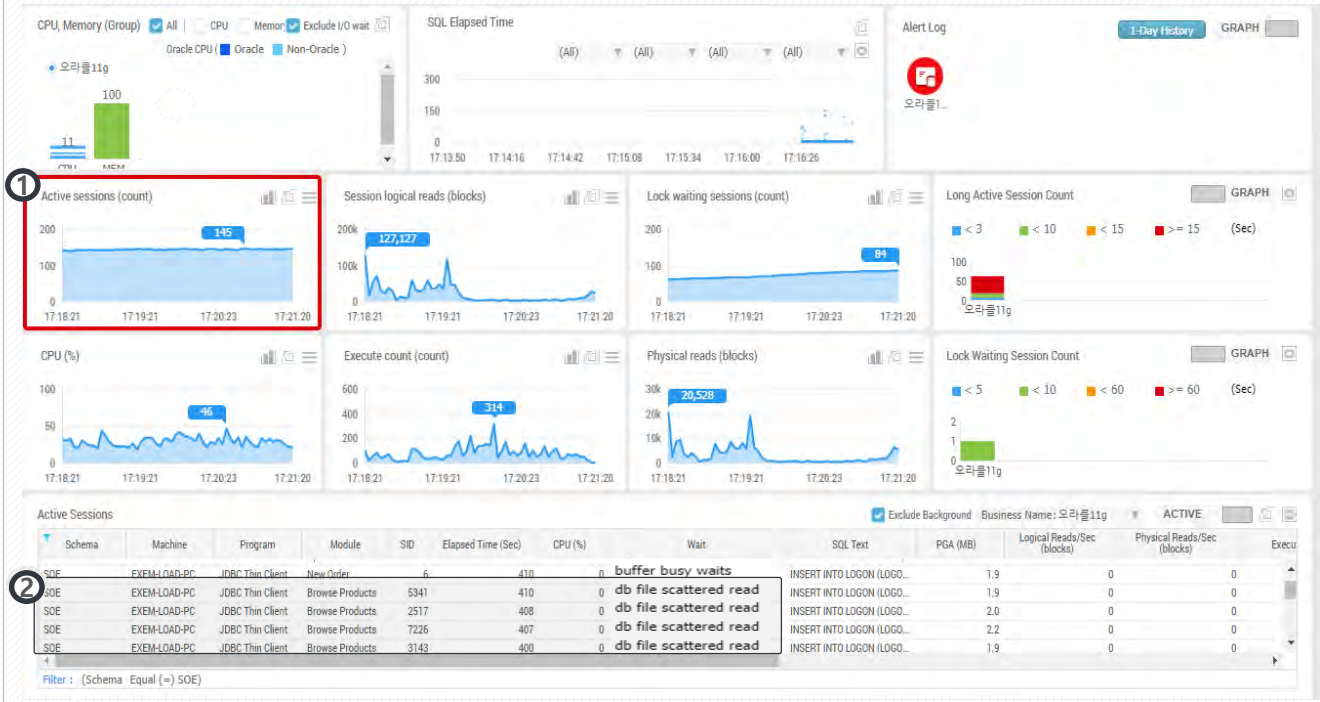
✓ 업무적인 확인 절차를 거쳐 사용되지 않는 인덱스임이 확인되면, 간추려진 인덱스를 Drop 합니다.

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 "

Access Statistics 검색 화면



✓ **시나리오**

INSTANCE : TPROD11WO
SCHEMA : SOE
TABLE : ORDERS

✓ **목표**

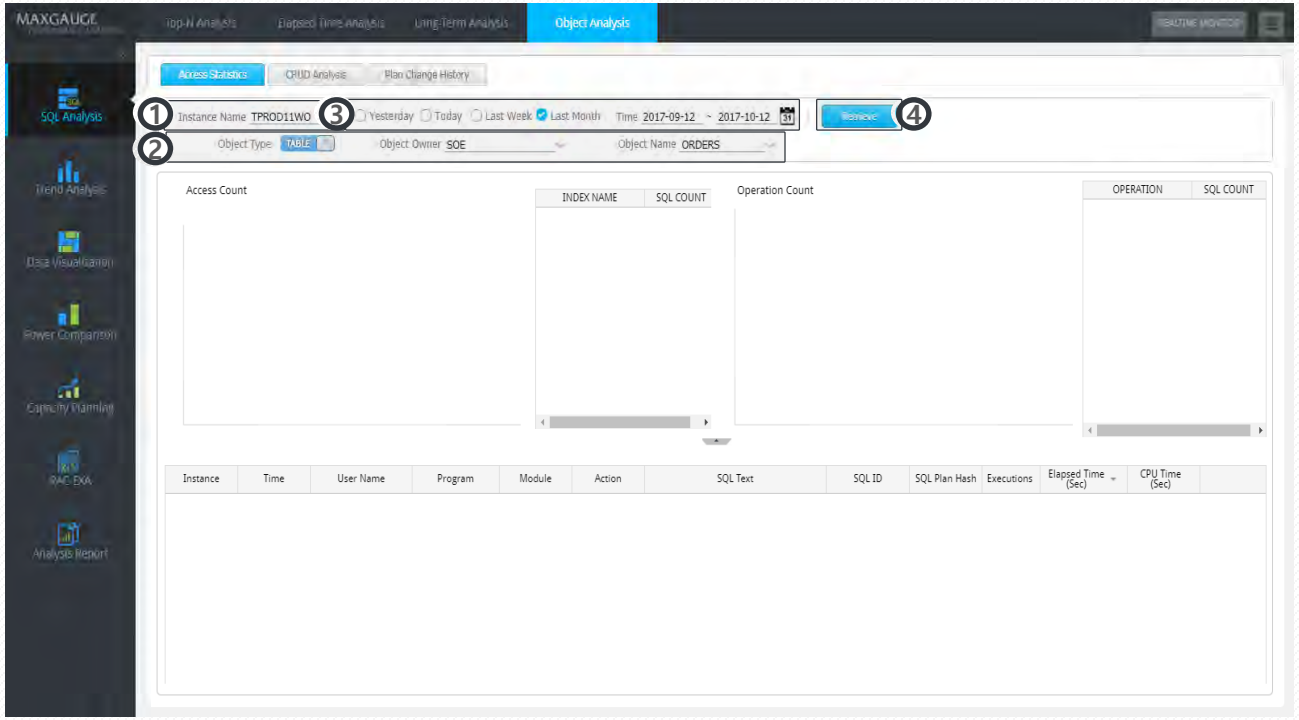
특정 테이블을 FULL TABLE SCAN 으로 수행하는 SQL 의 리스트를 확보하여 튜닝 여부를 판단

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.

" 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 "

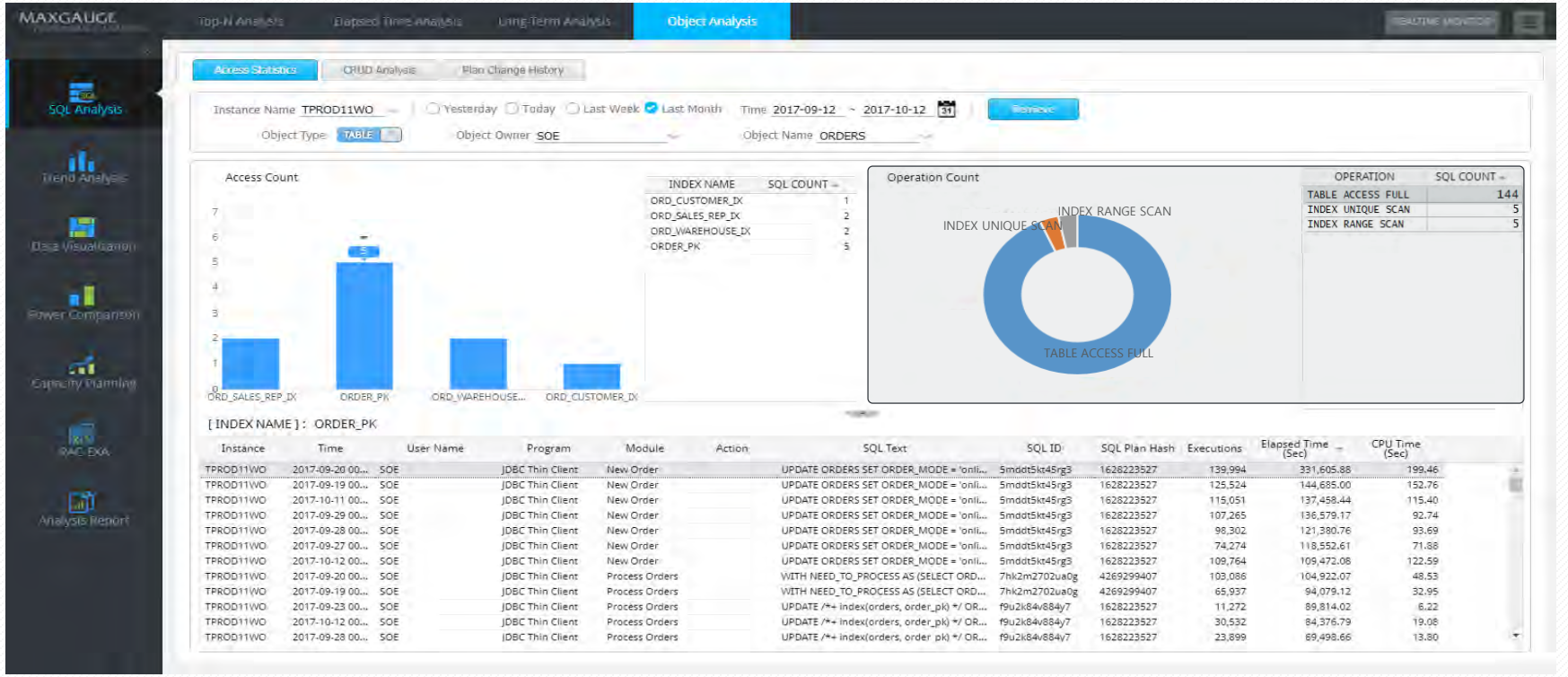
Access Statistics 검색 화면



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 Statistics를 실행**합니다.

" 지정된 테이블을 FULL TABLE SCAN 하는 SQL 목록을 뽑아내고 싶어요 "

» Access Statistics 조회 화면

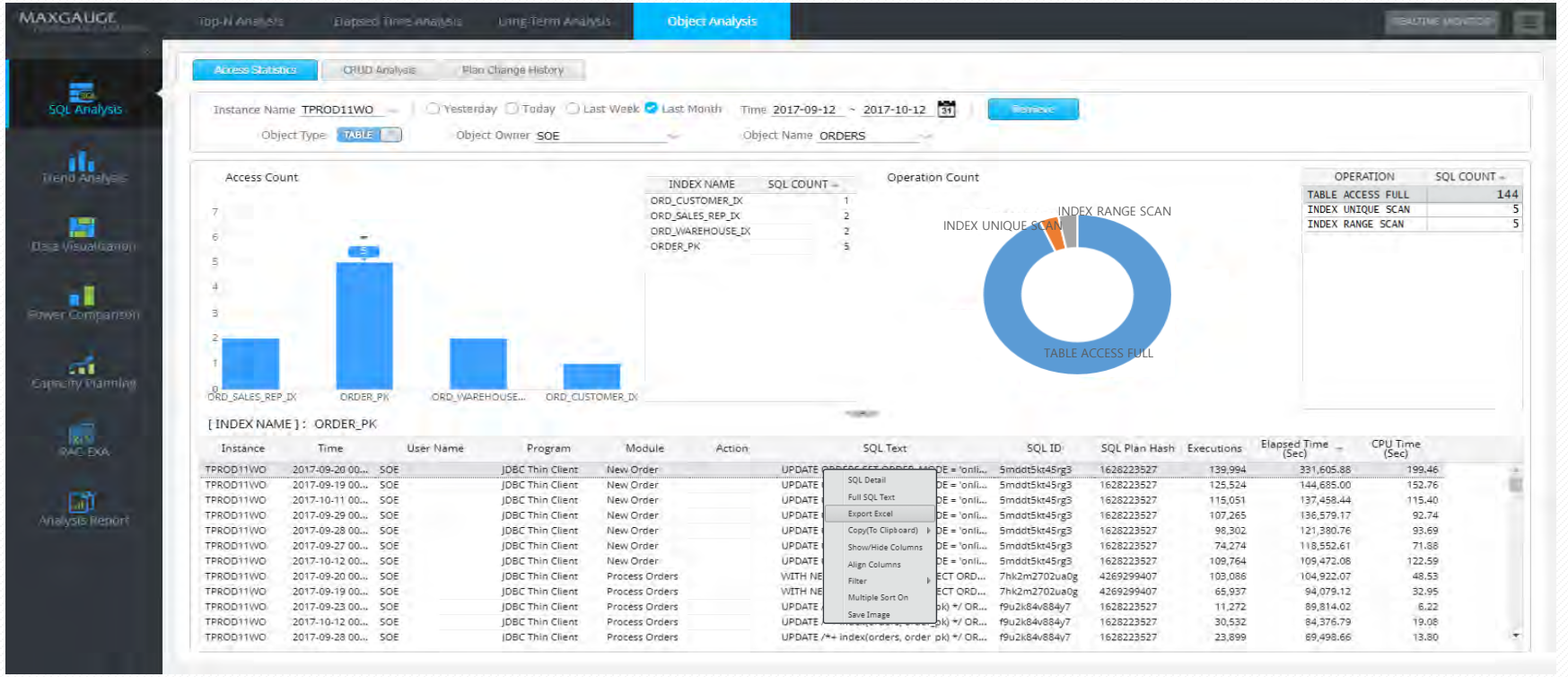


✓ 결과 화면의 중앙 우측 Operation Count 창에서 선택한 테이블이 어떤 Operation 으로 수행하였는지 표시됩니다.

✓ 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 "

» Access Statistics 조회 화면



✓ 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?

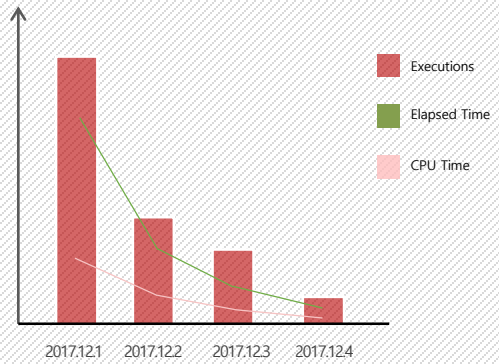
Application Call Tree는 언제 쓰나요?

특정 화면 및 업무
➔ 식별하기 위한 규칙 표준화(**식별자**)
➔ **식별자**를 SQL 주석에 기재

IO MODULE **METHOD**

```
SELECT /* MAINPAGE SUBPAGE1 */
       PARTITION_KEY,
       DB_ID,
       .....
FROM   ORA_OS_STAT_10MIN A ,
       ORA_SQL_STAT_10MIN B
WHERE ...
GROUP BY ...
```

적용된 **식별자** 및 **애플리케이션 정보**를 이용하여
해당 업무, 화면에서 수행된 SQL 의
수행 이력을 확인(분석) 할 수 있습니다.



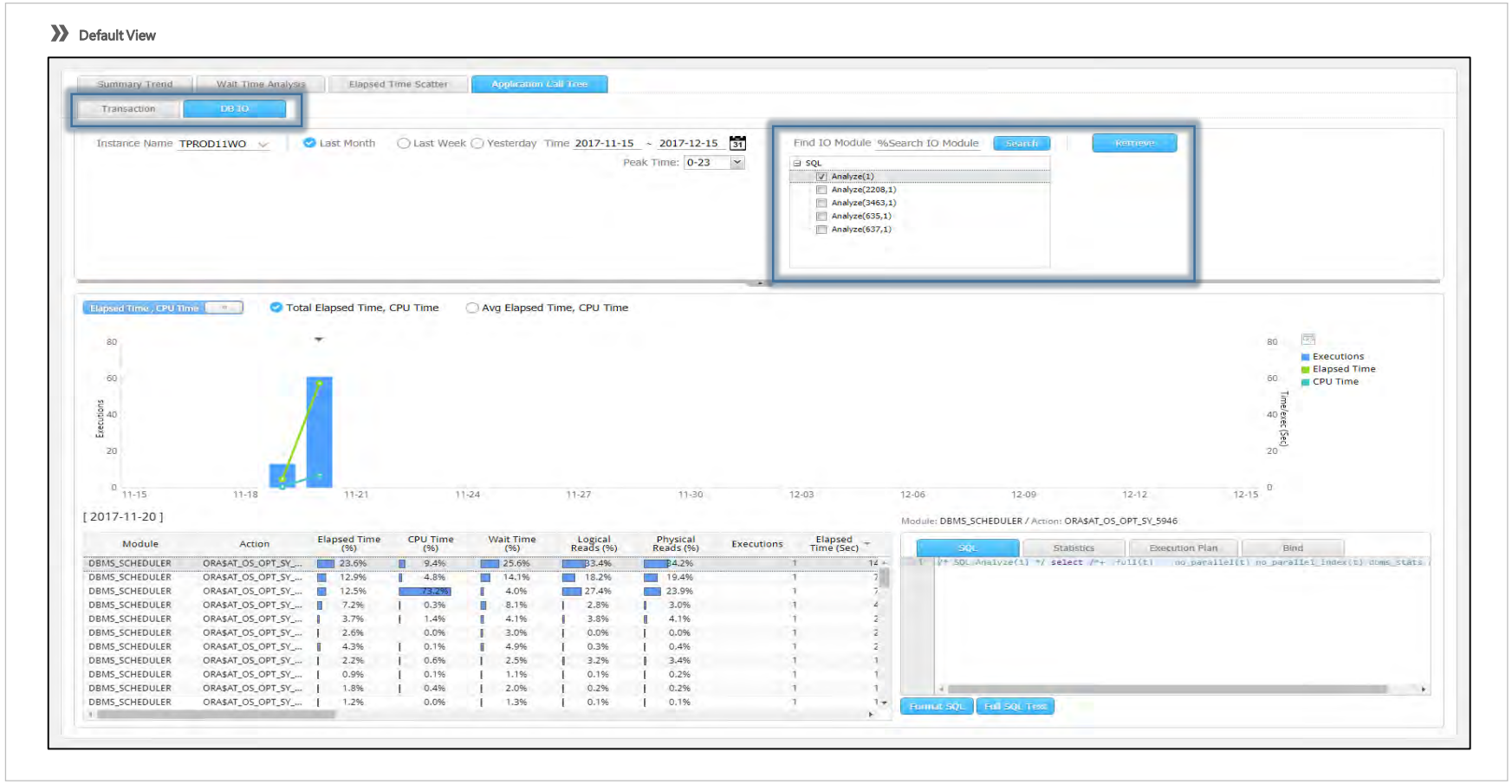
Application Call Tree

- 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 Tree조회 화면

① 대상 Instance를 선택합니다.

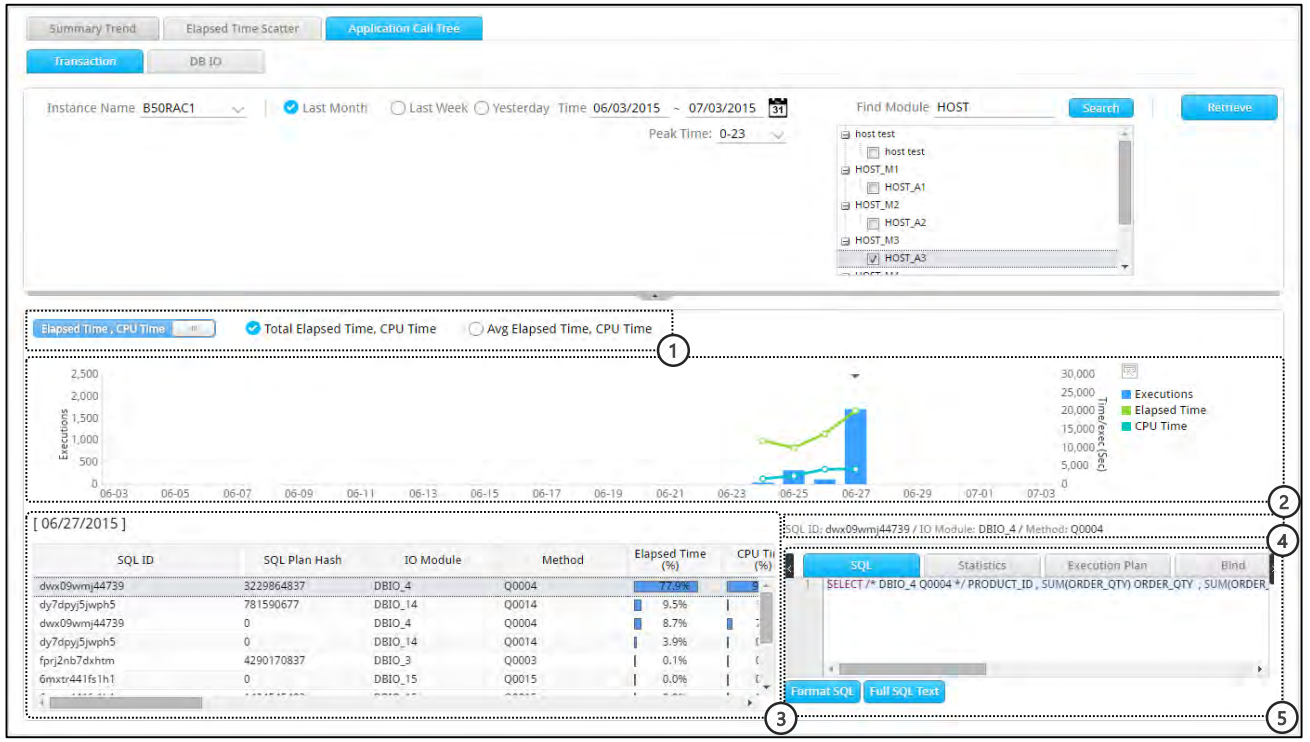
② 분석하고자 하는 기간과 시간을 설정합니다.

③ Repository에 저장된 **식별자**(IO Module, Method) 및 **Module, Action**에 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 Tree조회 화면(Transaction)



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.

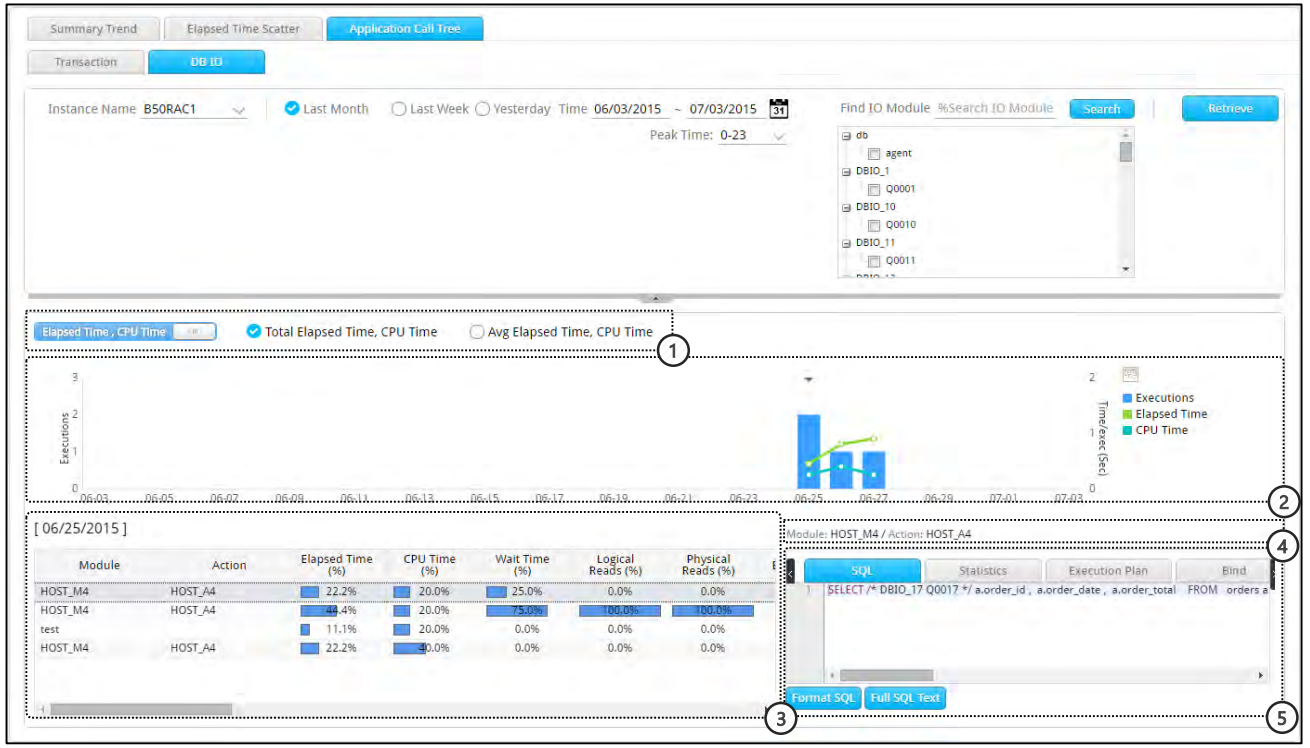
③ 선택된 날짜의 SQL 개별 일량을 확인합니다.

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 Tree조회 화면(DB IO)



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.

" 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

» Performance Analyzer 화면

✓ **업무 시스템 부하 원인 확인**

PA – 특정 업무가 느려진 날짜의 **CPU 증가 시점에 IO Class Wait이 증가**한 것을 확인.

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

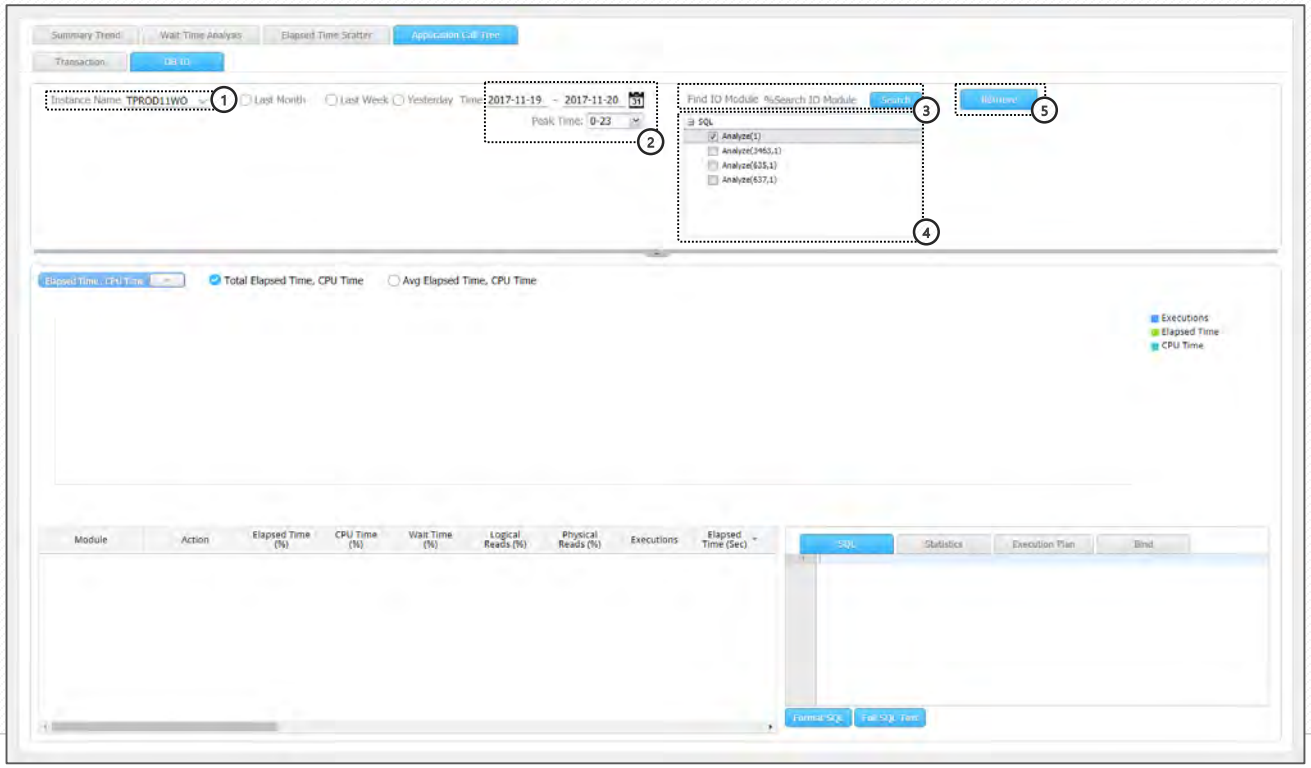
② 동일 시점에 IO Class Wait이 증가한 것을 확인할 수 있습니다.

" 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

Application Call Tree 검색 화면



✓ 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.

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.

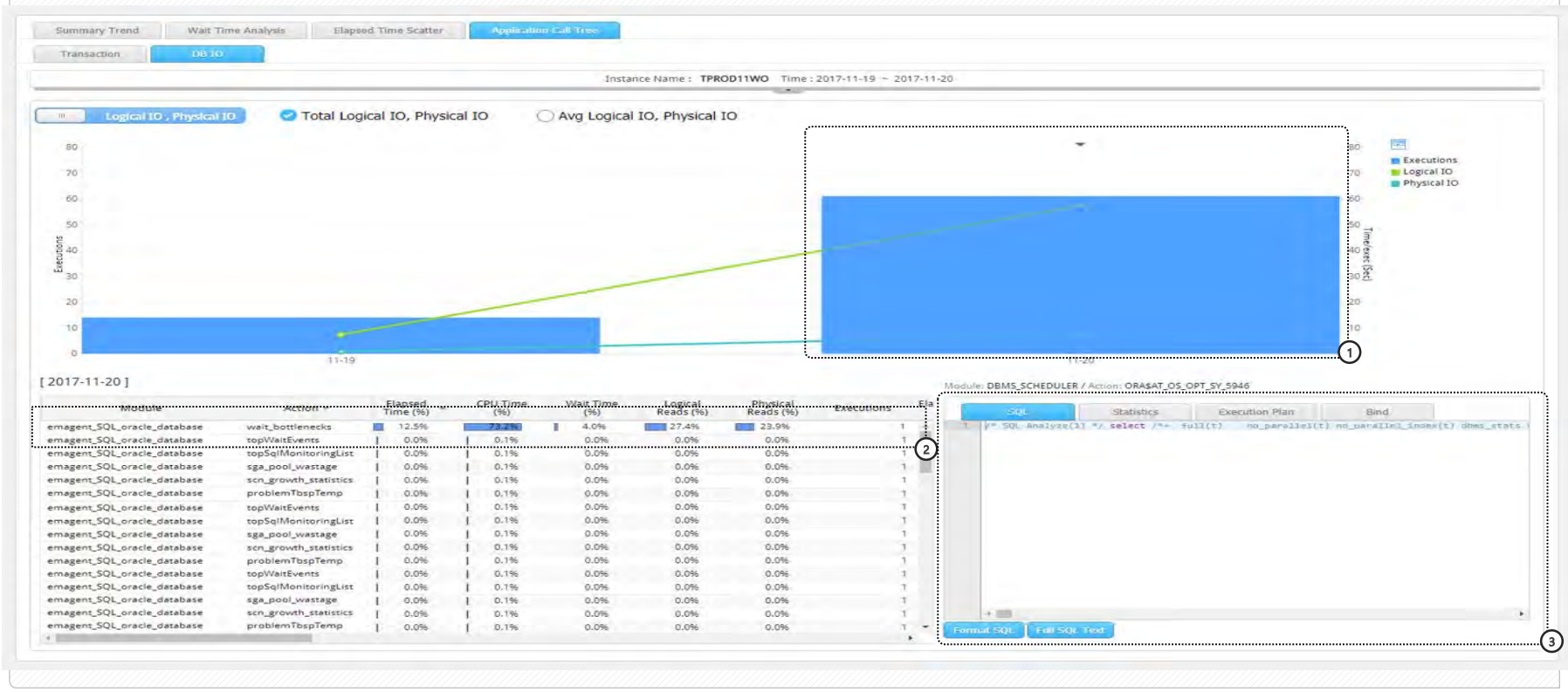
⑤ 위의 설정한 조건으로 **Application Call Tree를 실행합니다.**

" 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

» Application Call Tree 조회 화면



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.

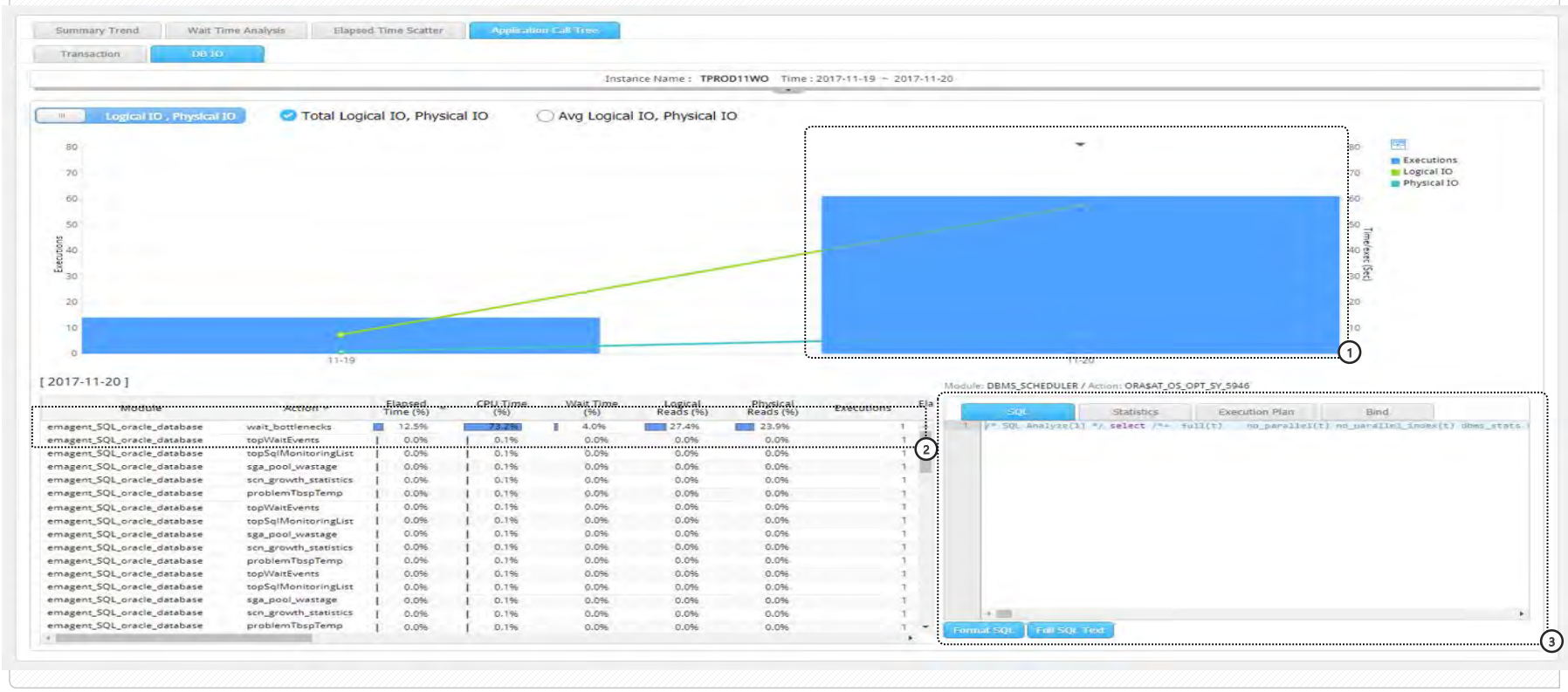
③ 추출된 SQL에 대한 SQL TEXT, Statistics, Execution Plan, Bind 변수를 확인할 수 있습니다.

" 특정업무의 SQL 수행내역을 확인하고 싶습니다."

STEP

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

» Application Call Tree 조회 화면



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.

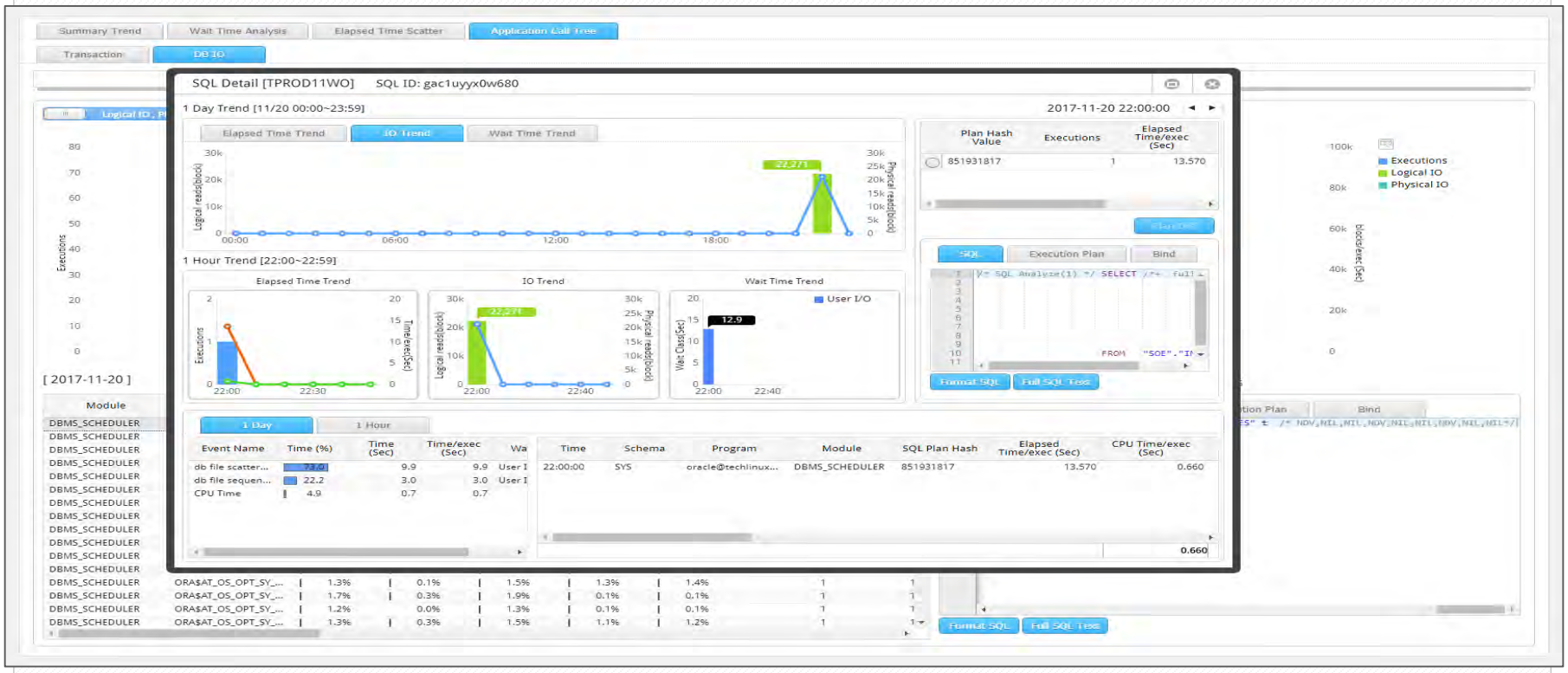
③ 추출된 SQL에 대한 SQL TEXT, Statistics, Execution Plan, Bind 변수를 확인할 수 있습니다.

" 특정업무의 SQL 수행내역을 확인하고 싶습니다."

STEP

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

» Application Call Tree 조회 화면



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

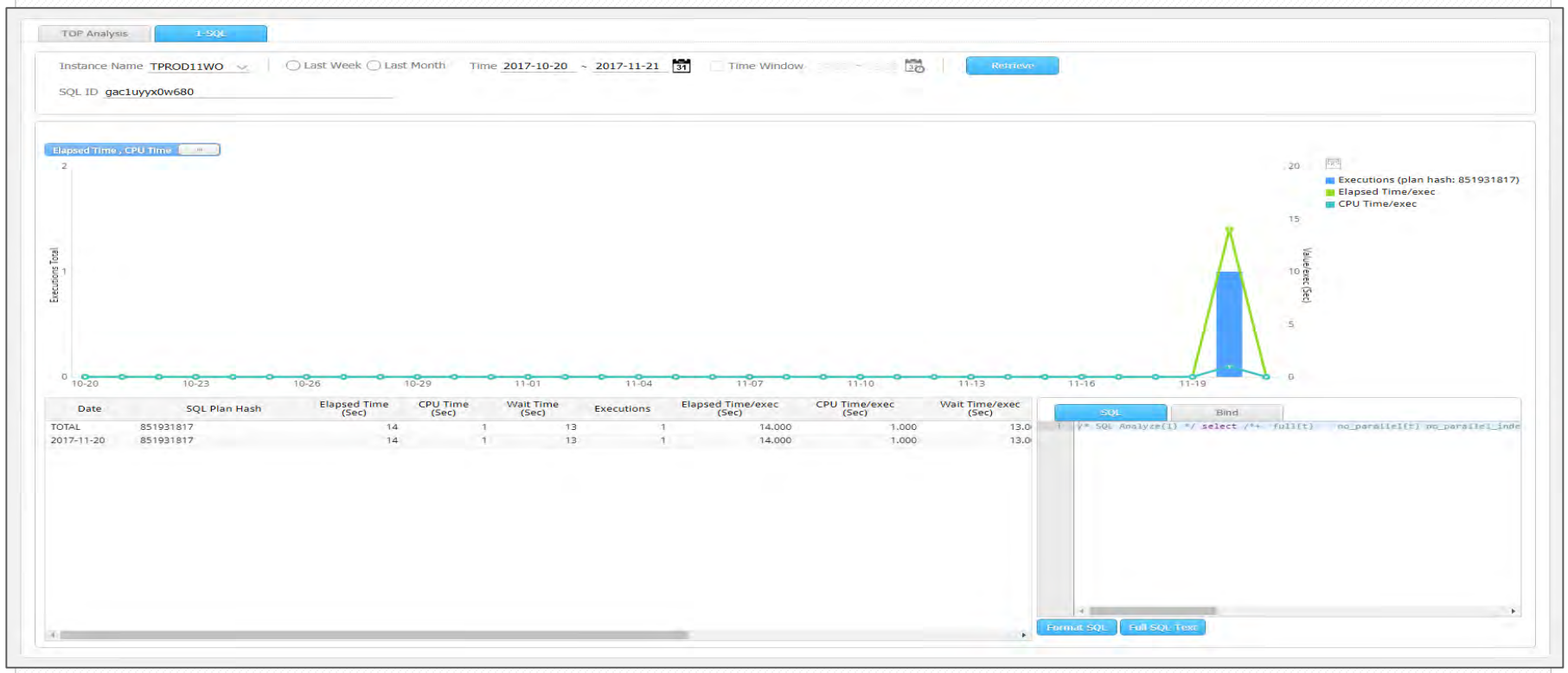
✓ 'IO Trend Tab'을 선택하면 시간대별 IO 발생 Trend를 확인할 수 있습니다.

" 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

» Long Term Analysis 연계 화면



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

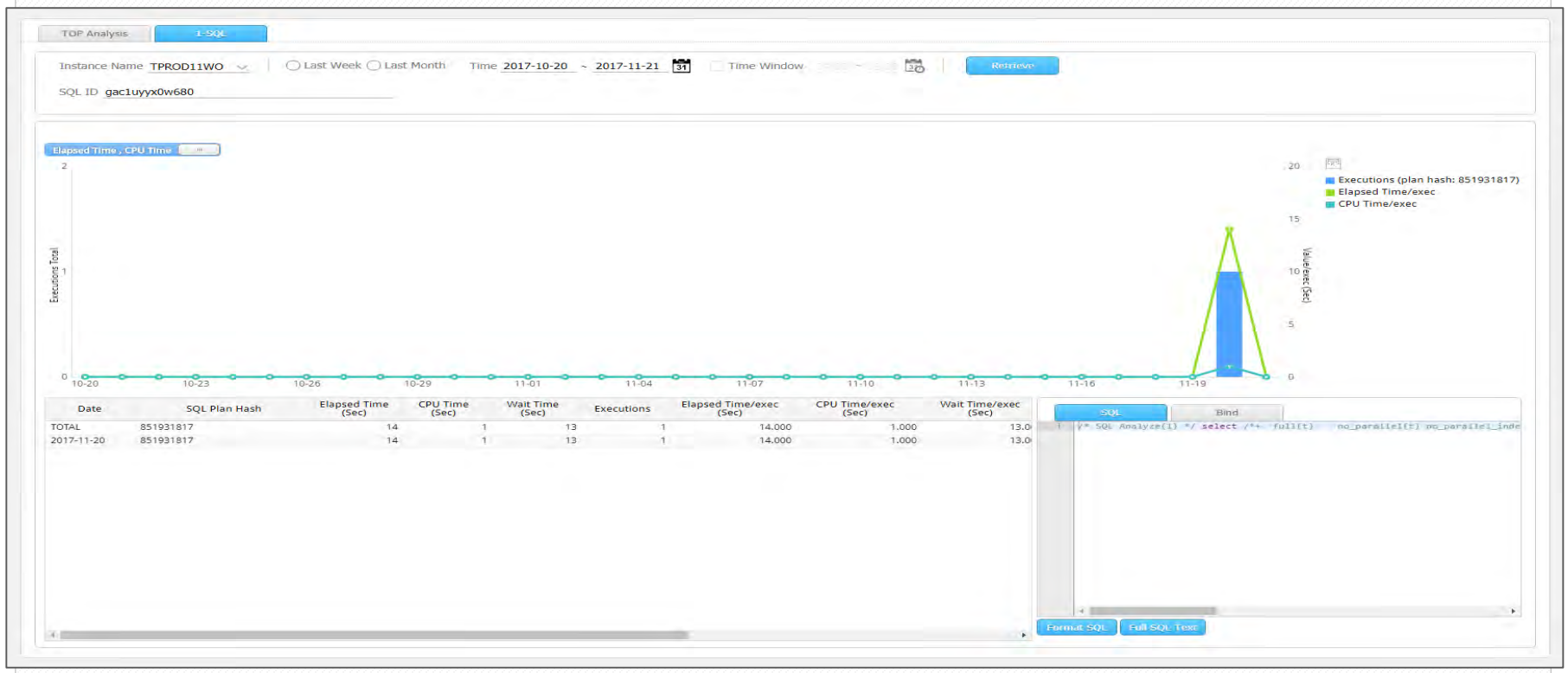
✓ 해당 SQL은 기존에 수행되지 않던 새로운 SQL로 최적화가 되지 않았기에 SQL TUNING이 필요합니다.

" 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

» Long Term Analysis 연계 화면



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

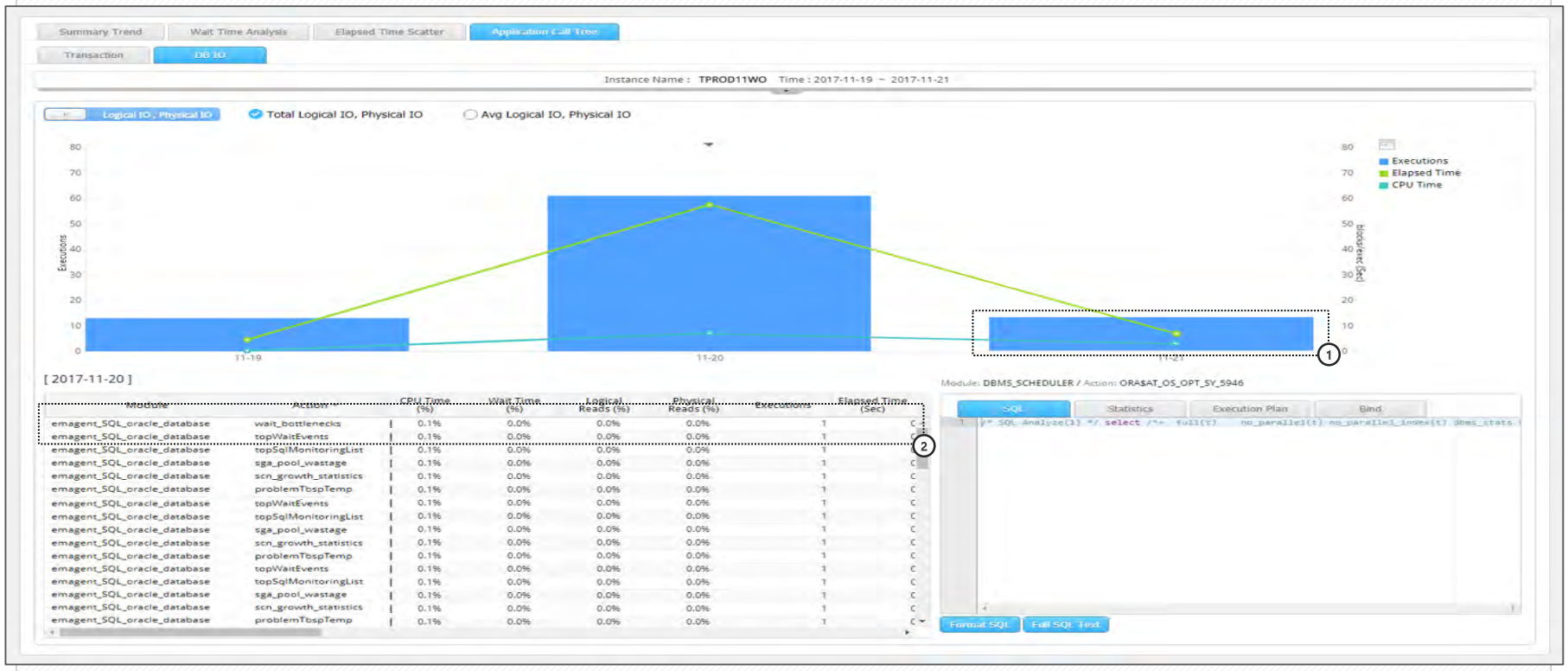
✓ 해당 SQL은 기존에 수행되지 않던 새로운 SQL로 최적화가 되지 않았기에 SQL TUNING이 필요합니다.

" 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**

» Application Call Tree 조회 화면



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]

PRODUCT
DOCUMENTATION

Practical Guide

Elapsed Time Scatter

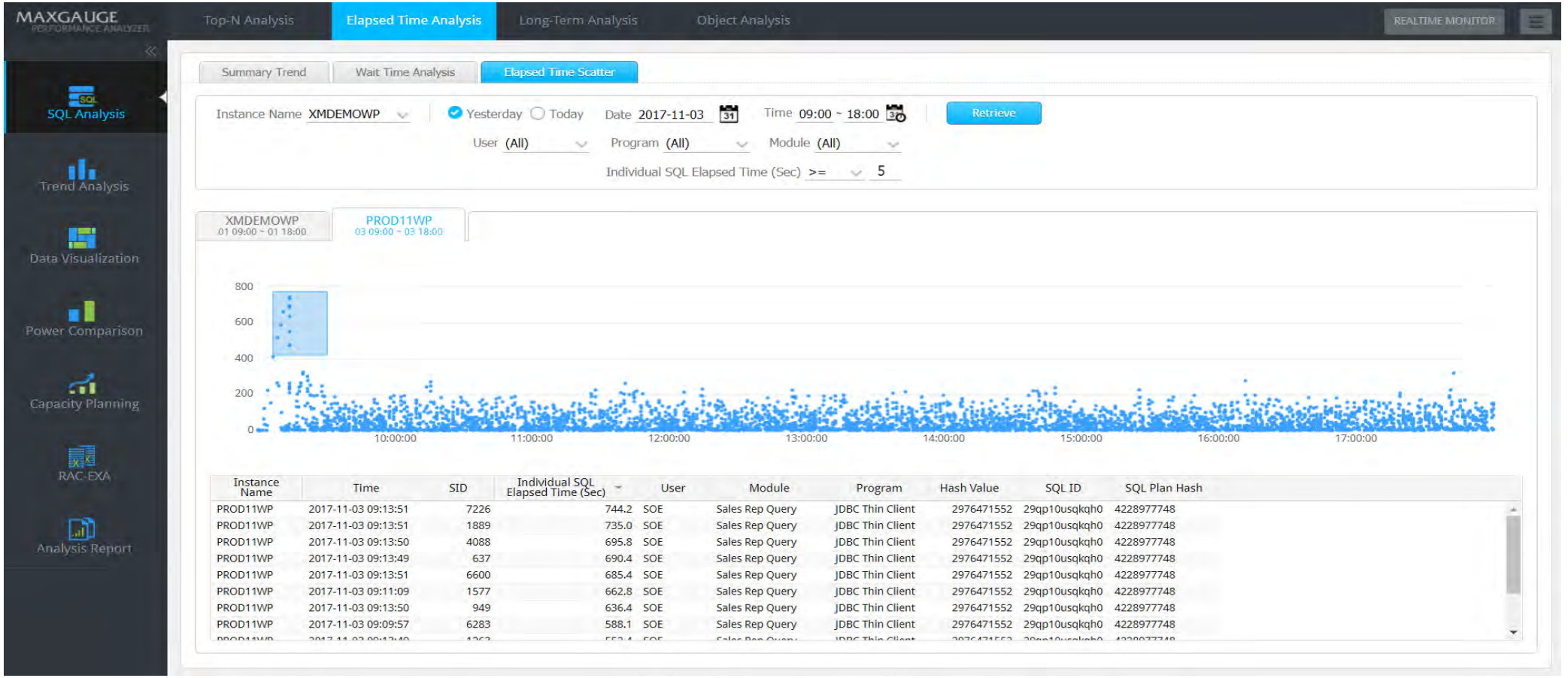
Contents

Elapsed Time Scatter?

©Copyright 2017 EXEM. All Rights Reserved.

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.



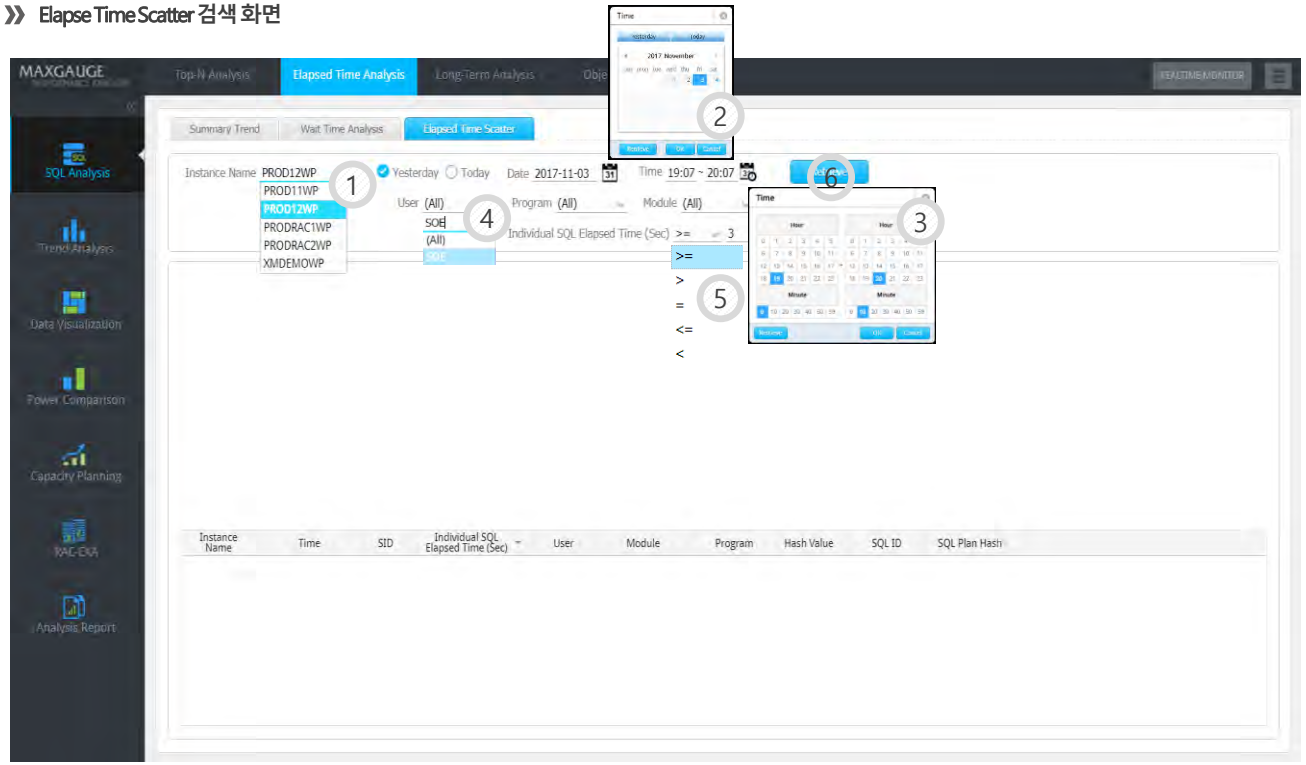
©Copyright 2017 EXEM. All Rights Reserved.

Elapse Time Scatter의 활용

©Copyright 2017 EXEM. All Rights Reserved.

Elapse Time Scatter의 사용 방법

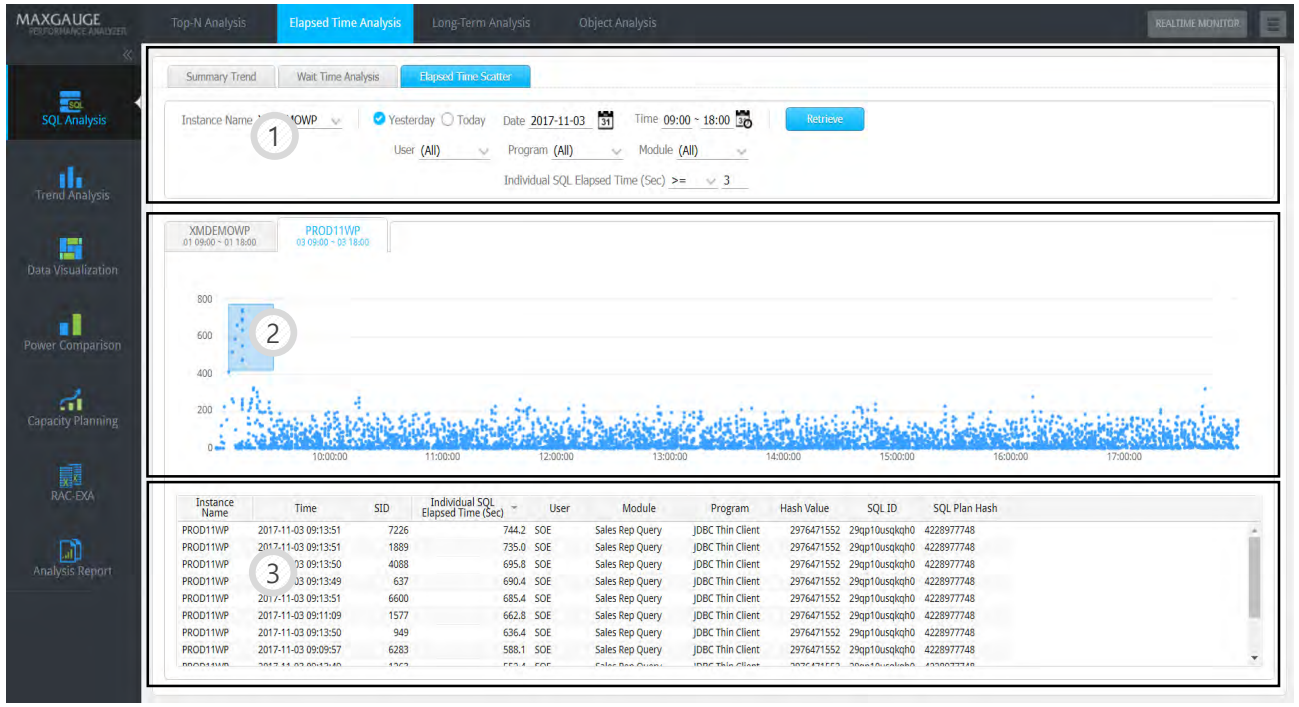
» Elapse Time Scatter 검색 화면



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

Elapse Time Scatter의 사용 방법

Elapse Time Scatter 조회화면



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

실전 분석 사례 Case 1.

5초이상 수행된 SQL중 수행시간이 긴 SQL을 분석하고 싶어요

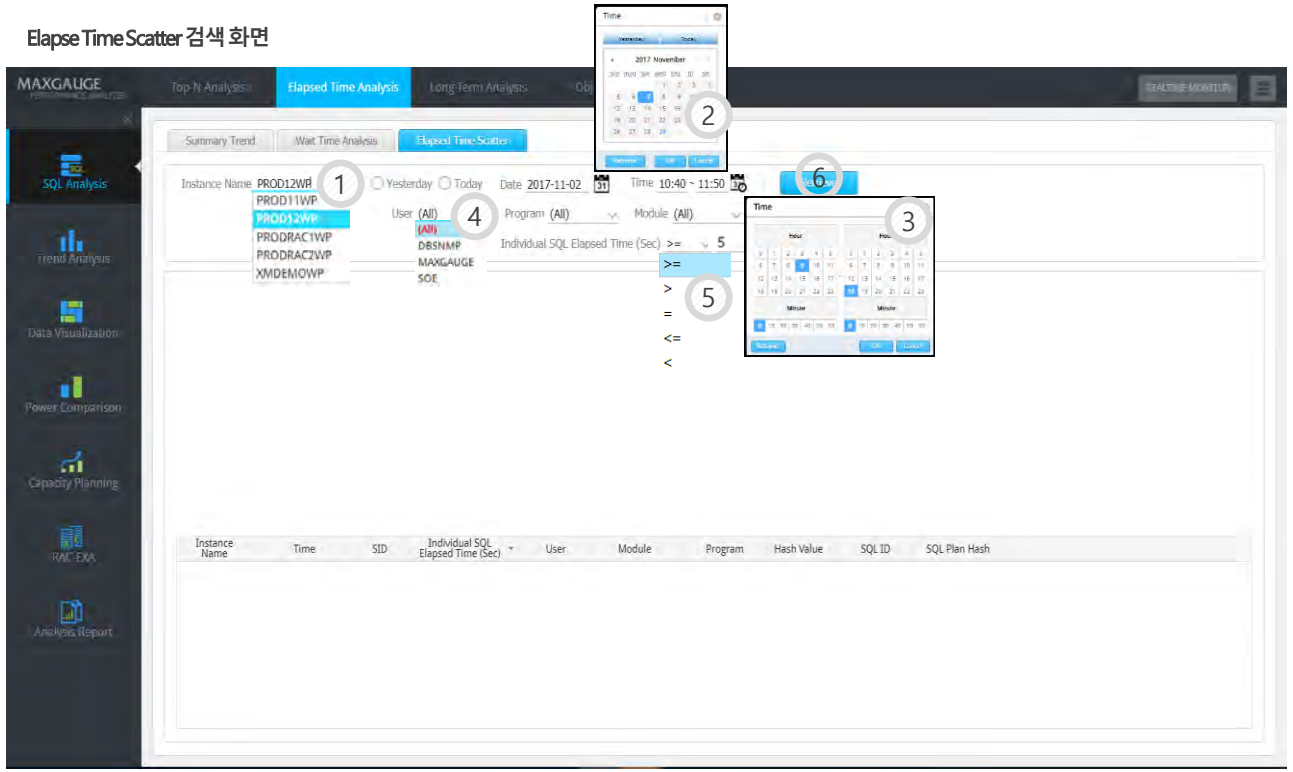
©Copyright 2017 EXEM. All Rights Reserved.

" 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

Elapse Time Scatter 검색 화면



✓ **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

1 분석대상 Instance (**PROD12WP**)를 선택합니다.

2 분석 일자를 원하는 일자인 **2017년 11월 7일**을 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.

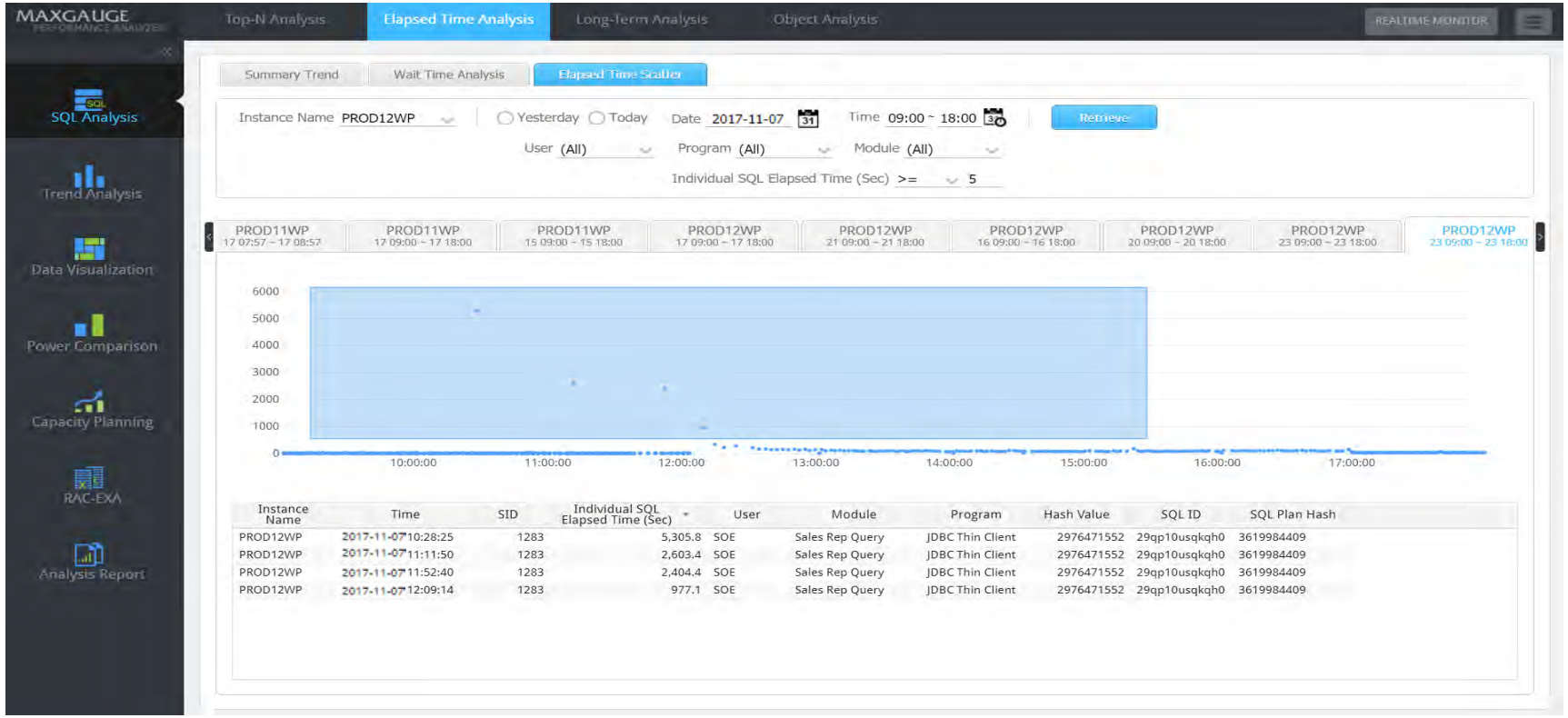
6 위의 설정한 조건으로 **Elapse scatter를 실행합니다.**

" 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

Elapse Time Scatter 검색 화면



✓ 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.

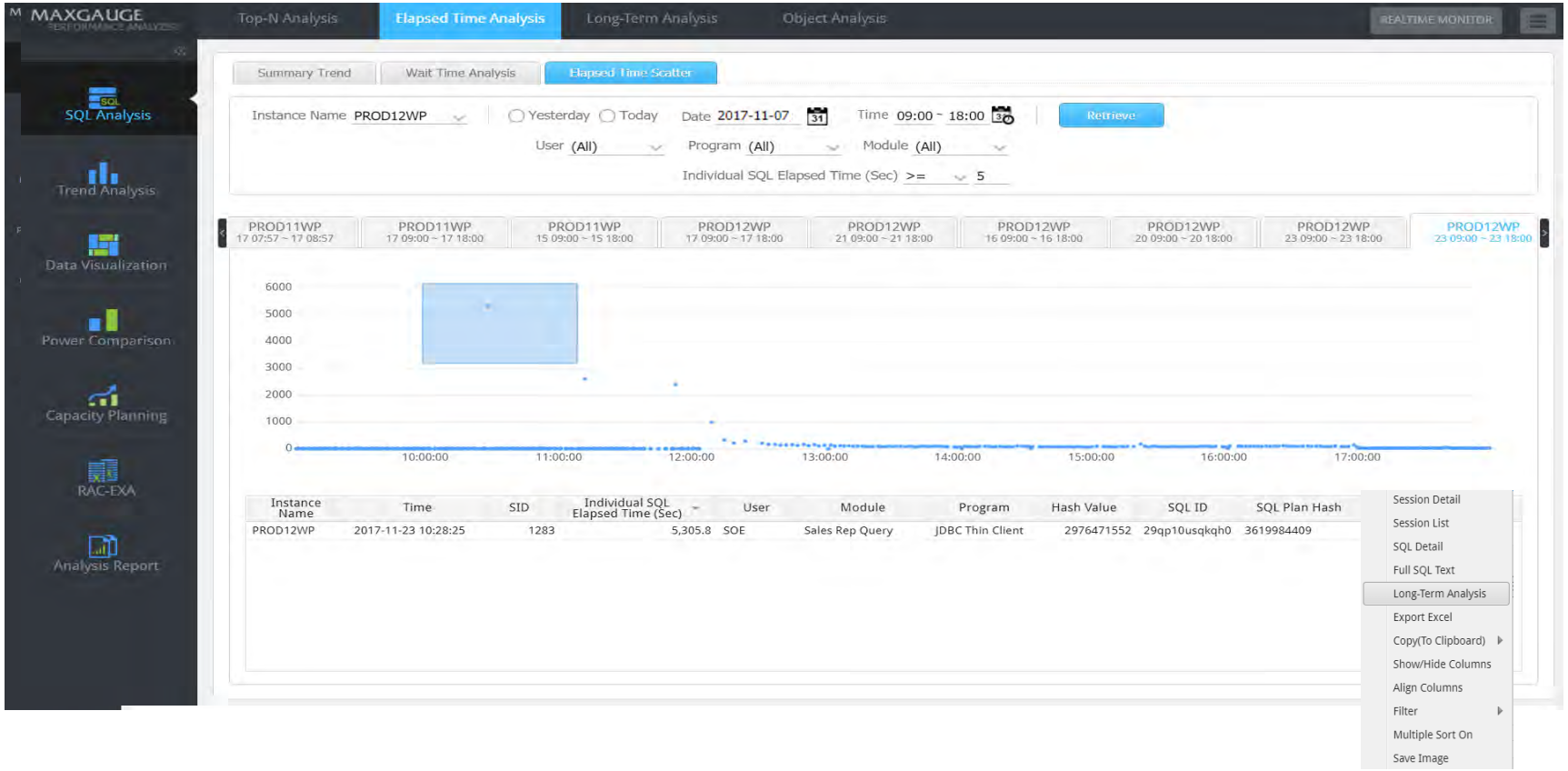
✓ 하단의 그리드는 선택된 박스 안에 해당 하는 SQL의 정보를 가지고 있습니다 (Time, SID, Elapsed Time, SQL ID, User, Module, Program 등)

" 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

Elapse Time Scatter 검색 화면



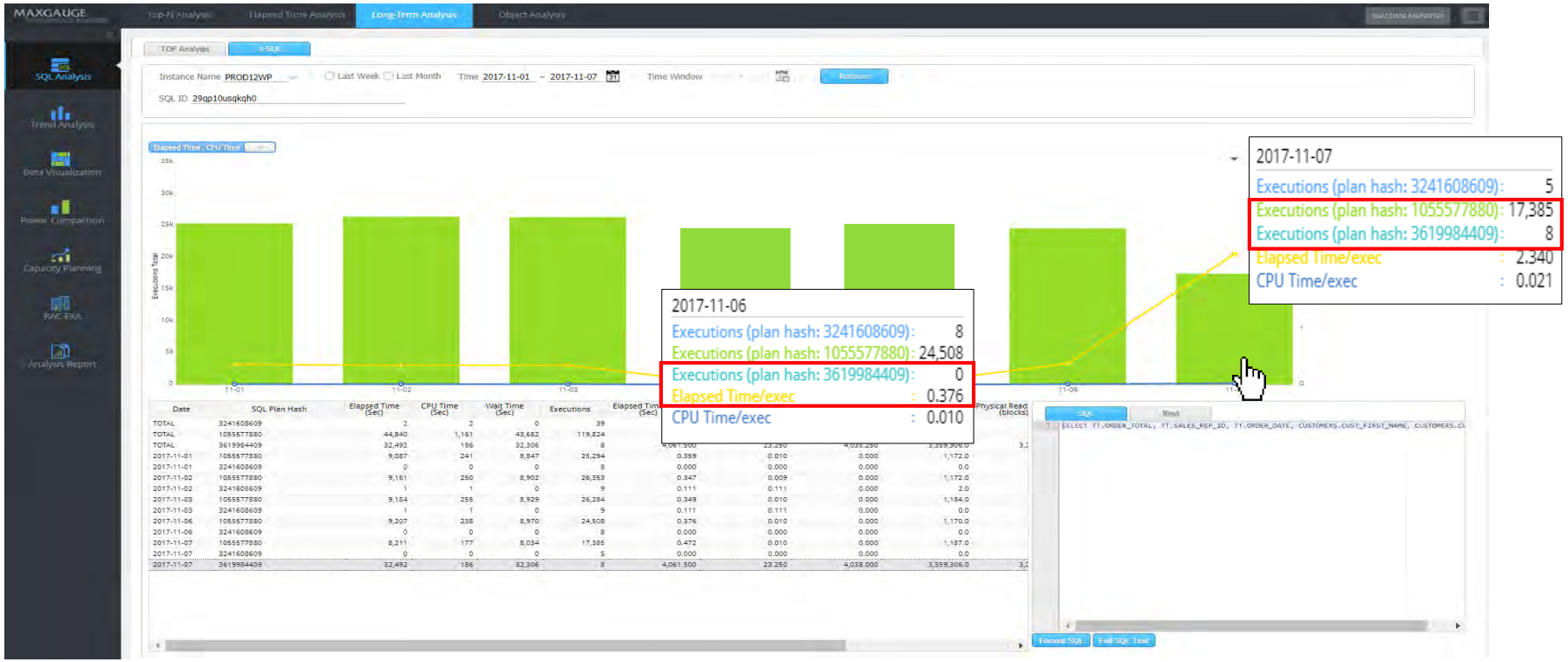
✓ Elapsed Time 시간이 5,305.8 초인 SQL의 수행이력을 확인해보기 위해 마우스 우클릭 후 Long Term Analysis 을 클릭합니다.

" 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.

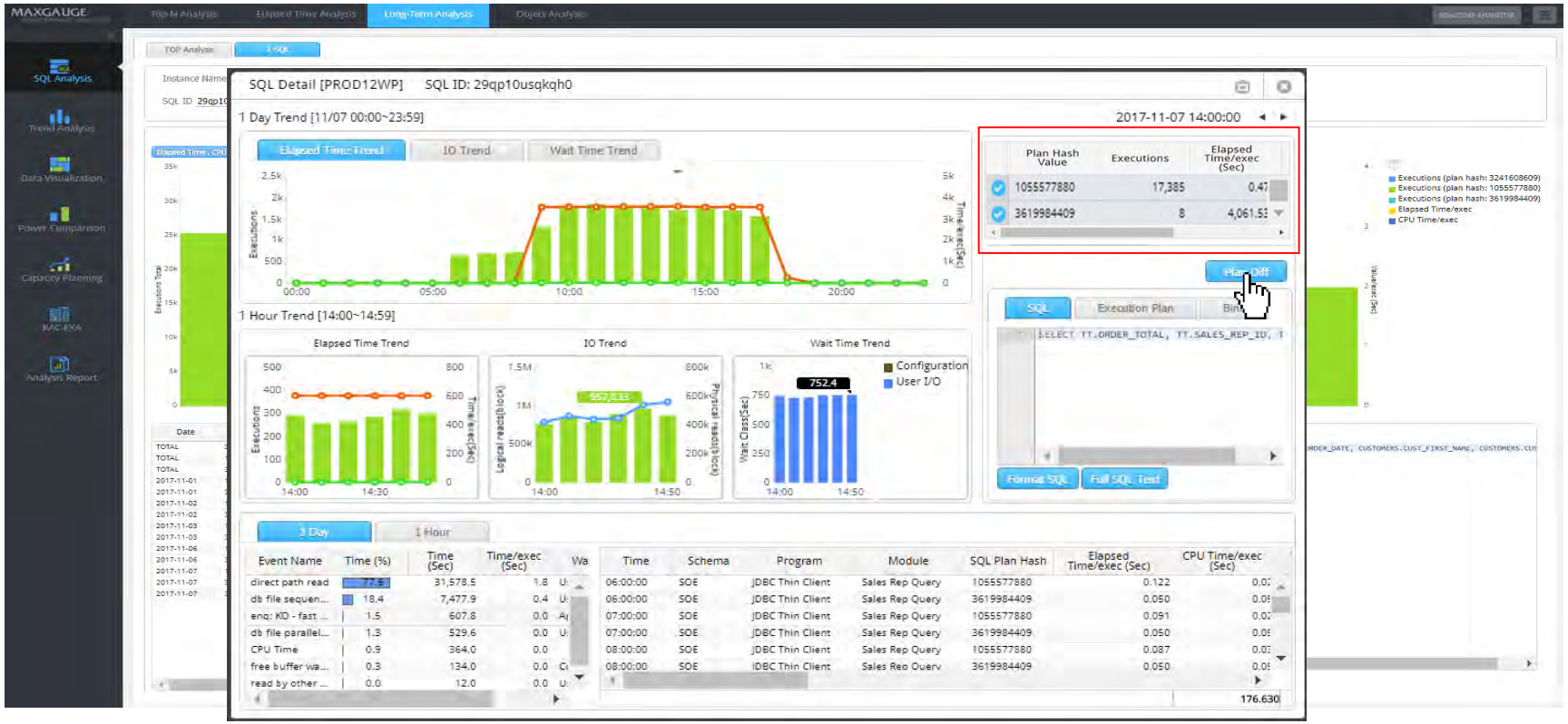
✓ 그럼 SQL Plan Hash 별 일량 및 플랜을 확인하기 위해 막대를 클릭하고 SQL Detail로 이동합니다.

" 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 Detail 연계화면



✓ 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 105557780은 1회 수행당 약 0.47초지만 3619984409은 4061초가 걸립니다. 즉 3619984409가 악성 Plan이라 예측할 수 있습니다.

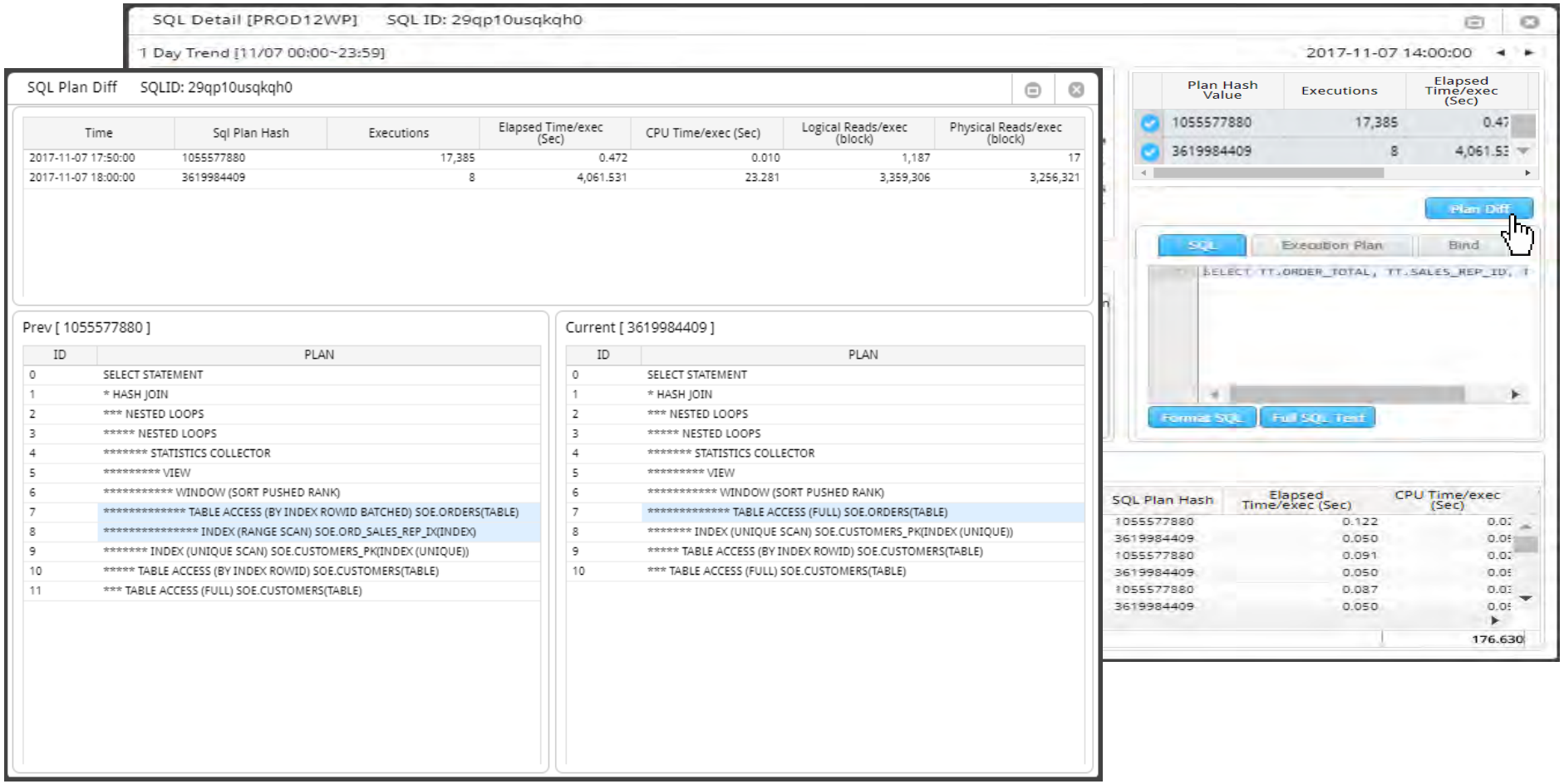
✓ 두 플랜의 차이와 어떤 부분이 문제를 일으켰는지 파악하기 위해서, 두 Plan Hash Value를 선택 후 Plan Diff 버튼을 눌러 플랜을 비교해 보겠습니다.

" 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 Detail 연계화면



✓ 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]

PRODUCT
DOCUMENTATION

Contents

Long-Term Analysis ?

Long-Term Analysis 은 언제 쓰나요?

MaxGauge

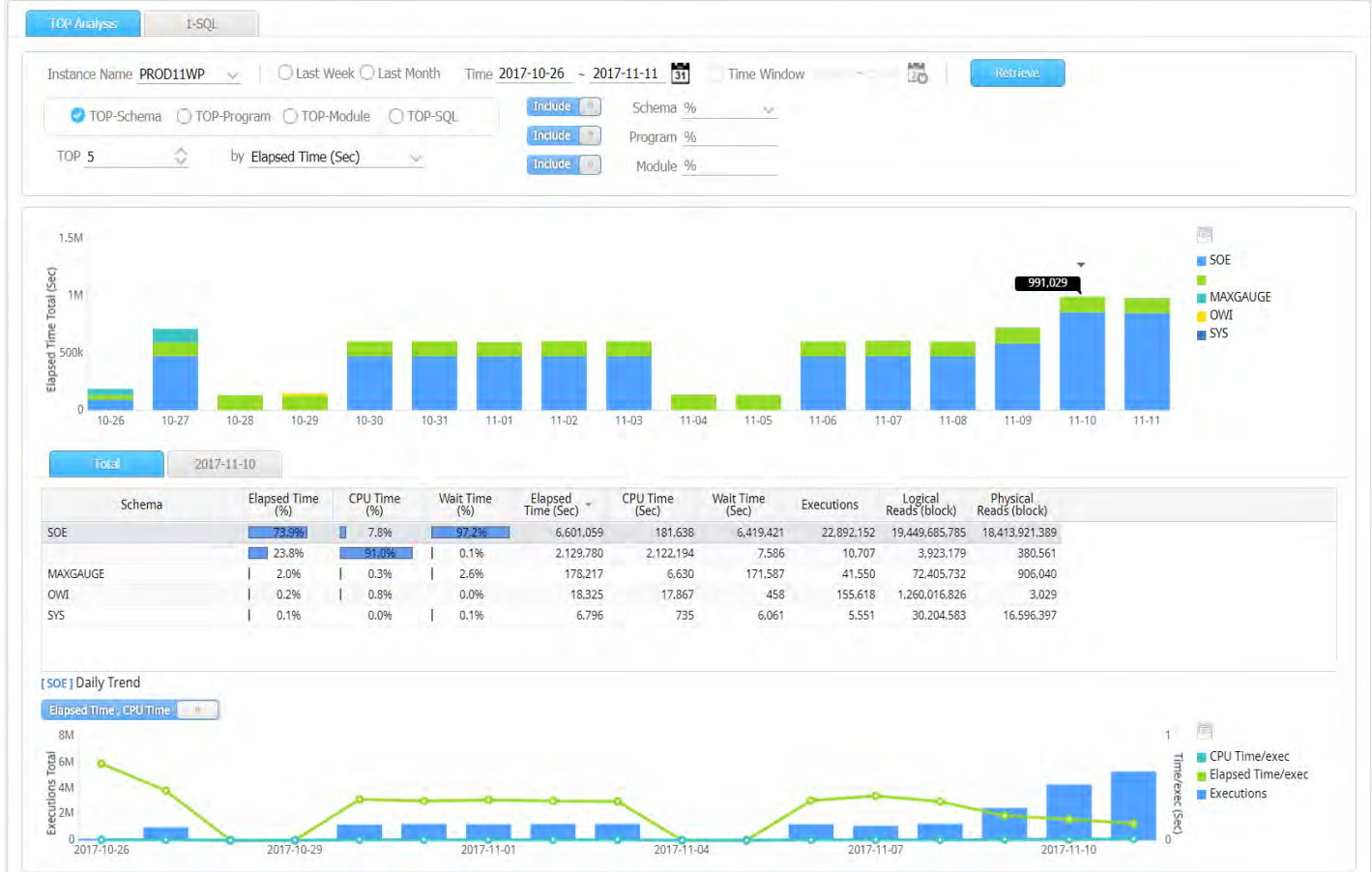
case 1.

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

case 2.

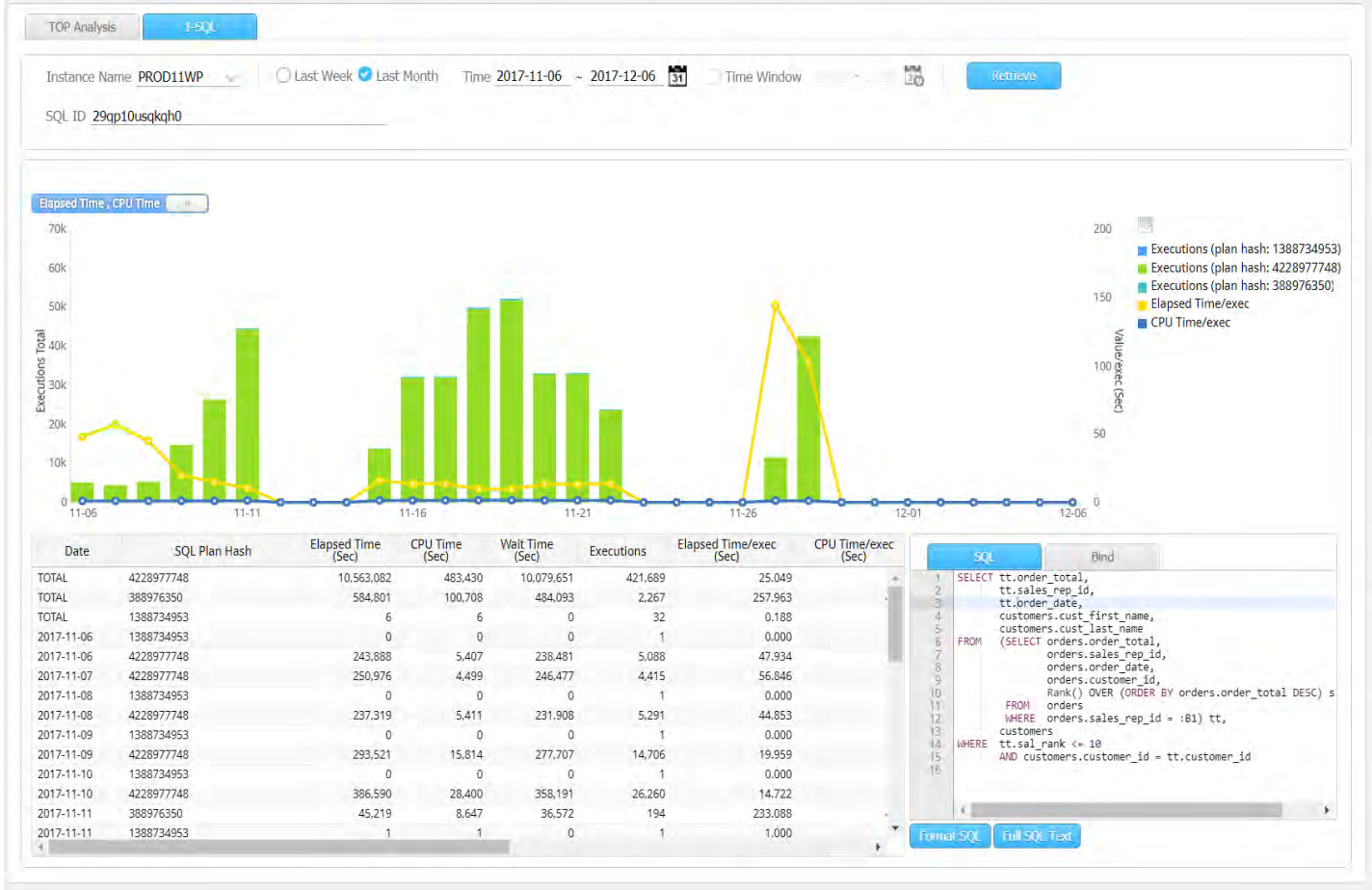
SQL 개선 후, 상태를 체크하고 싶어요!
특정 SQL 개선 후 성능을 확인하고 싶습니다.

Long-Term Analysis – TOP Analysis



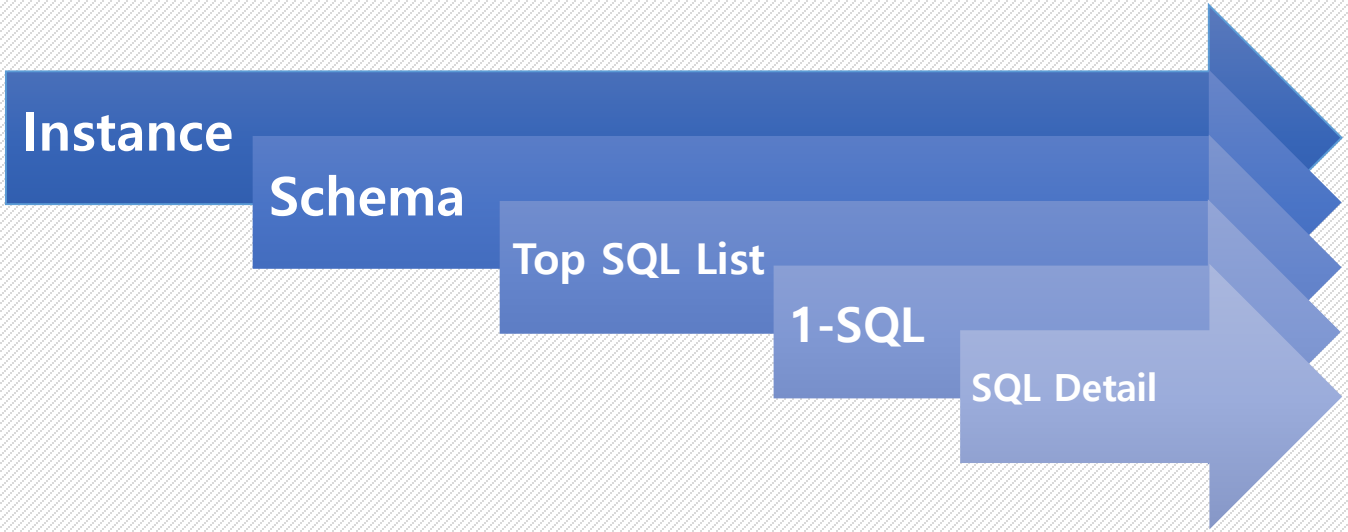
- Performance Analyzer > SQL Analysis 탭에 위치한다.
- Long-Term Analysis 2개의 기능 중 1개이다.
- TOP Analysis 기능은 TOP-N에 대하여 장기간 추이 분석 기능을 제공한다.
- TOP-N 기준은 Schema/Program/Module/SQL을 대상으로 진행한다.
- 기간은 Last Week / Last Month/ 날짜 선택으로 조회 가능하며, Time Window 기능으로 특정 시간대만 분석 가능하다.
- 조회한 TOP-N 기준으로 그래프와 리스트를 출력하며, 클릭 시 해당 정보에 대한 Daily Trend 정보가 하단에 출력된다.

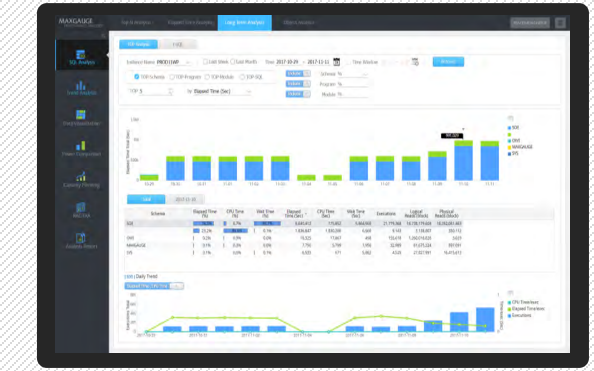
Long-Term Analysis – 1-SQL



- 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!



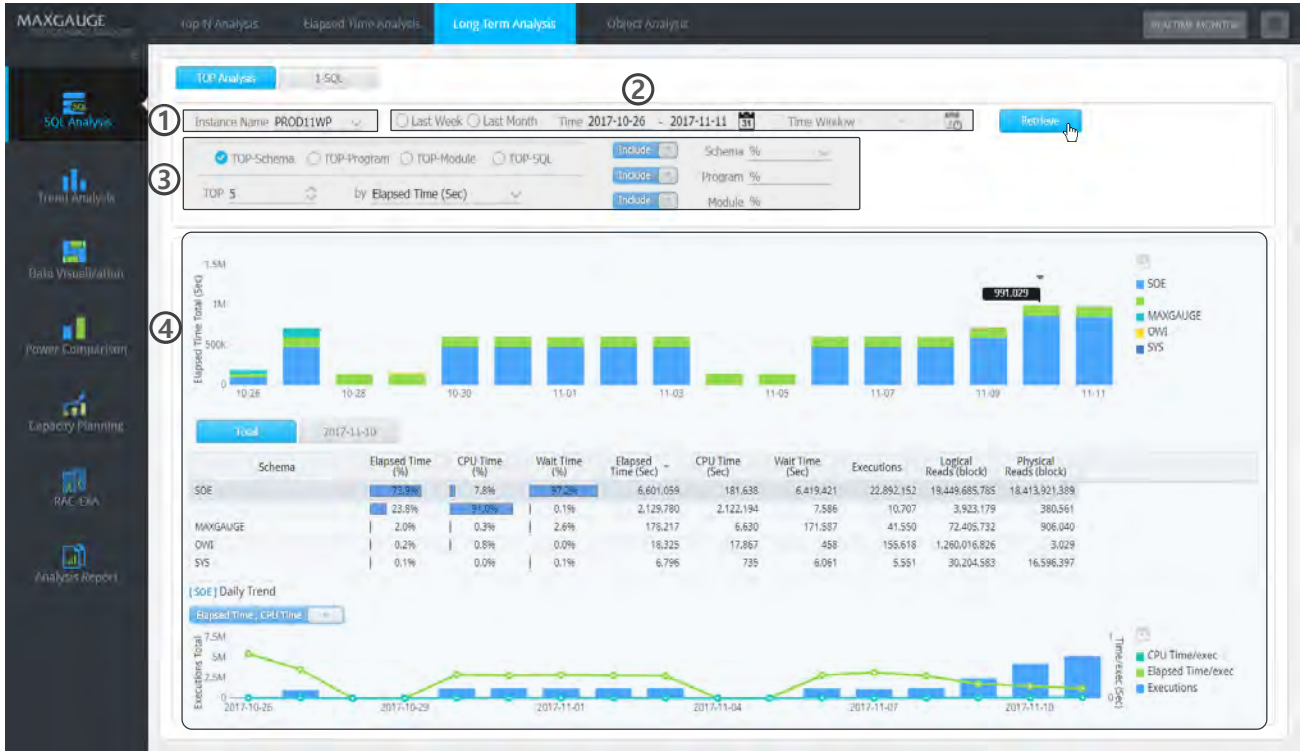


MaxGauge

Long-Term Analysis의 활용

Long-Term Analysis의 사용 방법

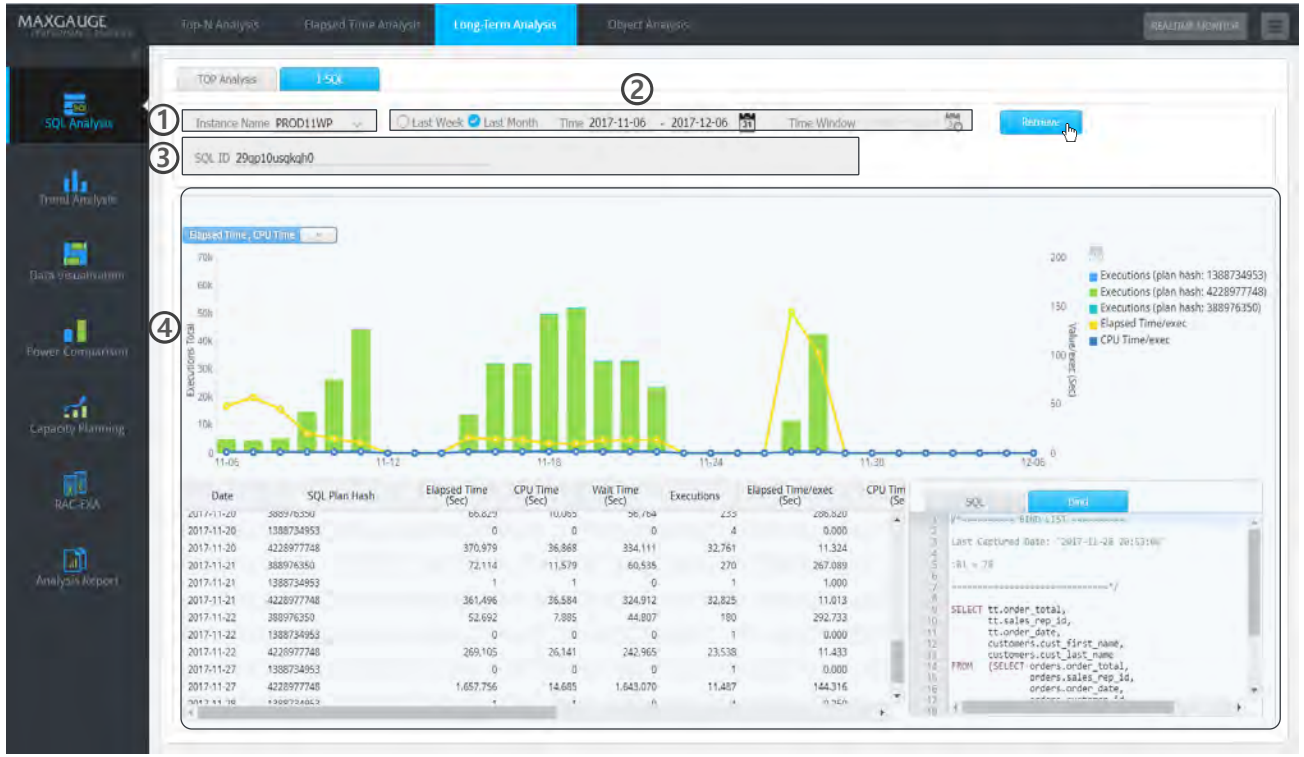
TOP Analysis 사용



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

1 SQL Analysis 사용



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

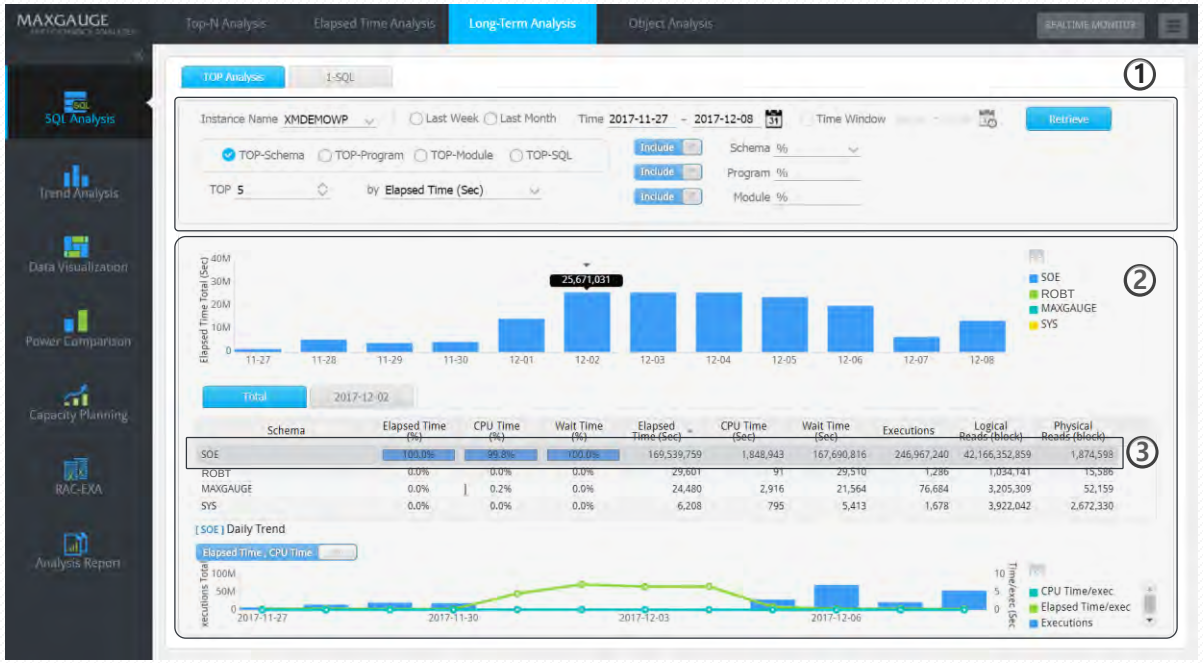
- Top Schema 조건으로 검색
  - Top Analysis > TOP-Schema 조회

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**

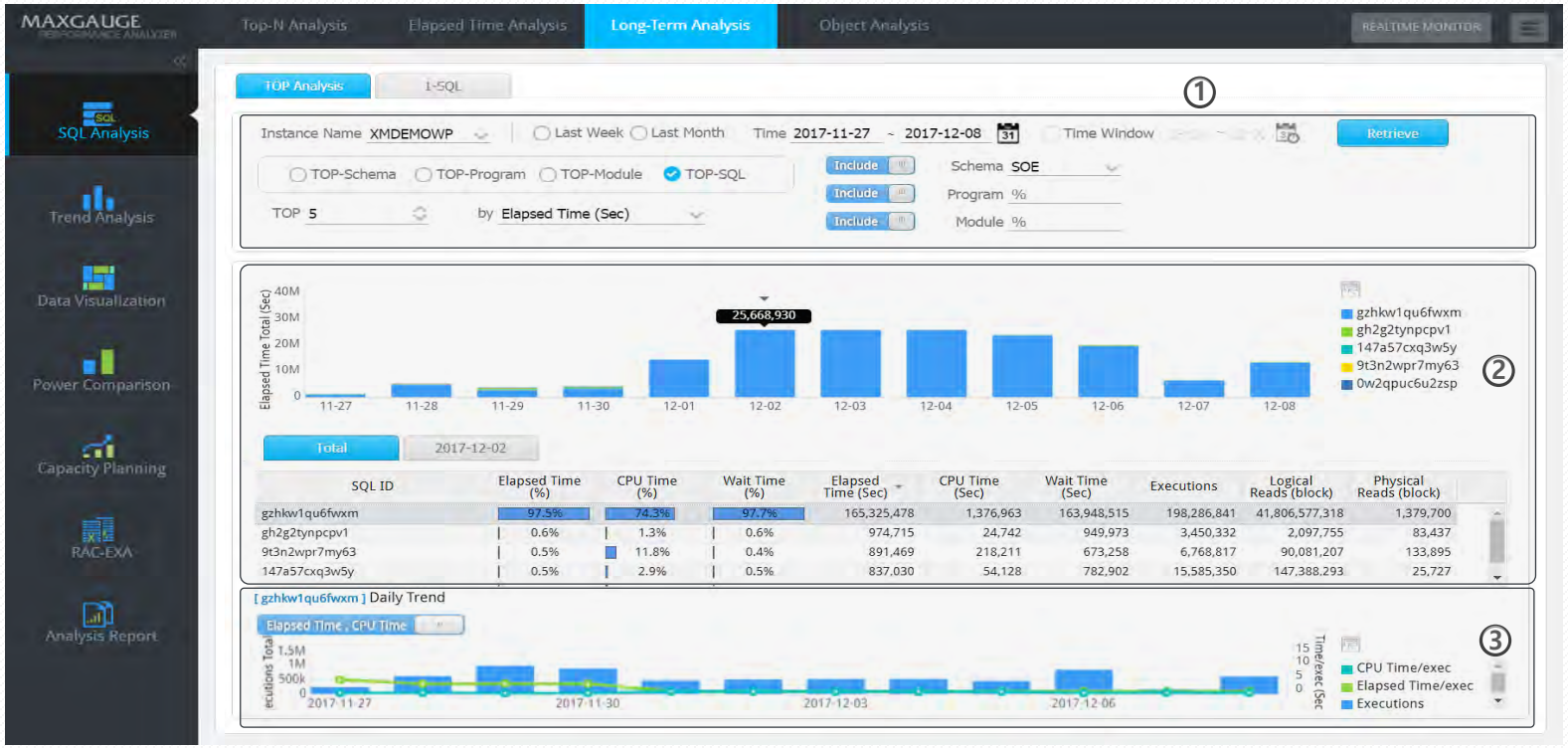
특정 Schema에서 동작하는 TOP SQL을 추출해 일별 시간대별 빈번하게 동작하는 SQL을 조회

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

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

"특정 Schema에서 매일 빈번하게 동작하는 SQL을 찾고 싶어요! "

» Top Analysis > Top - SQL 조회



① SQL Analysis ▶ Long-Term Analysis ▶ TOP Analysis 에서 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.

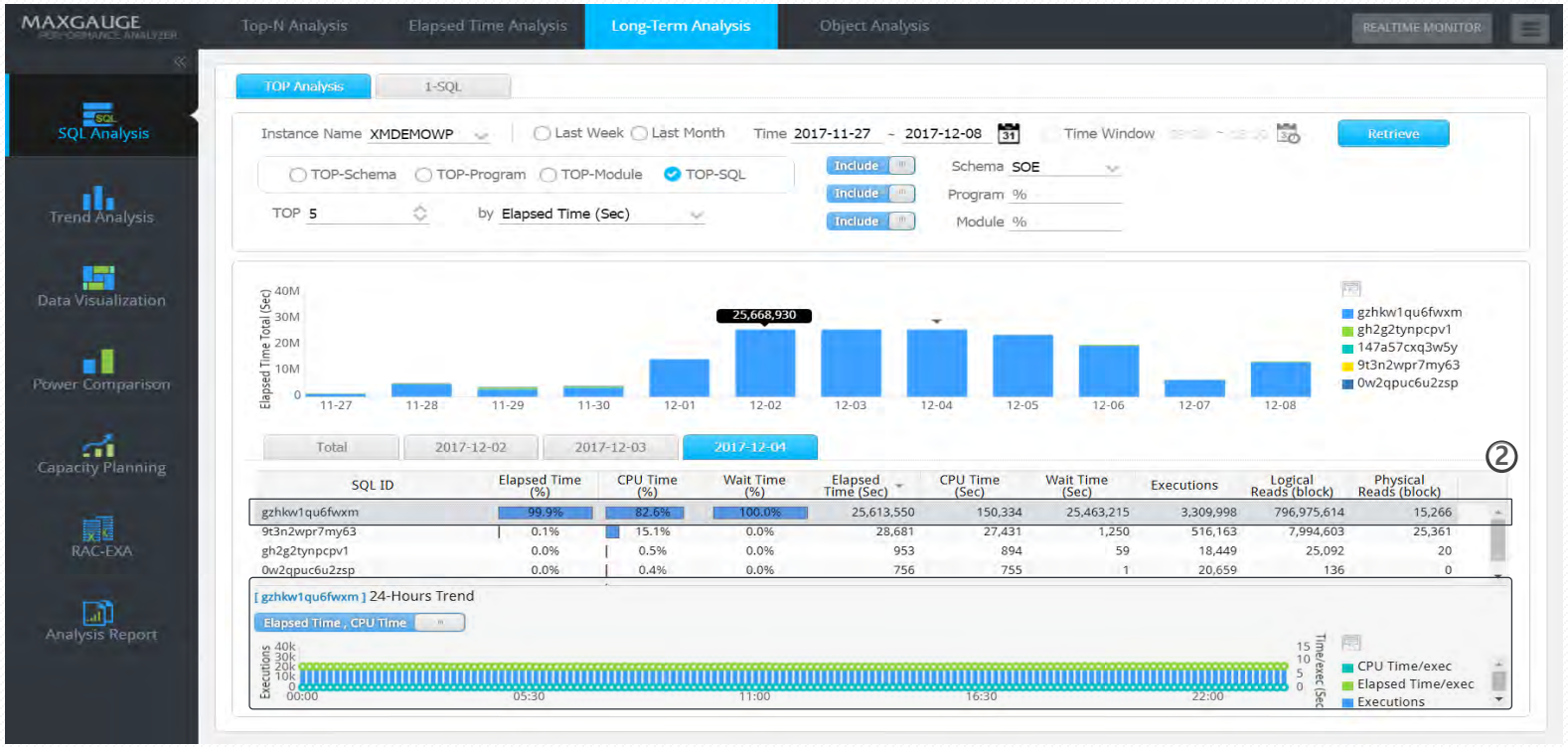
③ 특정 SQL을 클릭할 경우, Daily Trend탭으로 특정 SQL의 일별 동작 이력을 확인할 수 있습니다.

"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 SQL조회
   - 특정 SQL의 상세 정보 확인
     └ SQL Detail 연계

» 부하 SQL 검색



✓ 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.

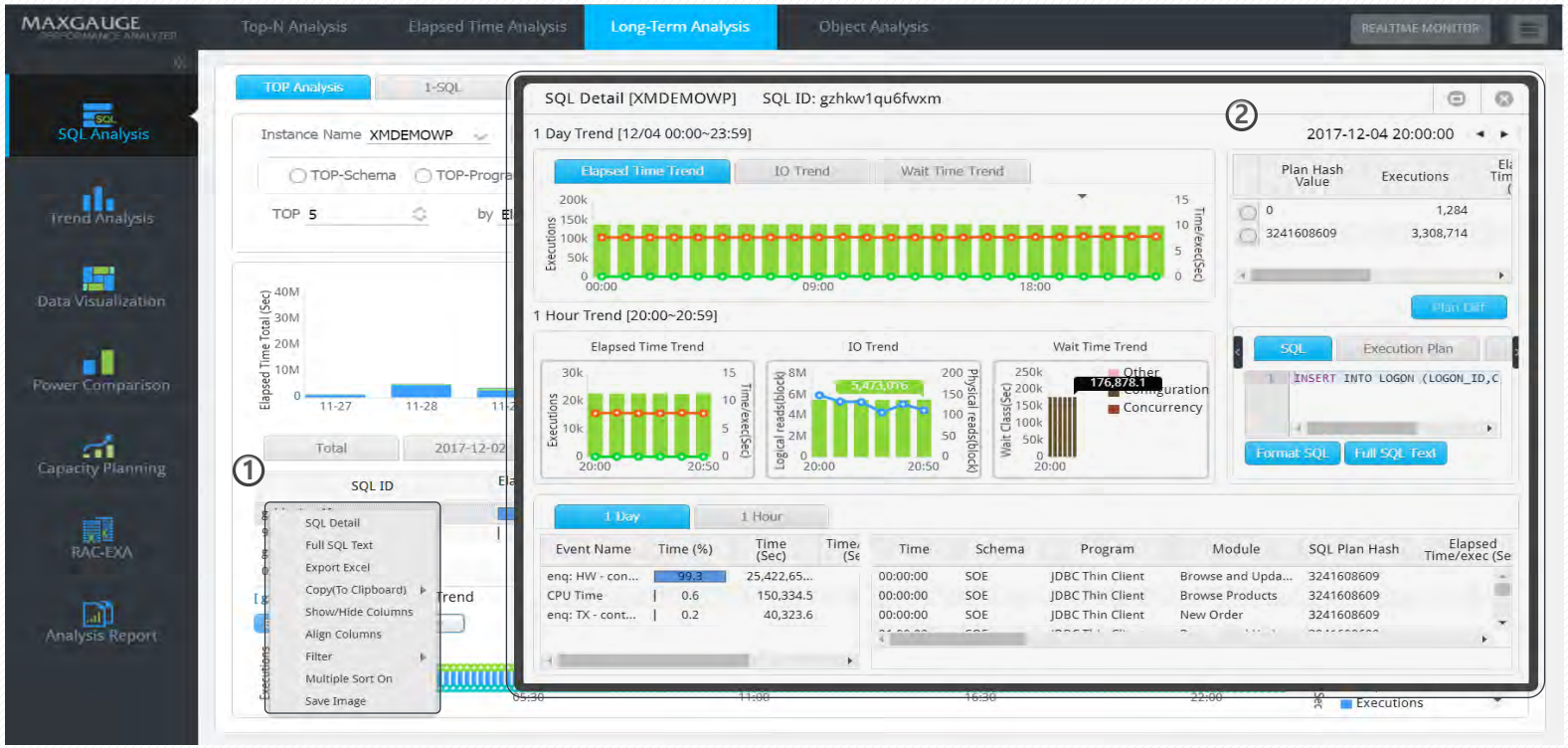
※ 해당 탭으로 일시적으로 발생한 TOP SQL과 주기적으로 발생하는 TOP SQL을 구분할 수 있습니다.

"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 SQL조회
   - 특정 SQL의 상세 정보 확인
     └ SQL Detail 연계

» 특정 SQL 분석



① 부하 쿼리로 판단되는 SQL의 상세 수행 정보를 SQL Details을 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!

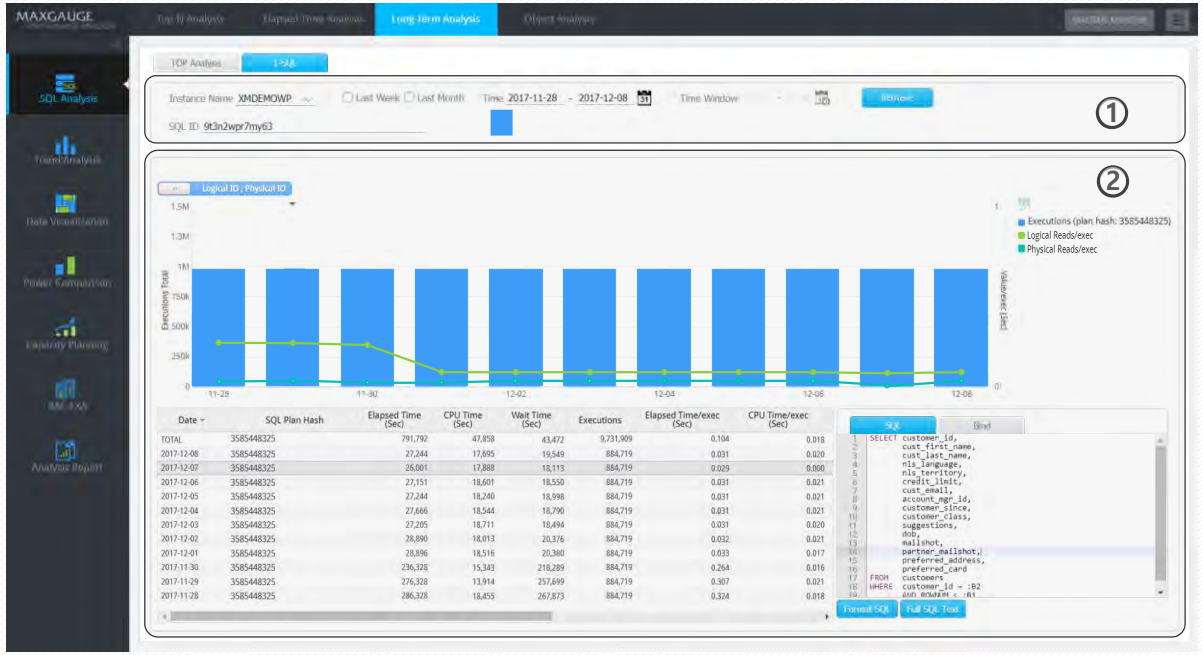
"SQL 개선 후, 상태를 확인하고 싶어요!"

STEP

1. 특정 SQL 검색

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

1-SQL조회



✓ **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.

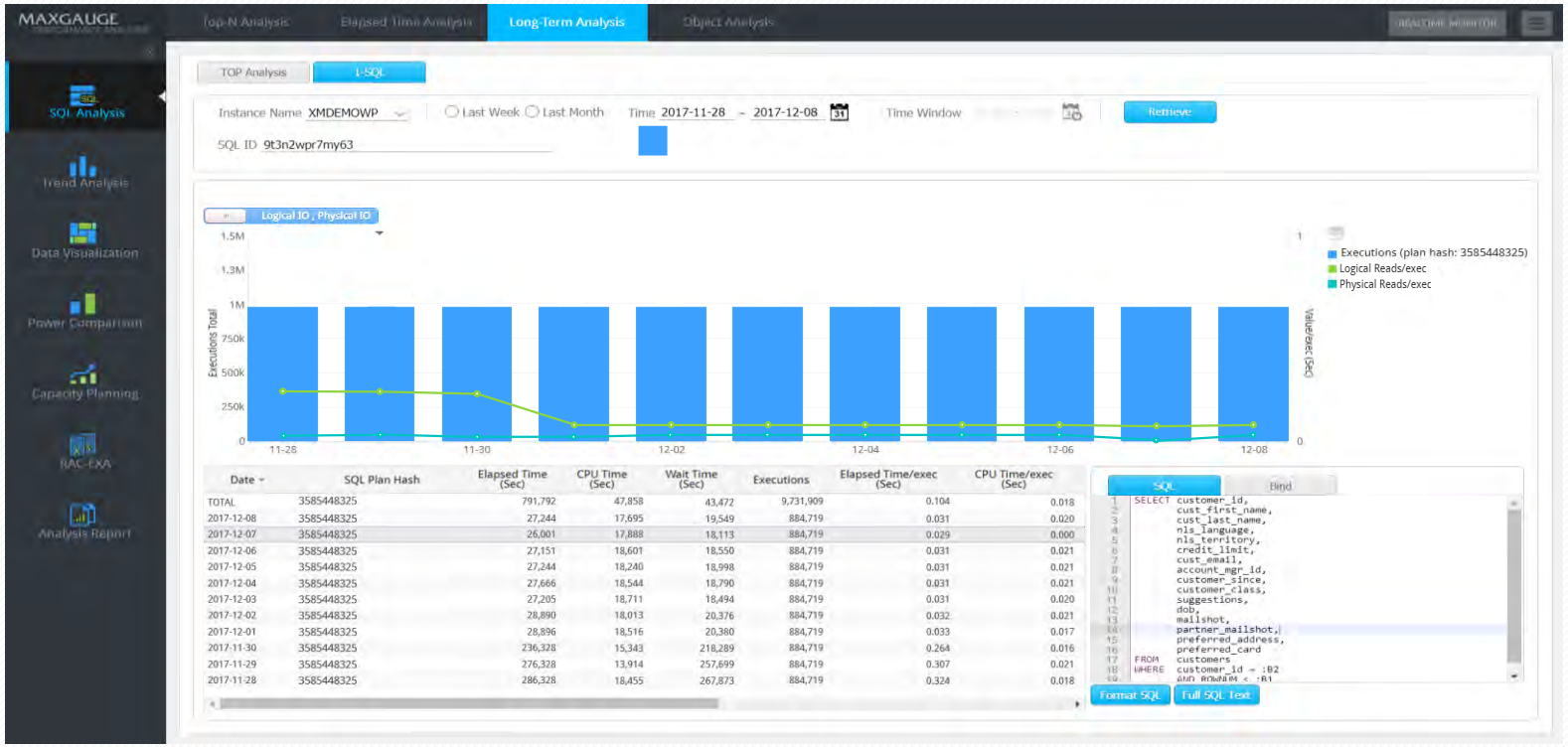
2 SQL ID(9t3n2wpr7my63)에 대한 일자 별 추이, SQL TEXT, BIND 정보가 출력됩니다.

"SQL 개선 후, 상태를 확인하고 싶어요!"

STEP

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

» SQL 개선 여부 확인



✓ 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 XMDEMOWP의 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

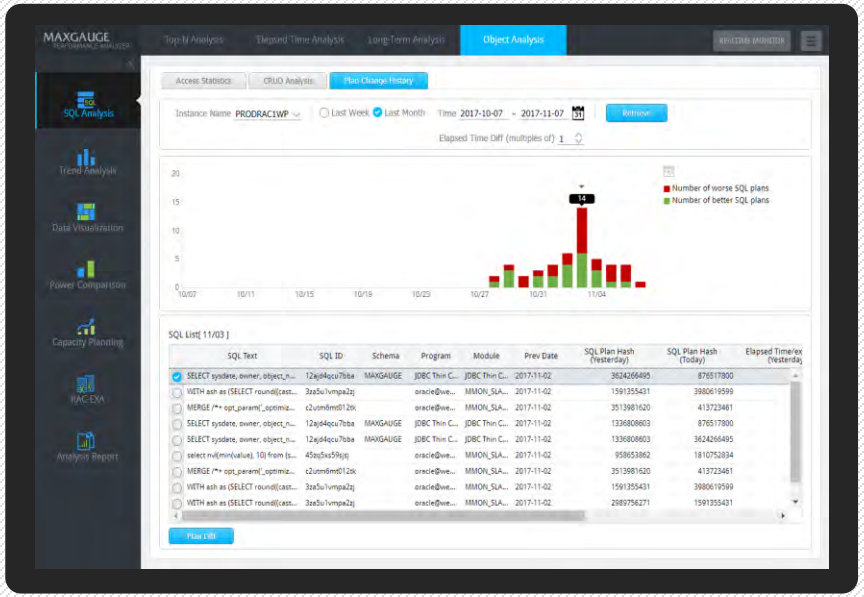
Plan Diff [Performance Analyzer]

PRODUCT
DOCUMENTATION

Contents

Plan Diff?

Plan Diff는 언제 쓰나요?



MaxGauge

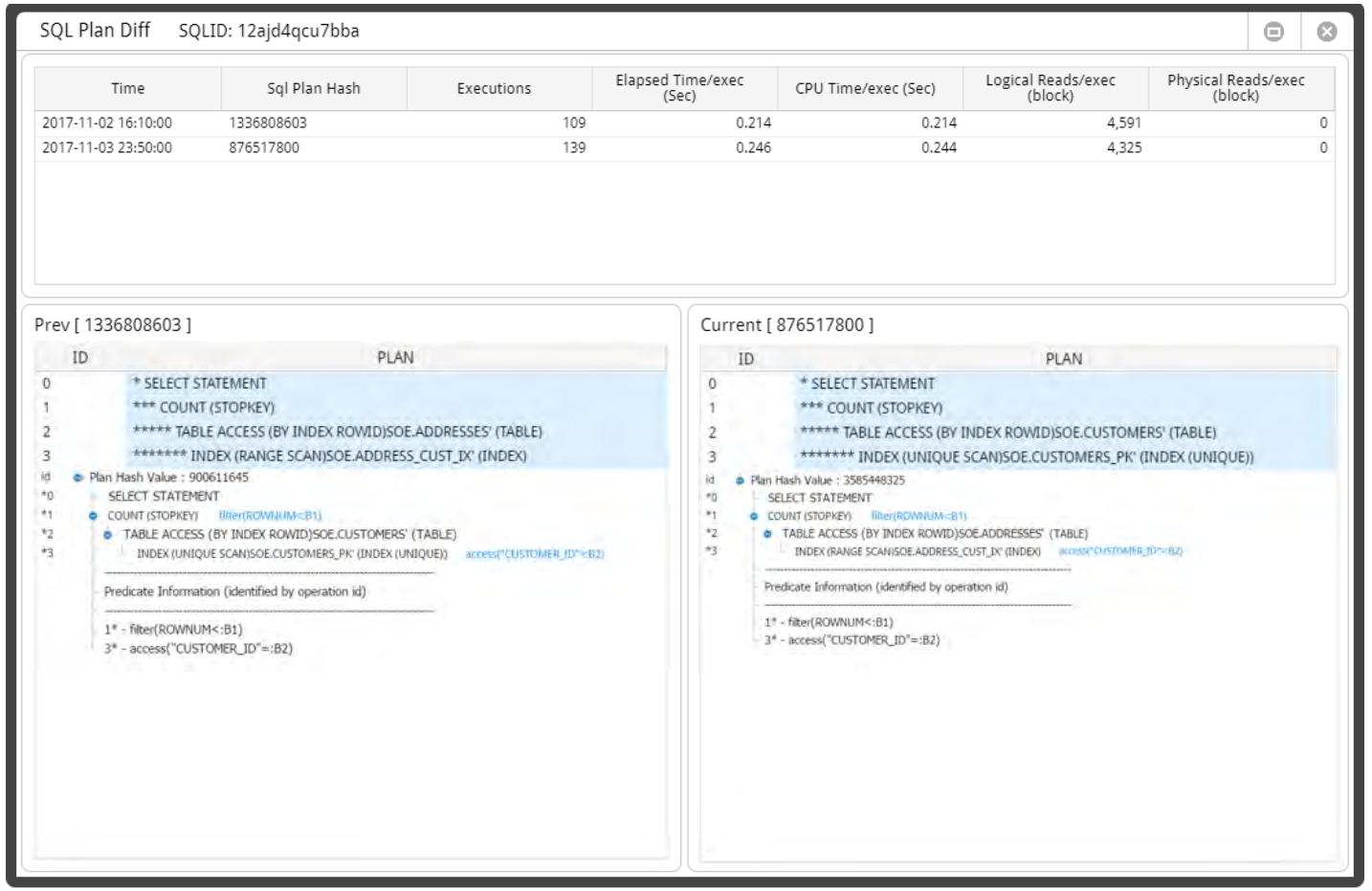
case 1.

SQL**이 갑자기 느려 졌습니다.**
Plan**의 변경 여부를 확인하고 싶습니다.**

case 2.

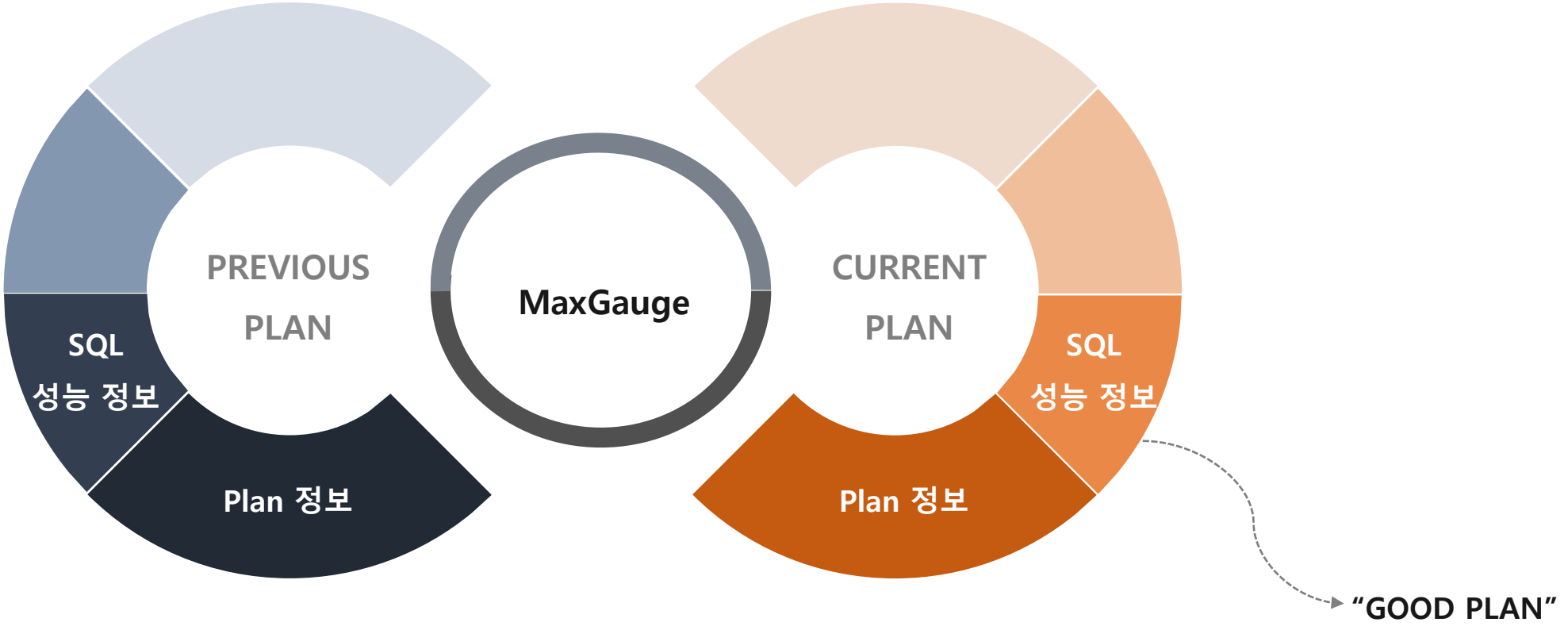
통계정보 갱신 작업을 하였습니다.
DB **전반의** SQL Plan**을 확인하고 싶습니다.**

Plan Diff



- **다른 Plan 을 갖는 동일한 SQL**의 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.

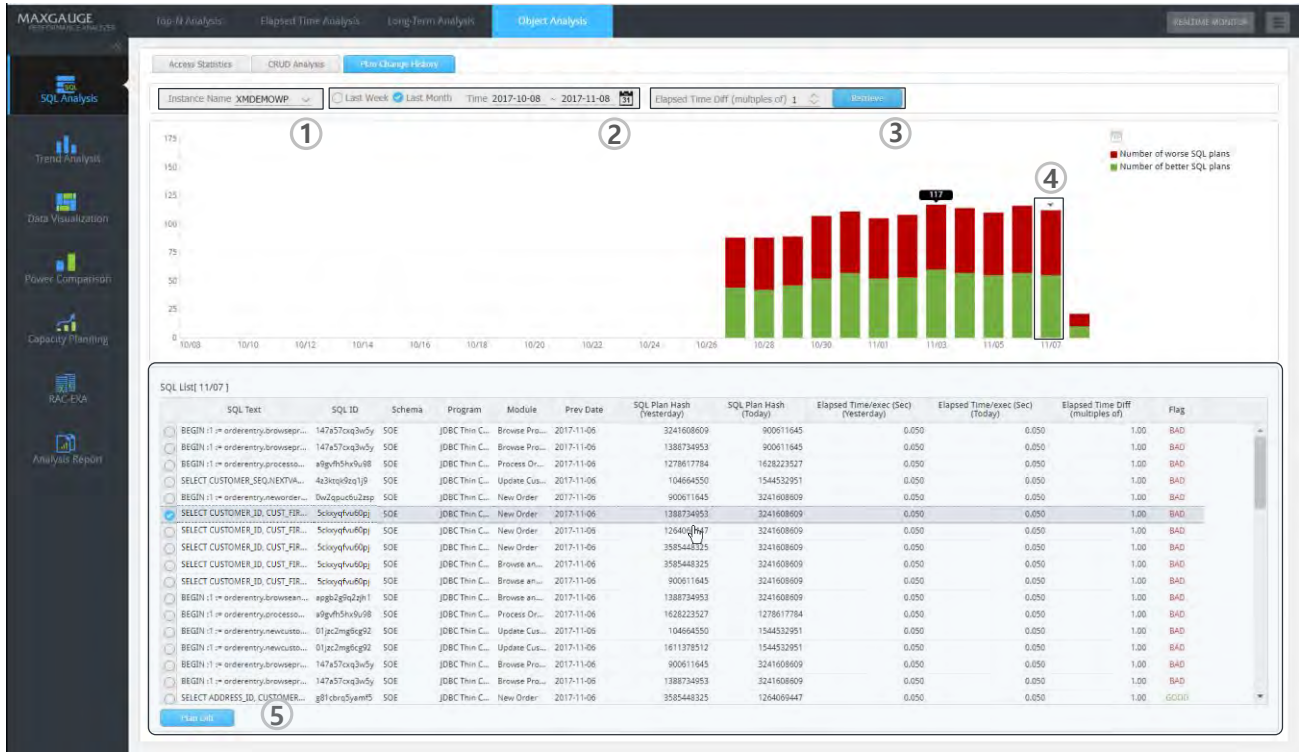
Plan 비교와 Plan 변경 원인 파악이 쉽습니다.



Plan Diff의 활용

Plan Diff의 사용 방법

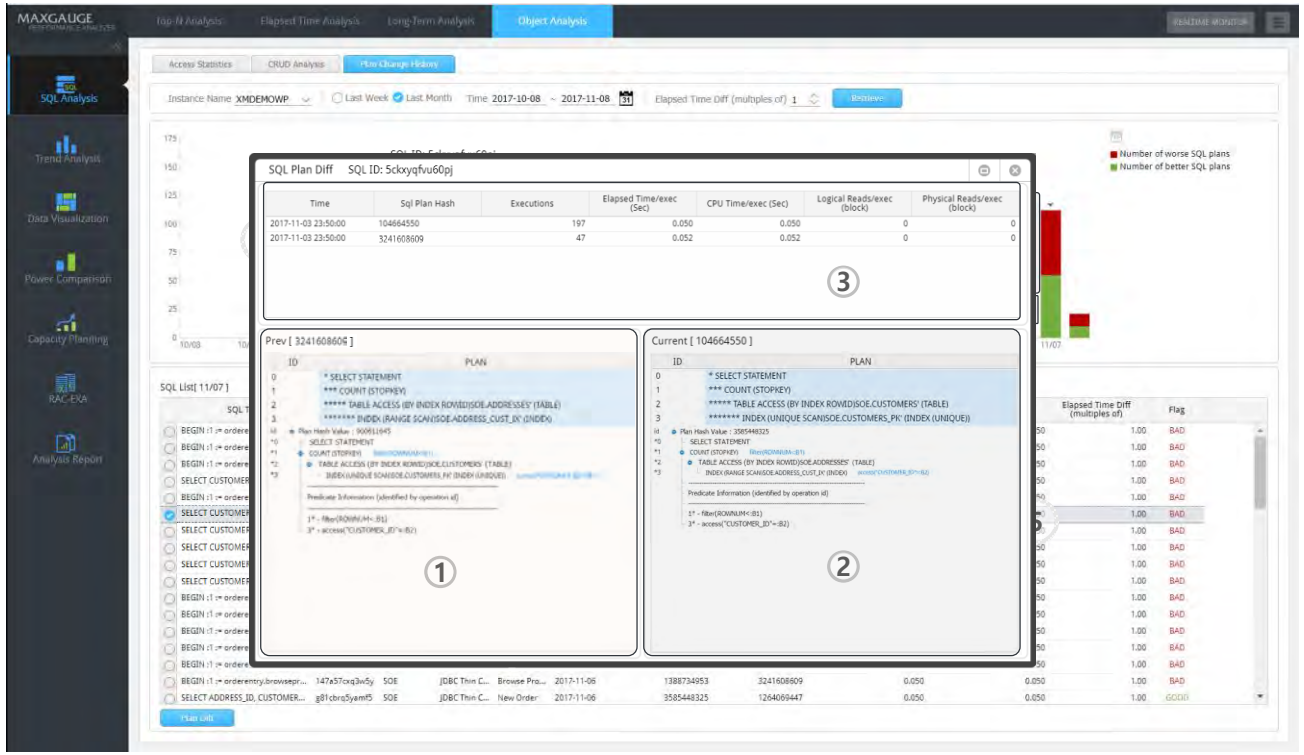
Plan diff 사용



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 사용 방법

Plan Diff 조회



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

실전 분석 사례 Case 1.

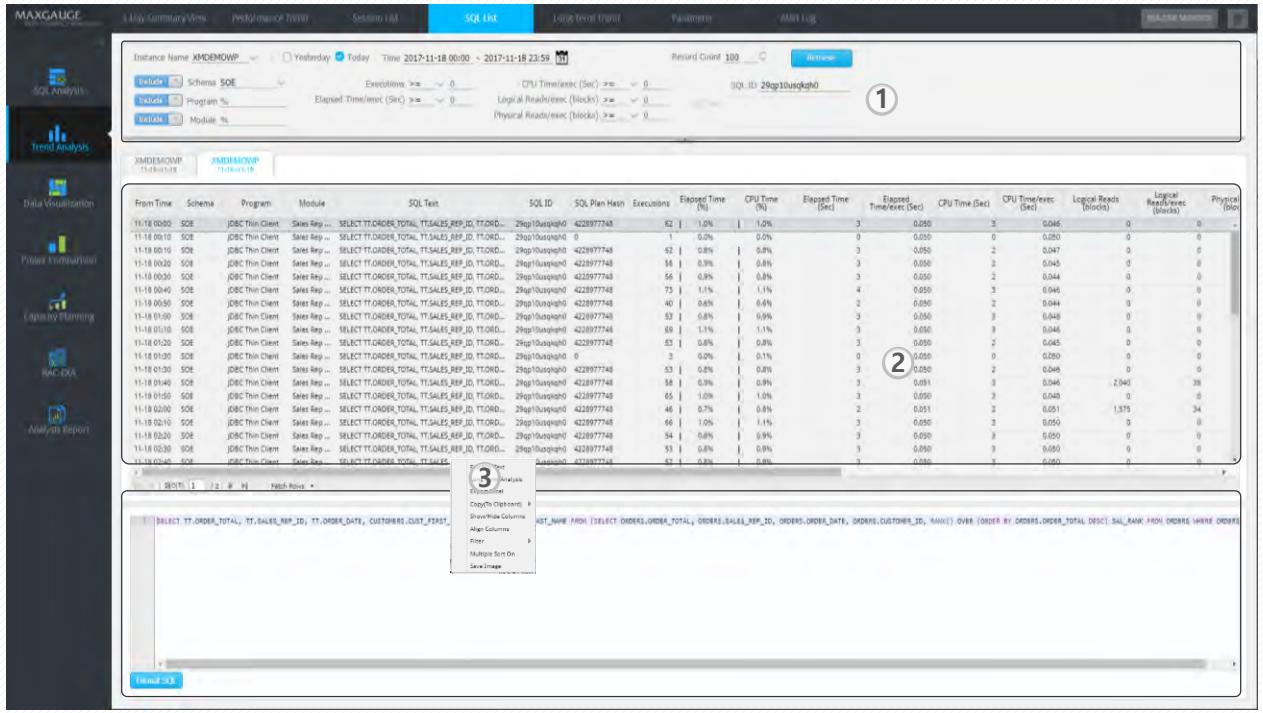
SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요.

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

"SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요! "

SQL LIST >> SQL 조회



✓ **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.

1 SQL Analysis ▶ TOP-N Analysis에서 인스턴스(XMDEMOWP) 명, Schema(SOE), 날짜(2017.11.18), SQL ID(29qp10usqkqh0)로 조회합니다.

2 해당 SQL 에 대한 10분당 Summary 데이터를 통해, SQL이 느려진 시간의 SQL수행 정보를 확인합니다.

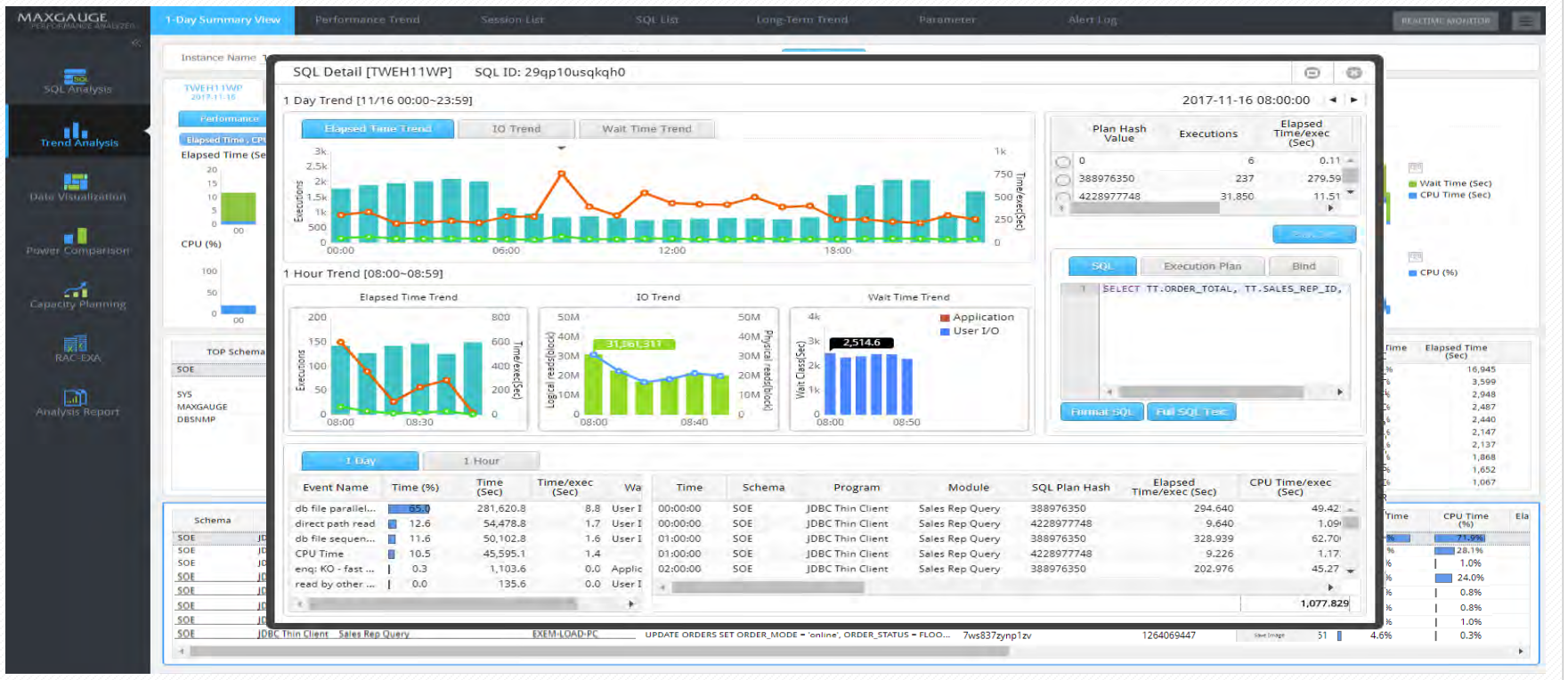
3 느려진 SQL 의 상세 수행정보를 확인하기 위해 오른쪽 마우스 클릭 후, SQL Detail을 선택합니다.

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

"SQL이 느려졌어요, Plan의 변경 여부를 확인하고 싶어요! "

Plan Diff



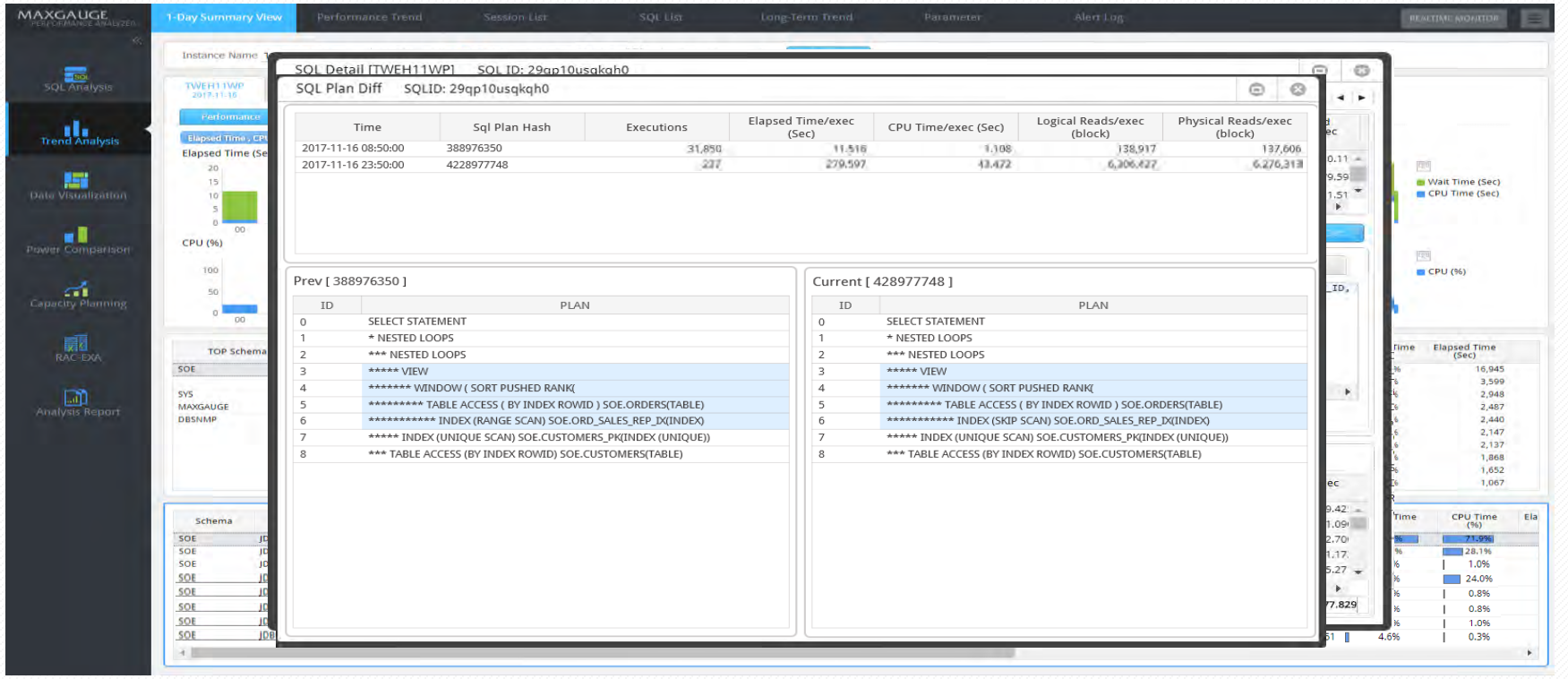
- ✓ SQL Details에서, SQL 상세 수행 정보를 확인합니다.
- ✓ 상단 우측 Plan Hash Value 뷰 에서 Hash value가 여러 개임을 확인합니다.
- ✓ Plan Hash value가 두 개 이상일 경우, Plan 변경으로 인해 SQL 수행이 오래 걸릴 수 있습니다.
- ✓ Plan Hash value 별 Elapsed Time, Executions 수치를 포함한 SQL 수행정보를 확인 후, 비교하고자 하는 Plan Hash value를 체크하고 plan diff를 클릭합니다.

"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



✓ SQL Plan Diff에서 해당 SQL의 상세 수행 정보를 비교합니다.

✓ 하단 화면에서 SQL이 다를 경우 하이라이트 되어 보여집니다.

✓ Current SQL과 Prev SQL의 Plan을 비교합니다.

✓ Index Scan이 Index Skip Scan으로 바뀌어 수행되어, SQL 수행 속도에 영향을 주었음을 알 수 있습니다.

실전 분석 사례 Case 2.

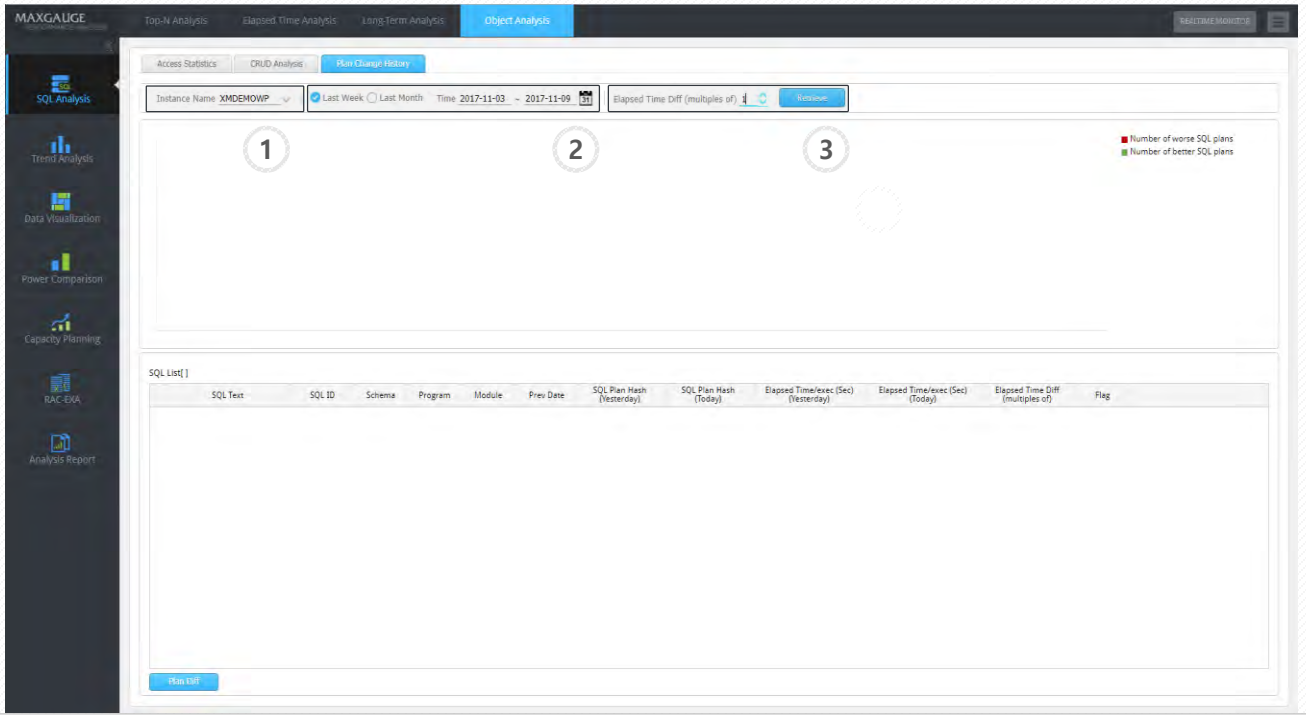
통계정보를 갱신했습니다.
전반적인 Plan 정보를 알고 싶어요!

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

" 통계정보를 갱신했습니다. 전반적인 Plan 정보를 알고 싶어요! "

Plan Diff 화면



✓ 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

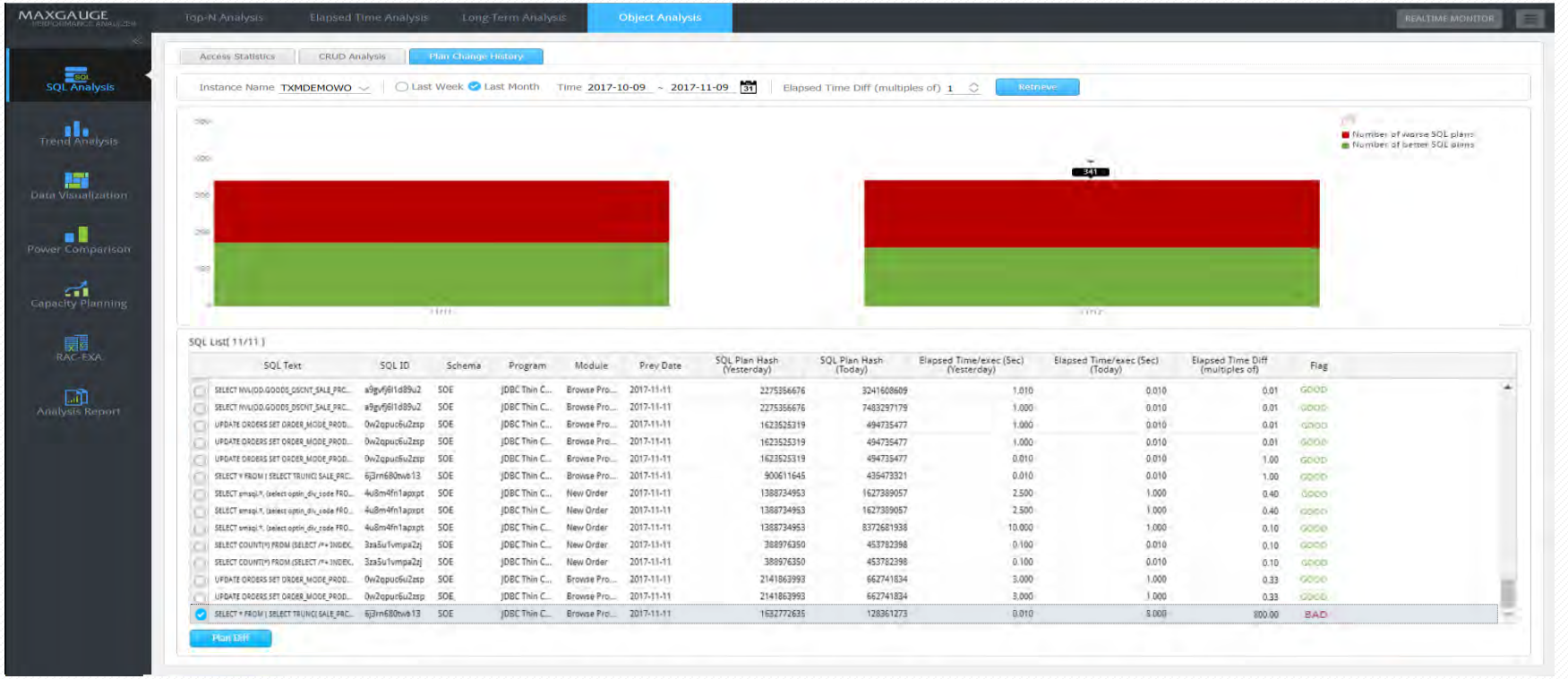
1. Plan 정보를 확인할 Instance명(**TXMDEMOWP**)를 선택합니다.
2. 통계정보 갱신 이후 날짜를 포함하여 기간을 선택합니다.
3. Plan 별 수행시간(Elapsed Time) 차이를 옵션으로 사용할 수 있습니다. 단위는 배수이며, 1배수도 선택 가능합니다.

" 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



✓ 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.

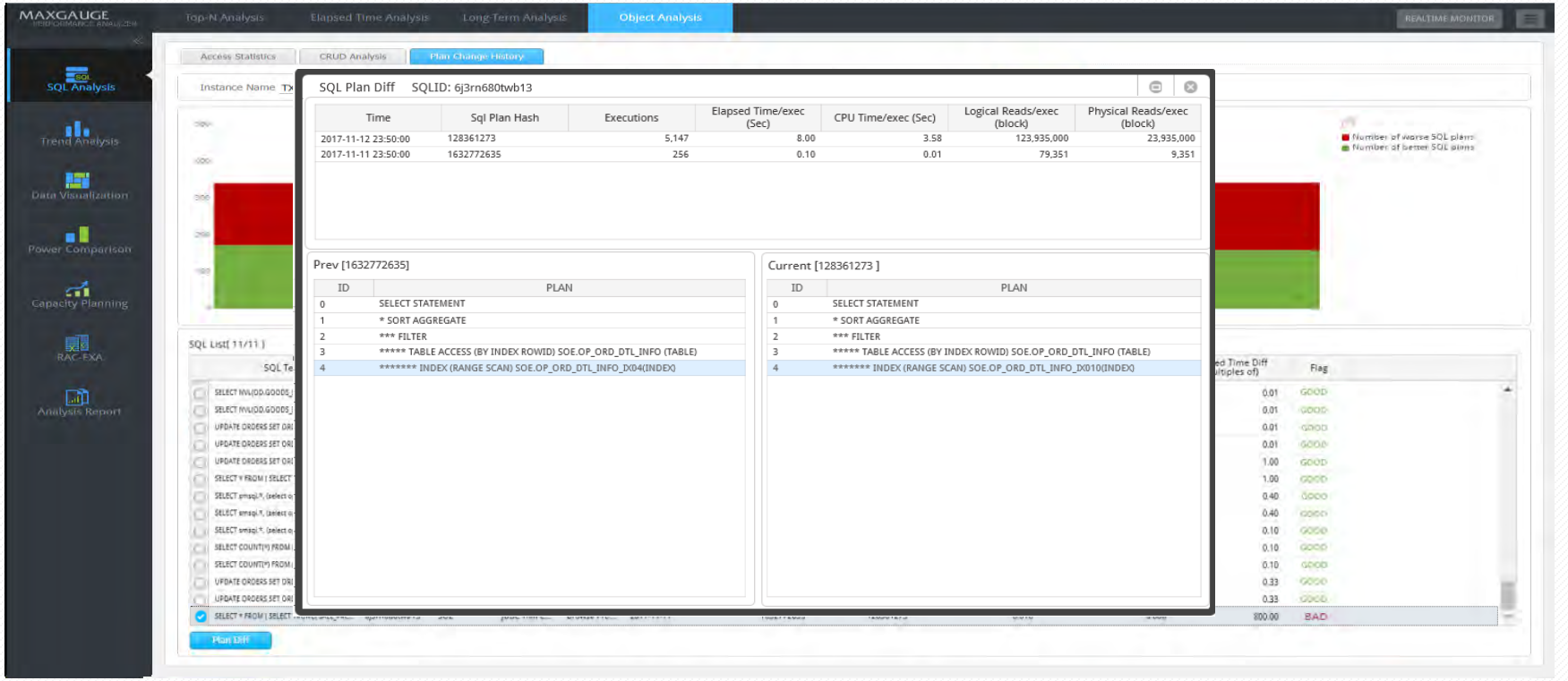
✓ Flag 칼럼이 "BAD"인 SQL은 Plan 이전보다 성능이 저하 된 SQL로 수행성능 및 분석이 필요합니다.

" 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.

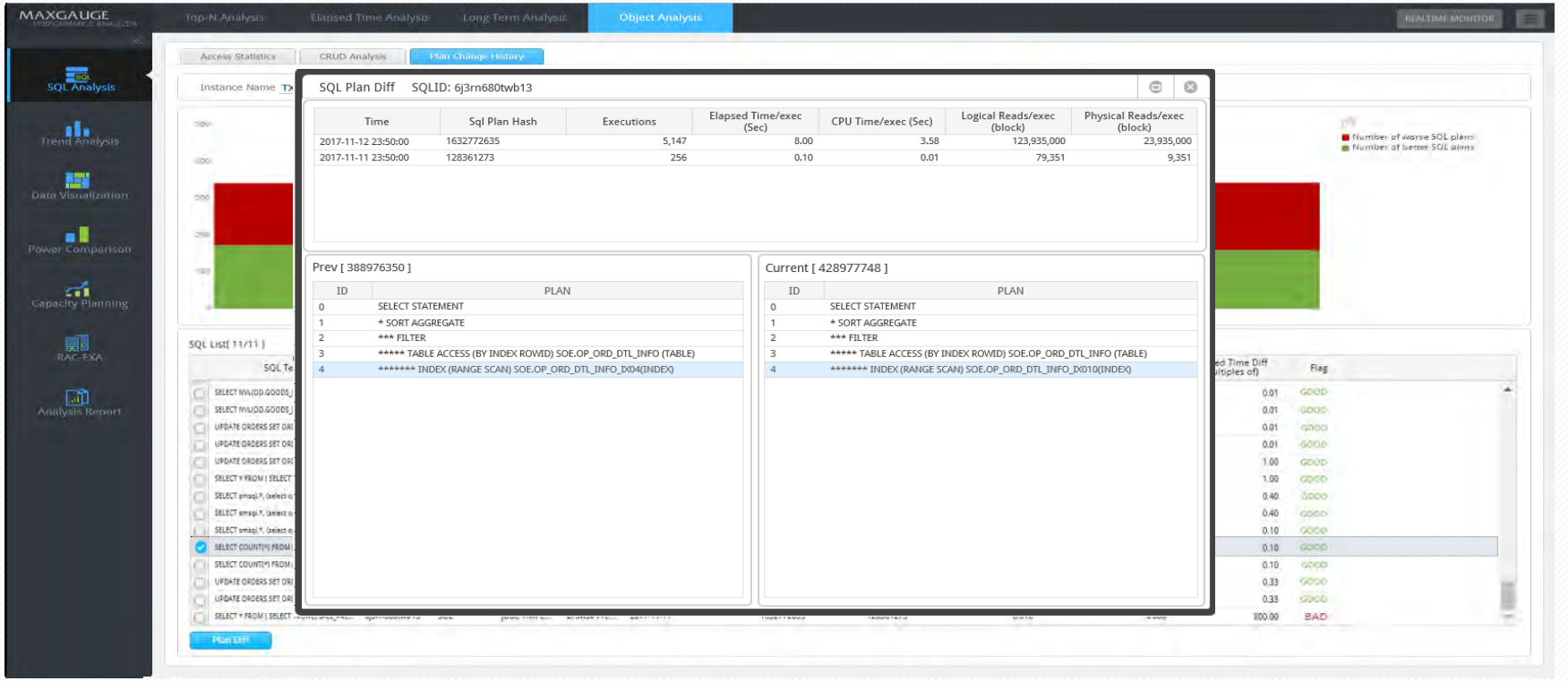
✓ 인덱스 정보 및 SQL 수행 이력을 고려하여, 해당 SQL 개선 여부를 고려해야 합니다.

" 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**

» Plan Diff 조회 화면



- ✓ 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_IX04)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]

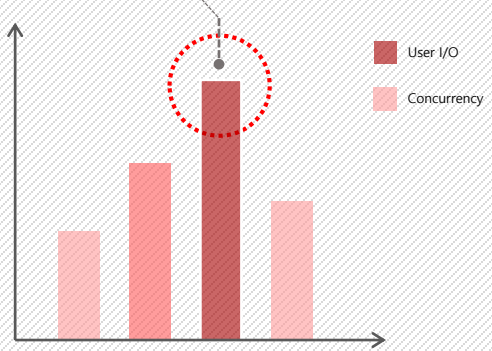
Contents

Contents

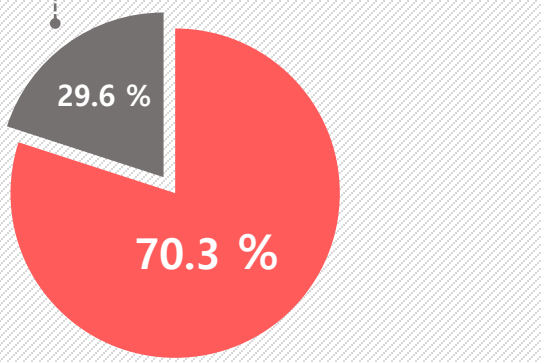
Wait Time Analysis?

Wait Time Analysis은 언제 쓰나요?

Top SQL의 **Elapsed Time(=wait)을 상세 분석**하고,

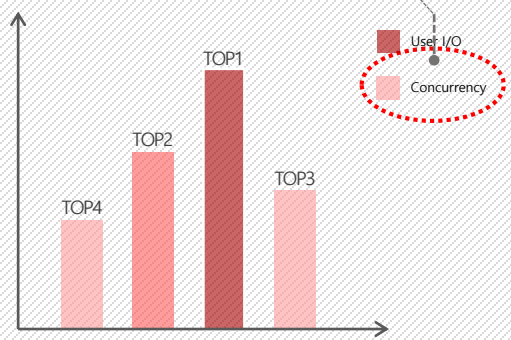


Safe Zone을 사용하여 **SQL 튜닝 대상을 선정**할 때 사용합니다.

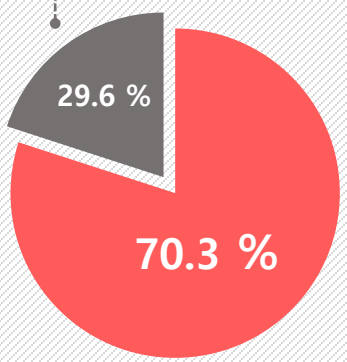


Wait Time Analysis은 언제 쓰나요?

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



Safe Zone을 사용하여 **SQL 튜닝 대상을 선정**할 때 사용합니다.



Wait Time Analysis

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

Wait Time 분석 방법 비교

Maxgauge 분석

01 *Real-Time Monitoring* **02** *Wait Time Analysis* **03** *SQL 선별 및 Wait Event 확인*

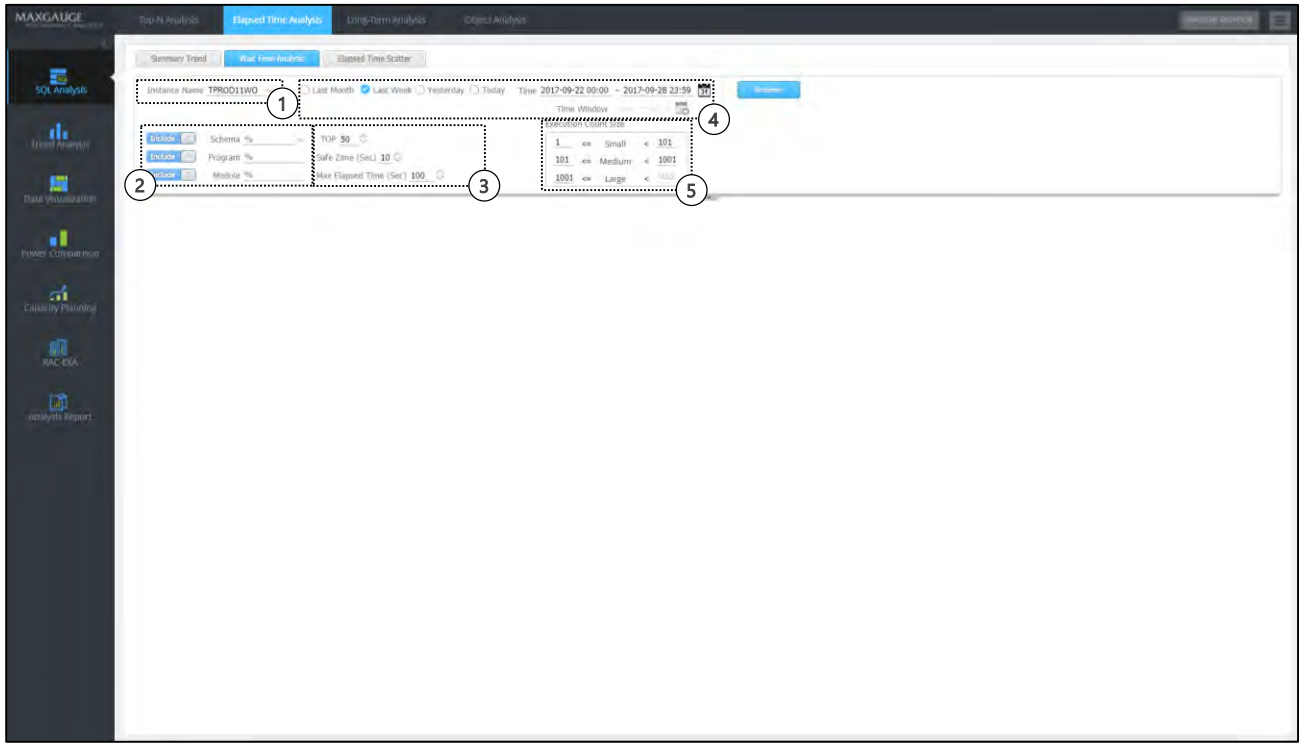
ORACLE AWR 분석

01 *AWR 자료수집* **02** *AWR Report 추출* **03** *SQL Wait Event 확인* **04** *SQL 성능이력 추출* **05** *SQL 실행계획 및 Bind 수집*

Wait Time Analysis의 활용

Wait Time Analysis의 사용 방법

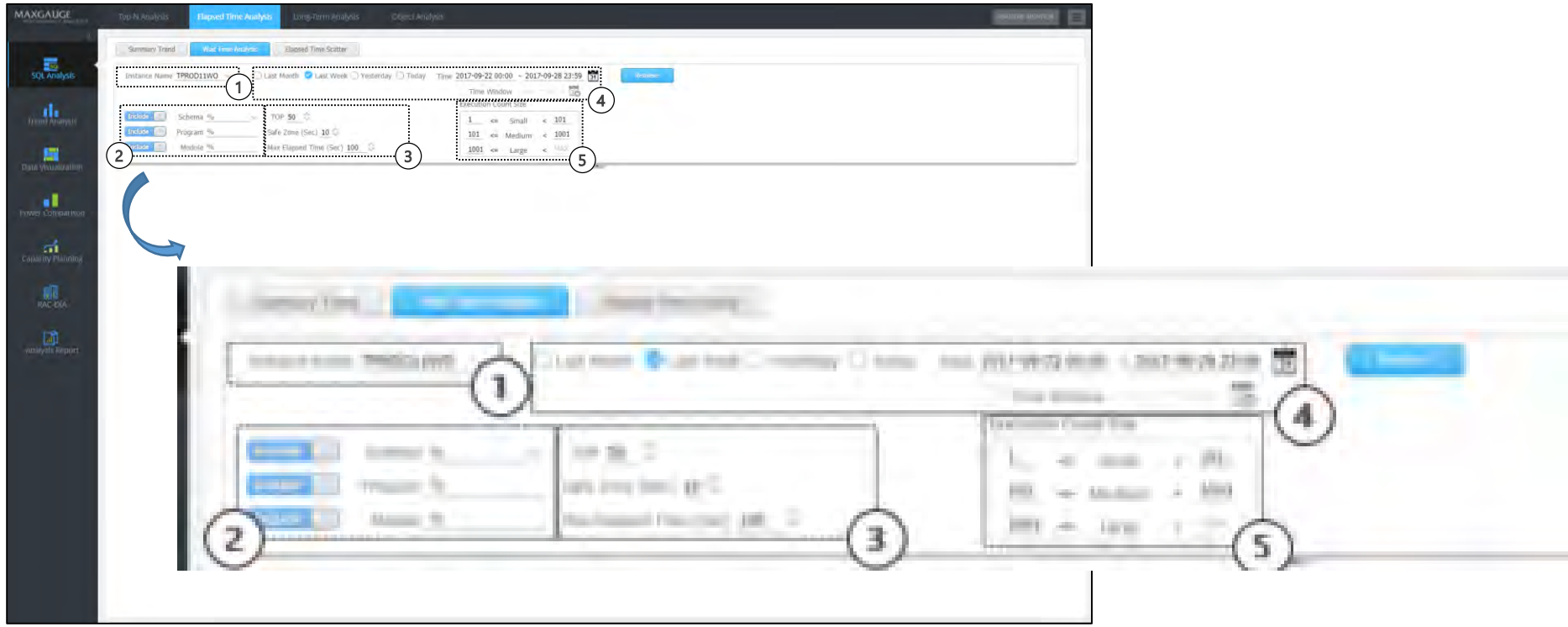
Wait Time Analysis 검색 화면



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 화면



① 대상 Instance를 선택합니다.

② Schema, Program, Module명을 선택적으로 추가할 수 있습니다.

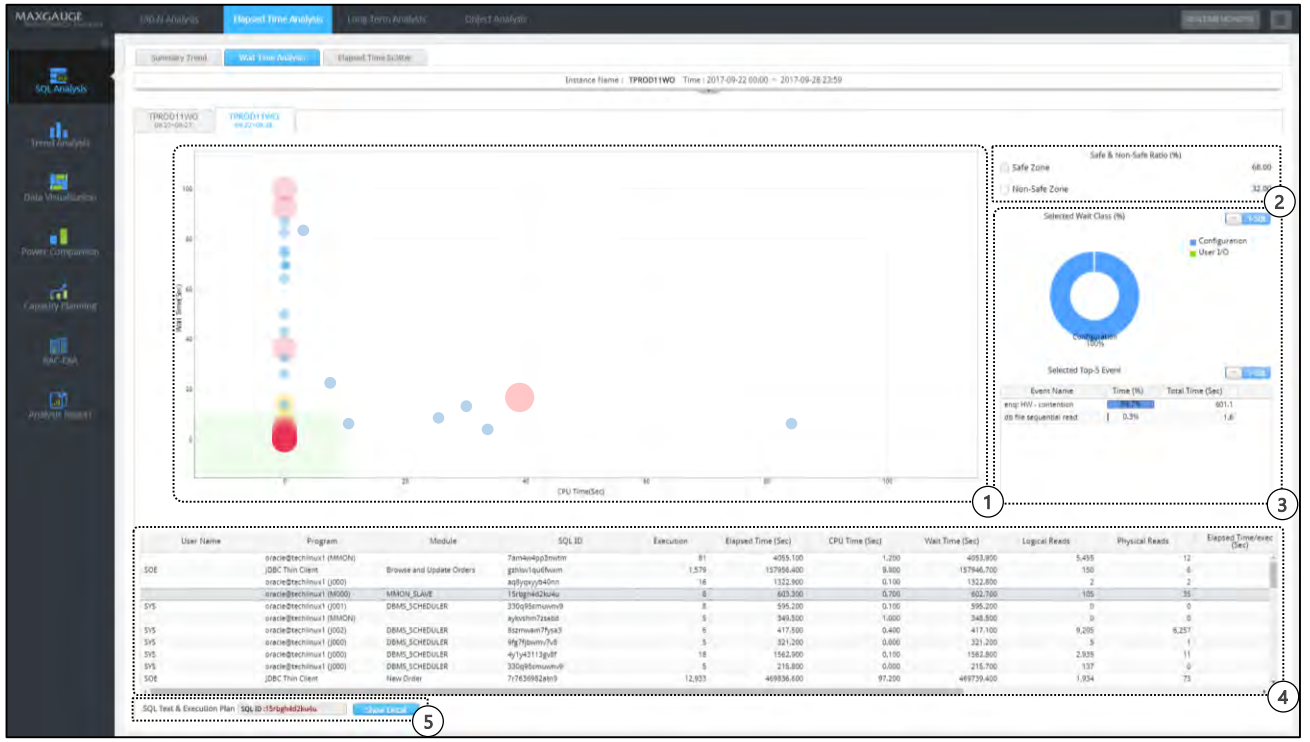
③ 출력되는 SQL의 개수 선택 및 Safe Zone 영역 설정과 SQL의 Max 수행시간을 설정합니다.
※ Safe Zone : 안정적인 수행 구간이라 생각되는 영역을 사용자가 임의로 설정

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

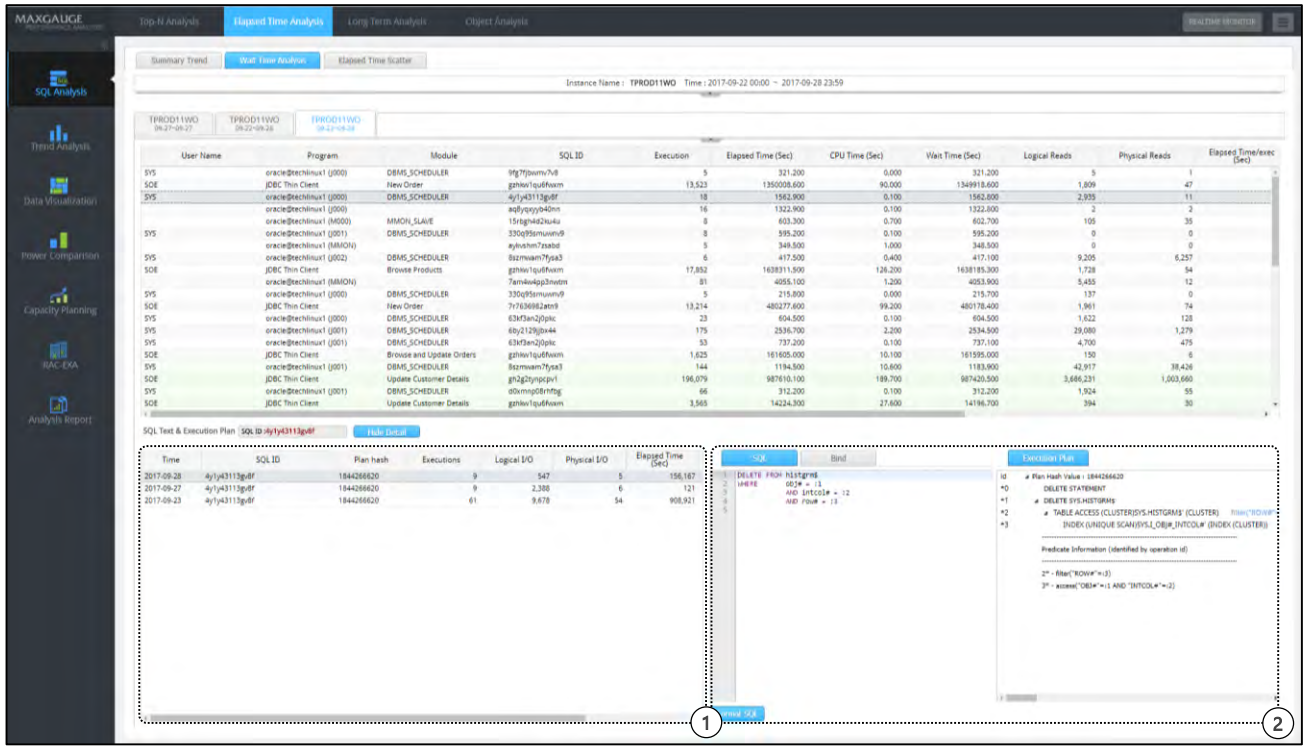
Wait Time Analysis 조회 화면



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 조회 화면

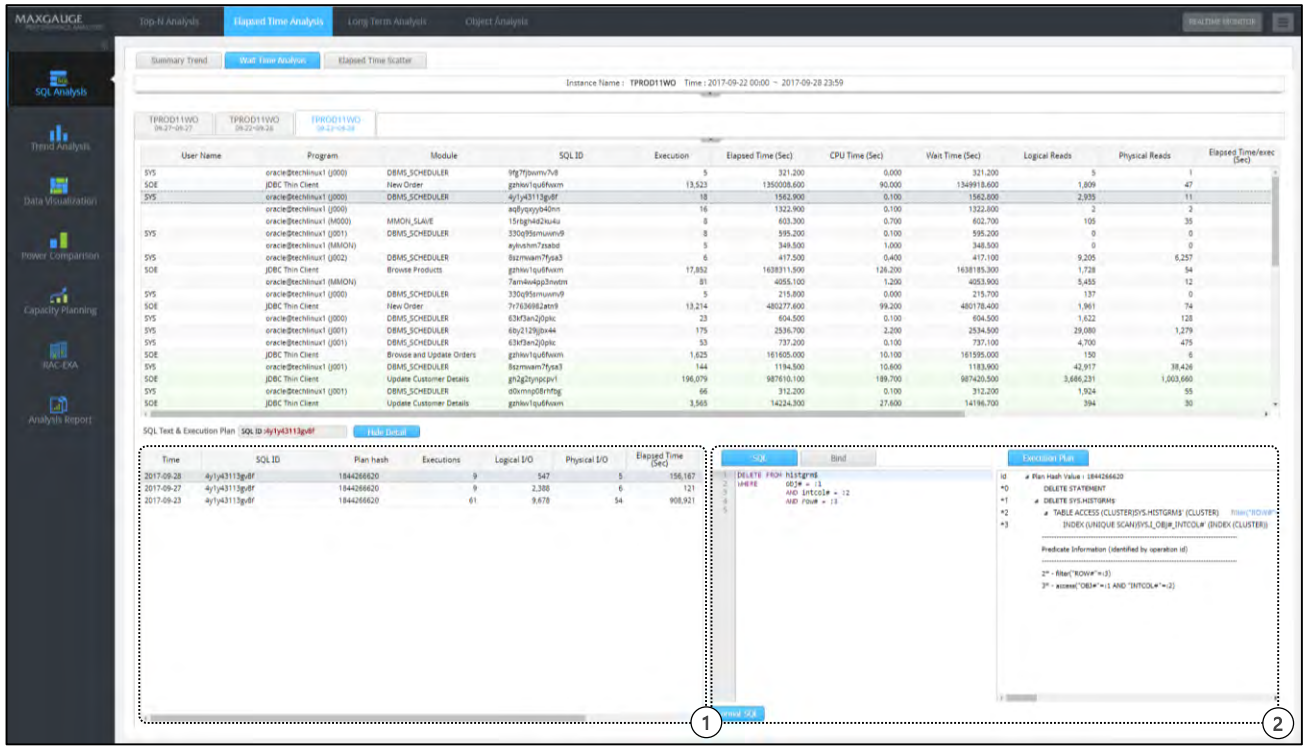


1 선택된 SQL의 일별 SQL 성능 Sum 정보를 확인할 수 있습니다.

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

Wait Time Analysis의 사용 방법

Wait Time Analysis 조회 화면



1 선택된 SQL의 일별 성능 정보를 확인할 수 있습니다.

2 선택된 SQL에 대한 Text 및 Bind, Execution Plan 정보를 확인할 수 있습니다.

실전 분석 사례 Case 1.

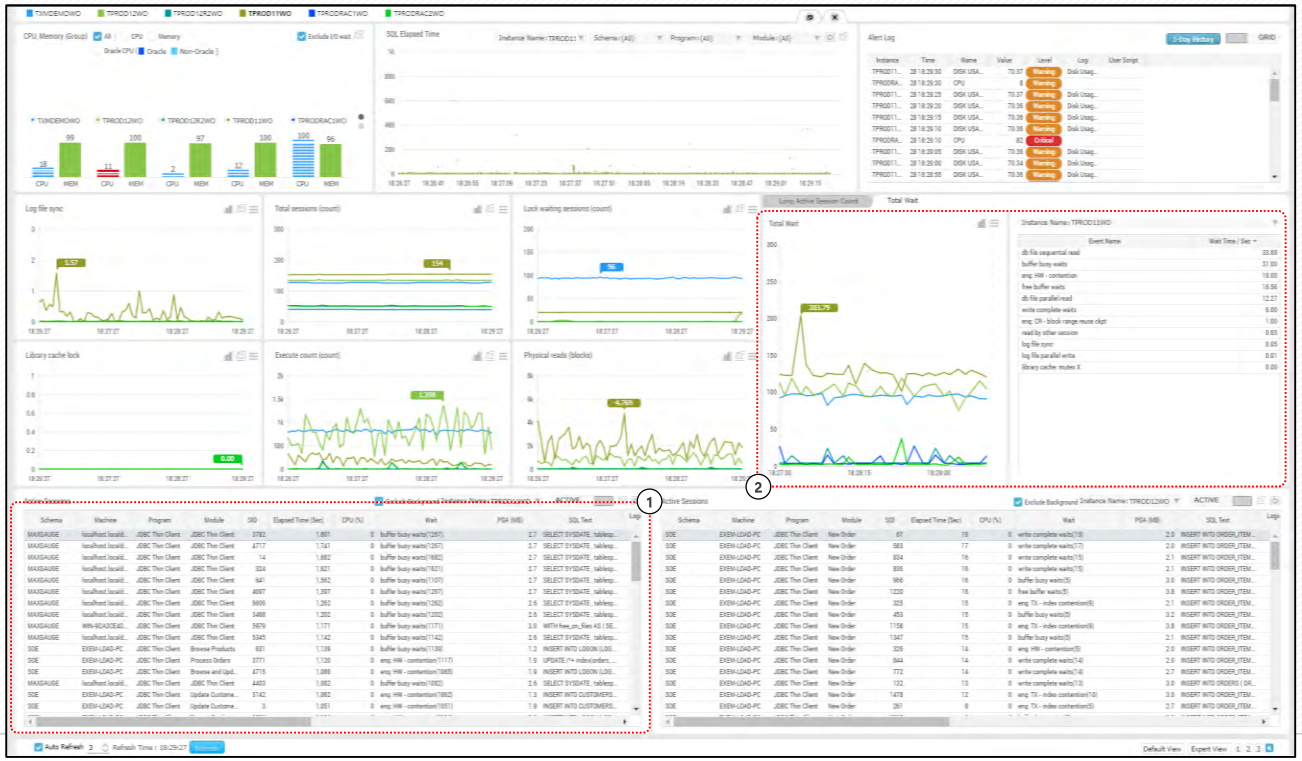
Real Time-Monitor에서 SQL Elapsed Time이 급증하였습니다.
Wait Time이 긴 SQL들을 확인하고 싶습니다.

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

" Wait Time이 긴 SQL들을 확인하고 싶습니다."

Real Time Monitor 화면



✓ **장애발생 감지**

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

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

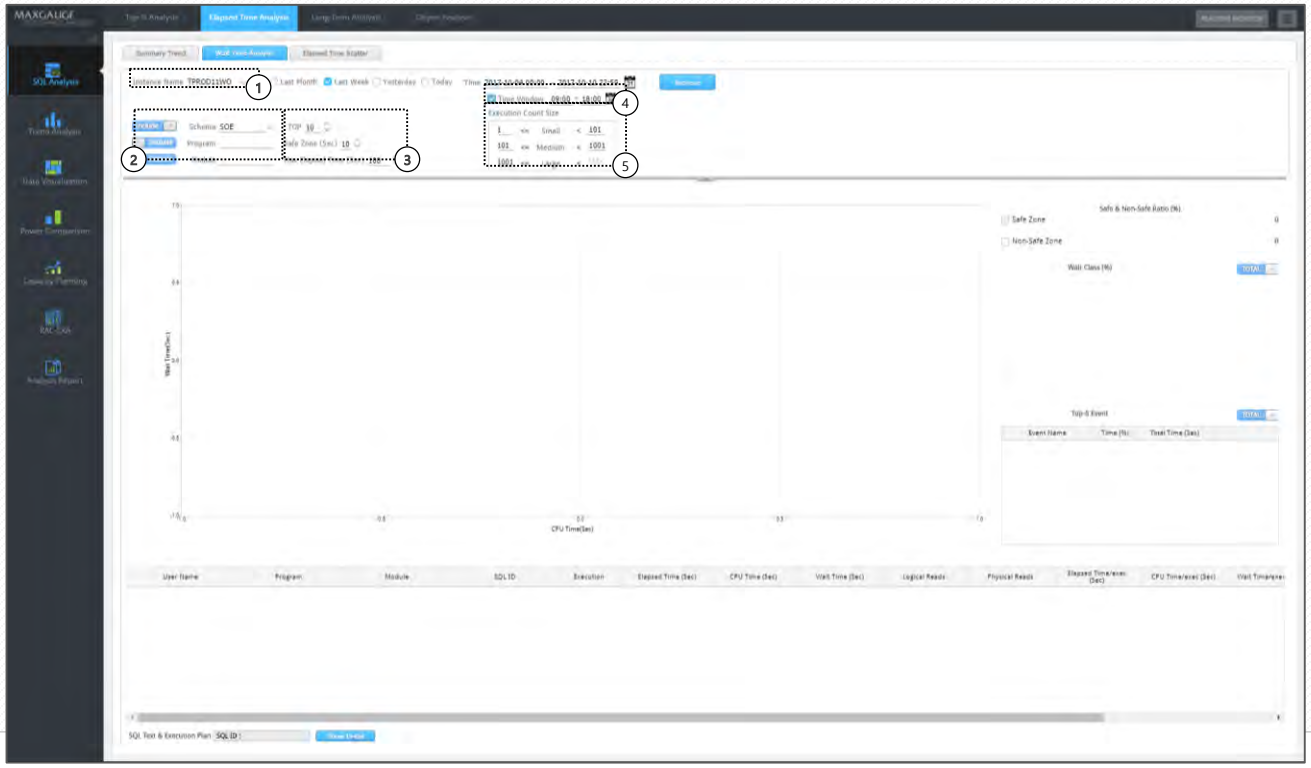
2 Total Wait Grid에서 Total Wait이 증가한 것을 확인할 수 있습니다.

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

" Wait Time이 긴 SQL들을 확인하고 싶습니다."

» Wait Time Analysis 검색 화면



✓ **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

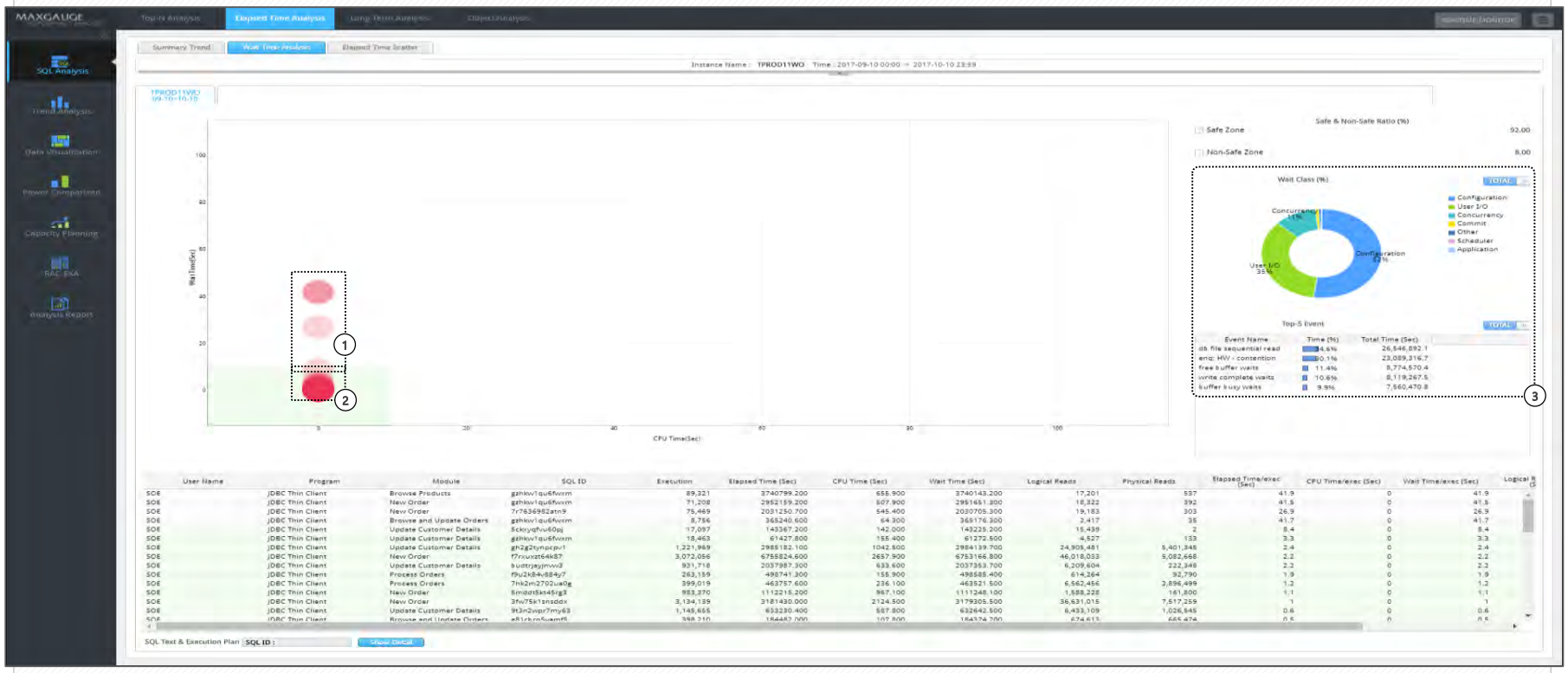
1. 대상 Instance (**TPROD11WO**)를 선택합니다.
2. Schema (**SOE**), Program, Module명을 선택적으로 *빼거나* 추가할 수 있습니다.
3. 수행시간이 100초 이하의 SQL 중 Top-10 SQL만 출력합니다.
4. 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (**09:00 ~ 18:00**) 을 선택적으로 선택할 수 있습니다.
5. SQL별 수행횟수에 따라 보여지는 원의 크기를 설정할 수 있습니다. 위의 설정한 조건으로 **Wait Time Analysis를 실행합니다.**

" 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

Wait Time Analysis 조회 화면



1 Wait Time 30초 이상인 SQL을 차트에서 확인할 수 있습니다. (4건)

2 Safe Zone에 위치한 SQL은 Wait Time이 짧아 성능상 문제가 없습니다. (6건)

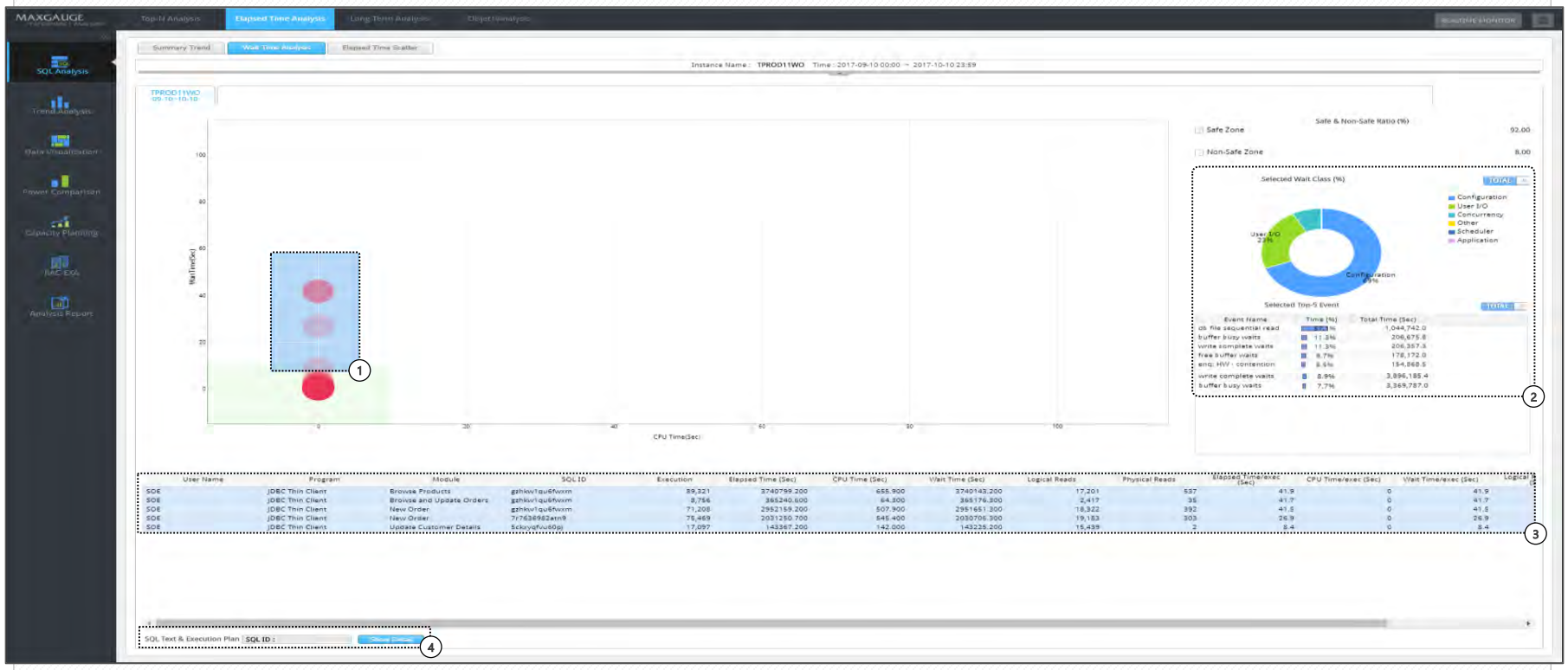
3 추출된 Top-10 SQL에 대한 Total Wait Class와 Top 5-Event를 확인할 수 있습니다.

" Wait 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

Wait Time Analysis 조회 화면



1 Wait Time 30초 이상인 SQL들을 마우스로 드래그합니다.

2 추출된 4건의 SQL에 대한 Selected Wait Class와 Top 5-Event를 확인할 수 있습니다.

3 추출된 4건의 SQL에 대한 개별 SQL의 일량 정보 확인할 수 있습니다.

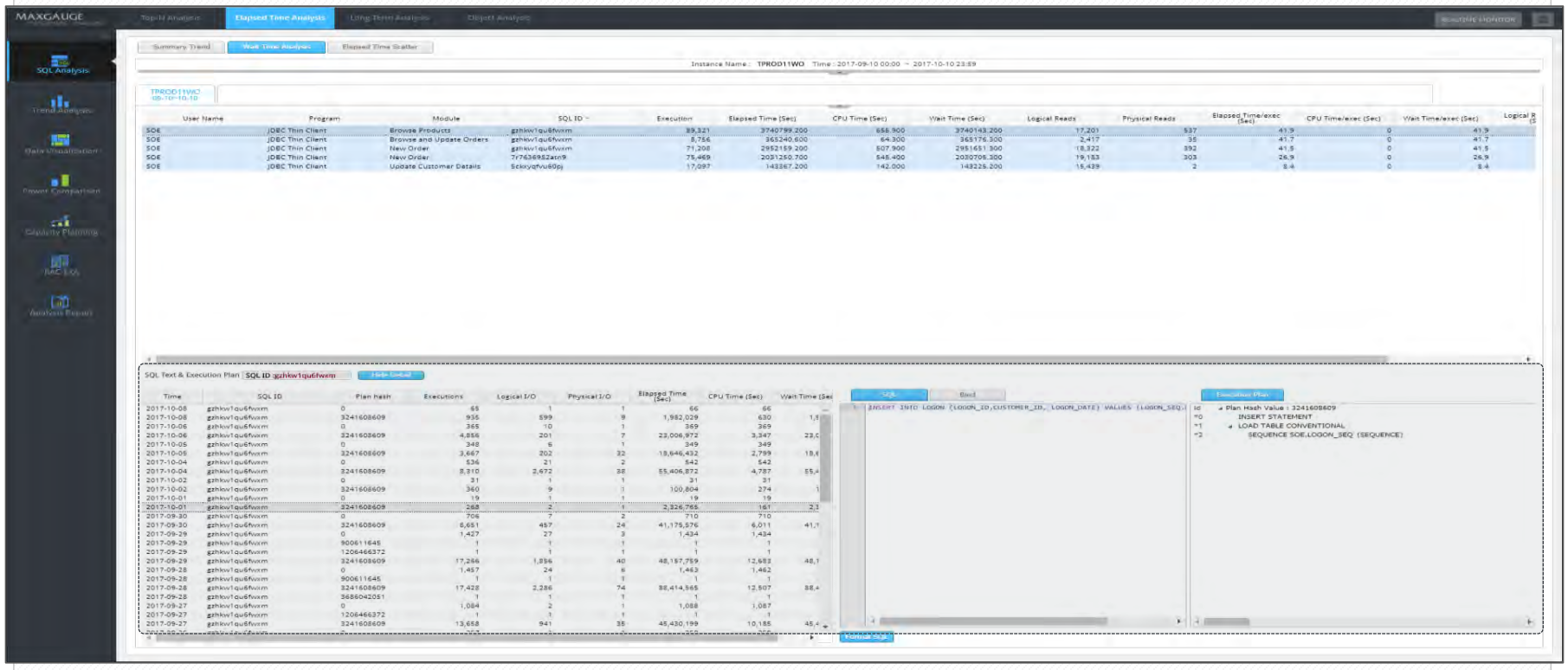
4 1-SQL을 선택한 다음, Show Detail을 선택합니다.

" 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

Wait Time Analysis 조회 화면



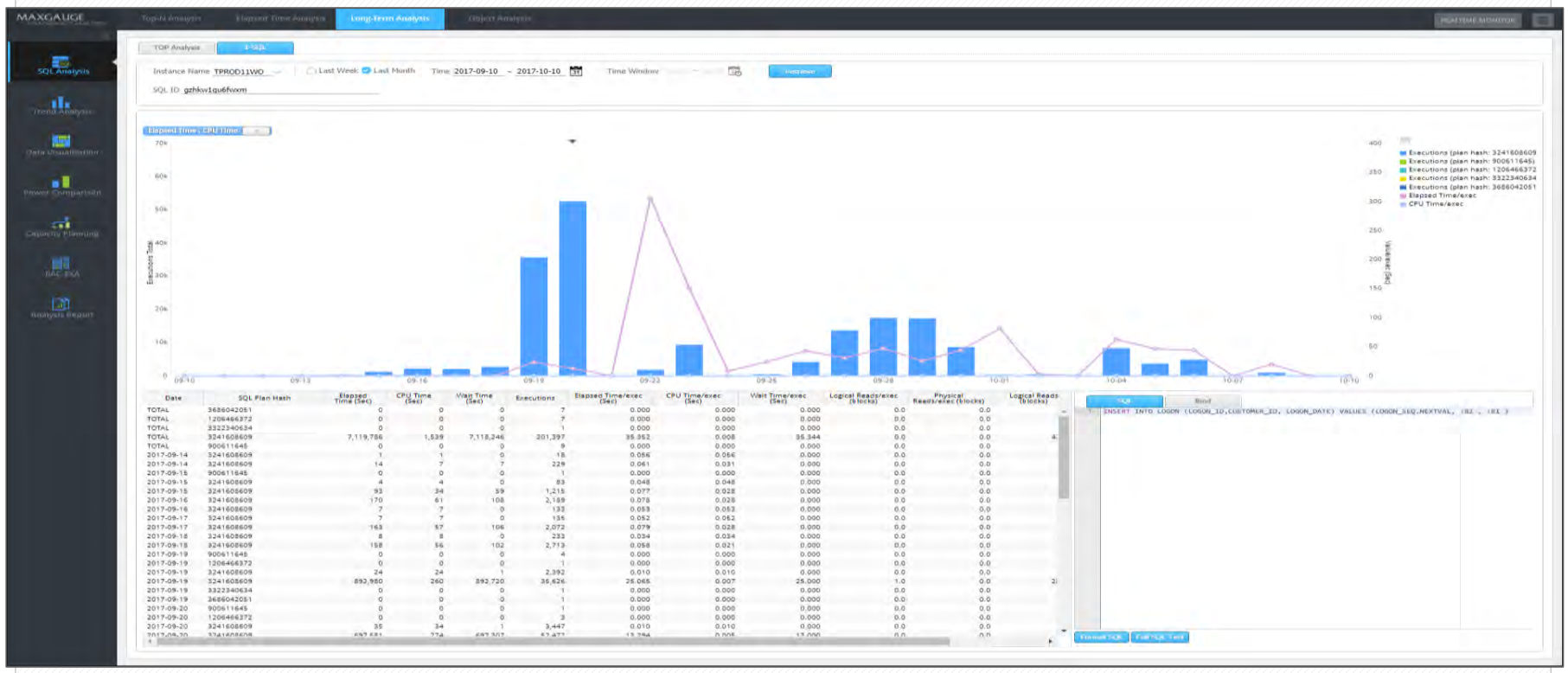
✓ 해당 SQL 일량 정보 및 SQL Plan과 Bind 정보를 확인할 수 있습니다.

" 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

Wait Time Analysis 조회 화면



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

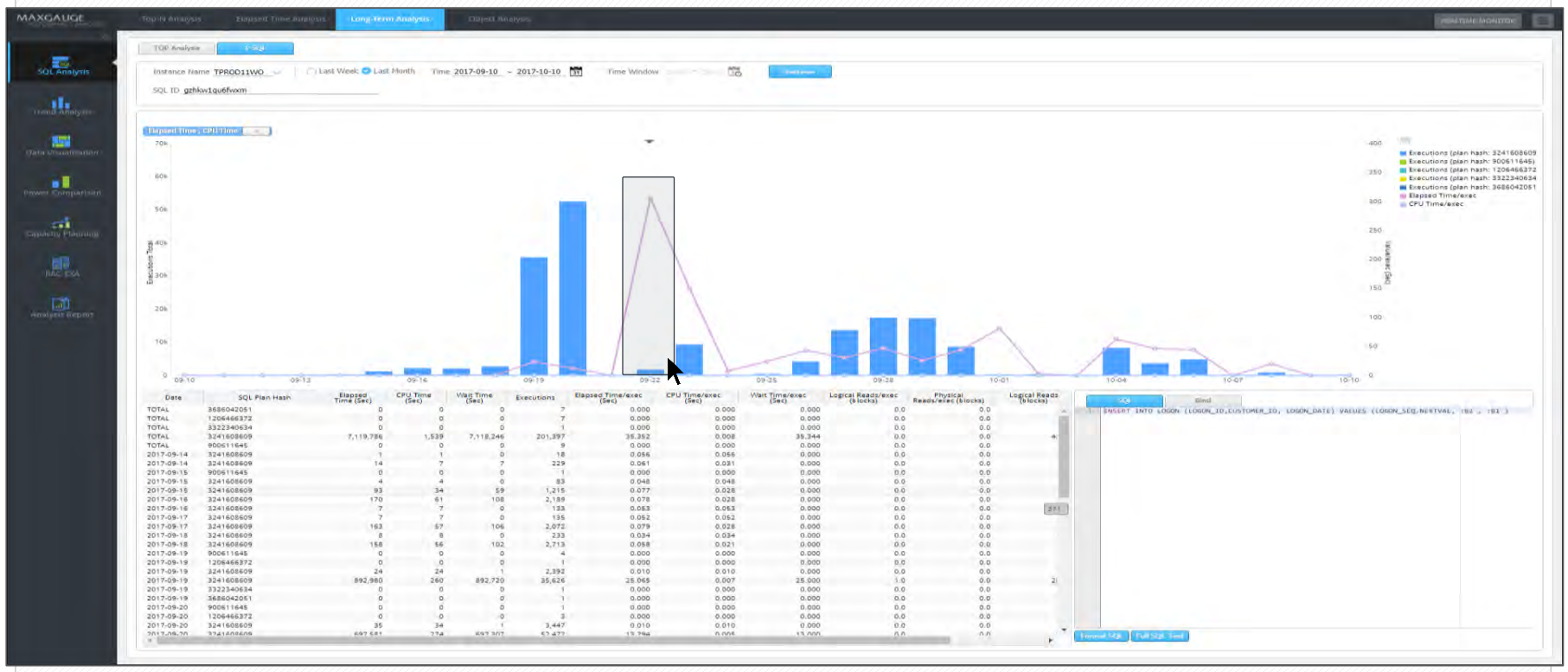
✓ 해당 SQL의 보다 상세한 일량 정보와 추이를 확인할 수 있습니다.

" 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

Wait Time Analysis 조회 화면



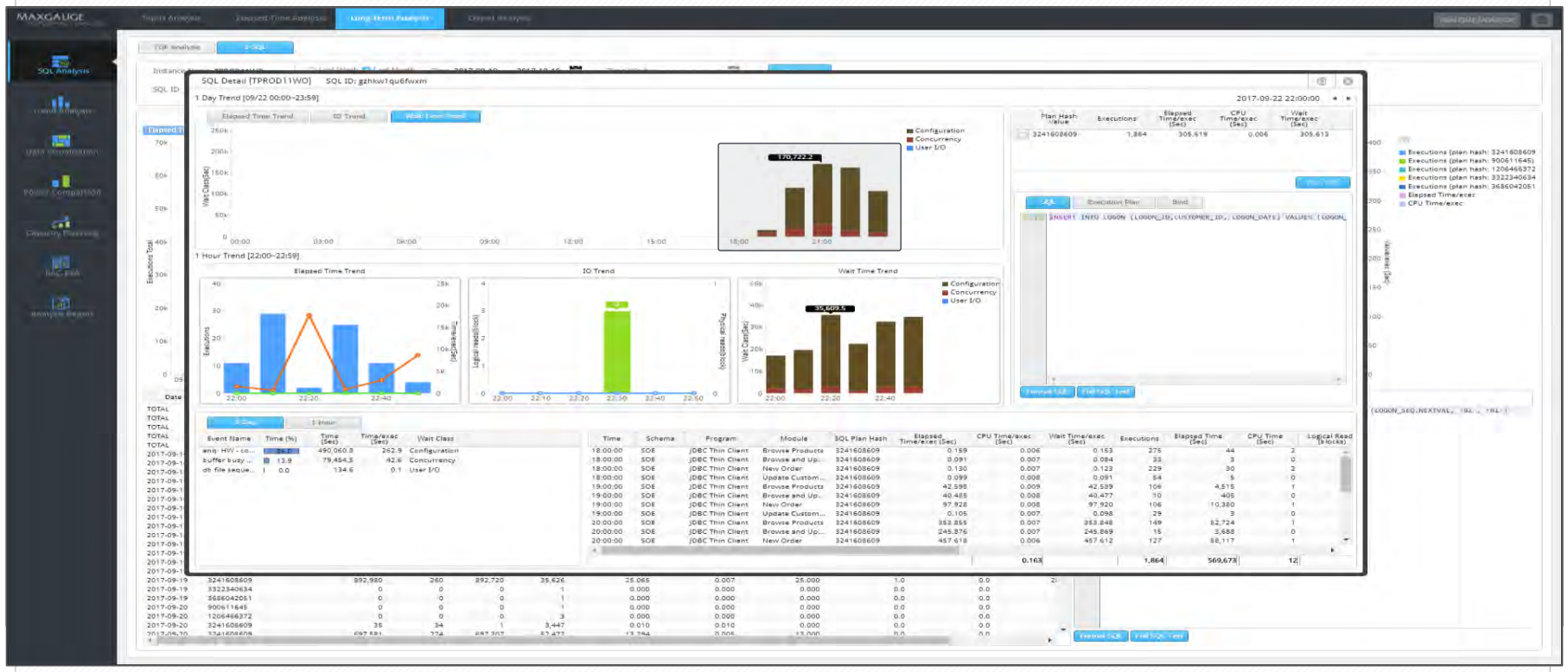
✓ Elapsed Time(분홍 실선)이 가장 컸던 9/22일을 상세 분석하기 위해 더블클릭 합니다.

" 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

Wait Time Analysis 조회 화면



✓ SQL Detail을 통해 선택한 날짜에 SQL의 상세한 정보를 확인할 수 있습니다.

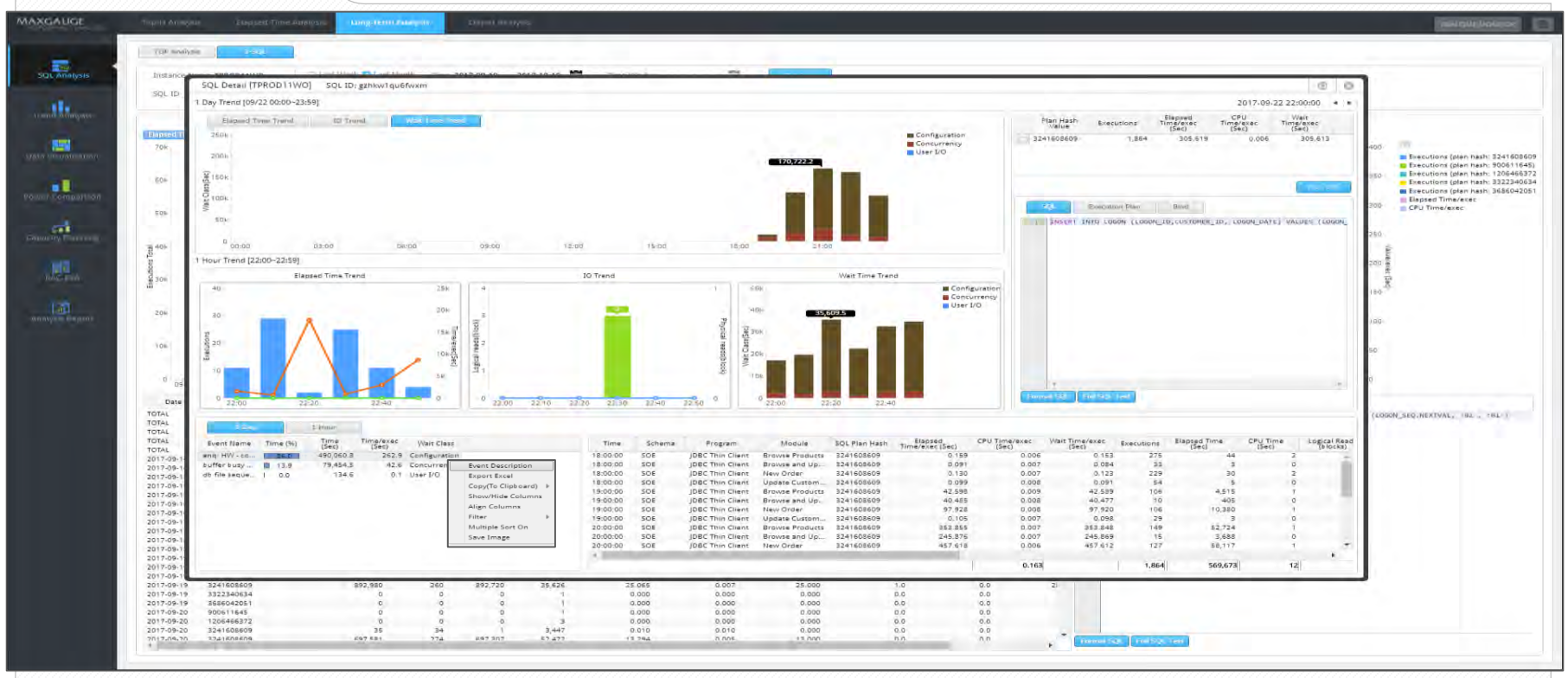
✓ 'Wait Time Trend Tab'을 선택하면 시간대별 주요 대기 Class와 Event를 확인할 수 있습니다.

" Wait 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

Wait Time Analysis 조회 화면



✓ "enq : HW – contention" Event를 대기하는 시간이 긴 것을 알 수 있습니다.

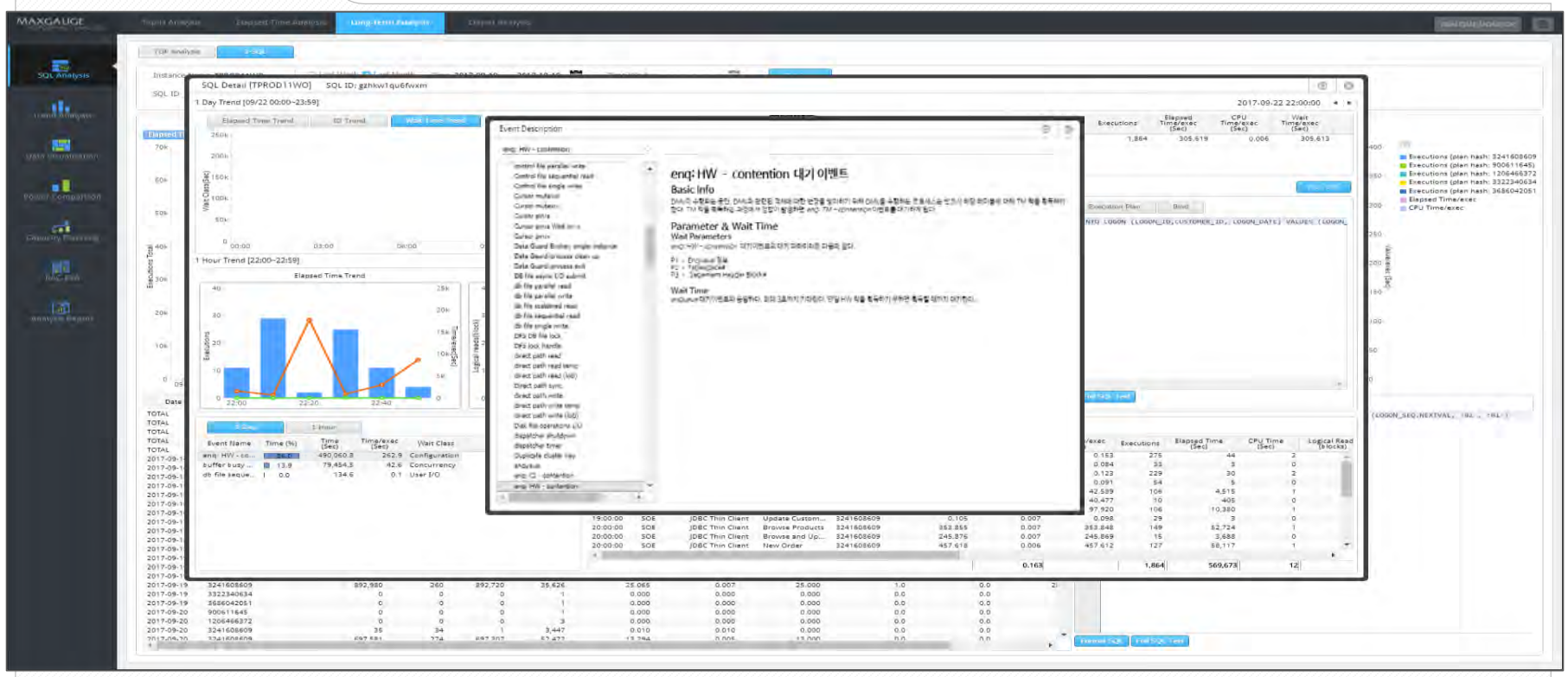
✓ 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 Event Description으로 연계합니다.

" Wait 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

Wait Time Analysis 조회 화면



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

MAXGAUGE Practical Guide

Trend Analysis [Performance Analyzer]

PRODUCT
DOCUMENTATION

Practical Guide

Trend Analysis

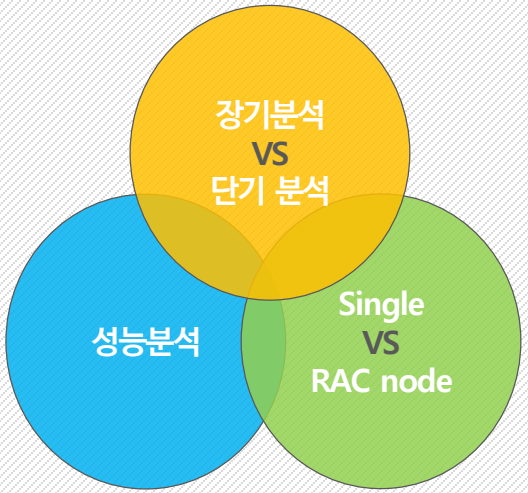
Contents

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

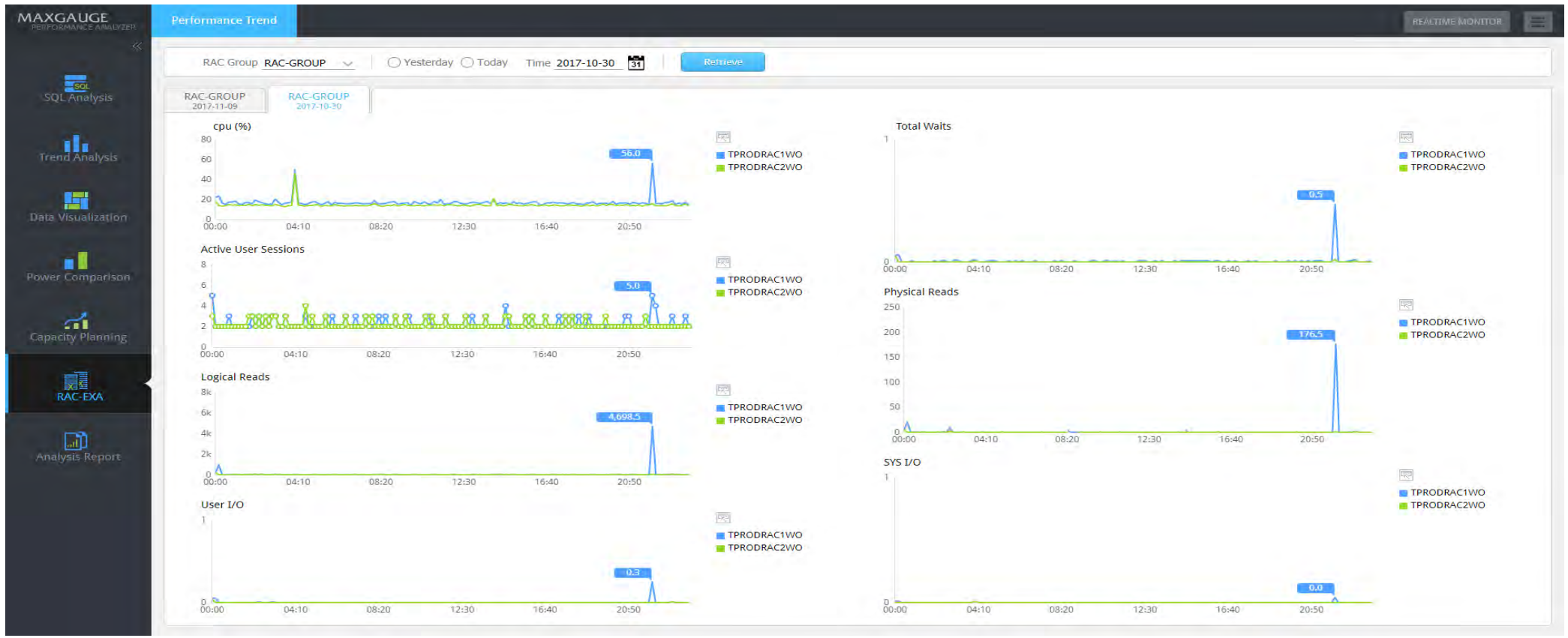
Performance Trend ?

- 시간대별 성능 지표 제공을 통해 **부하 시점에 대해 직관적인 파악**이 가능 하다. (시간대별 성능 지표 비교를 통해 **급격한 지표 변동**을 통해 부하시점을 한눈에 파악가능)
- 시스템의 성능 부하 시점 파악은 물론이고, 부하 발생 시점의 **상세 분석**까지 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.

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 Trend** 에서 사용 할 수 있다.

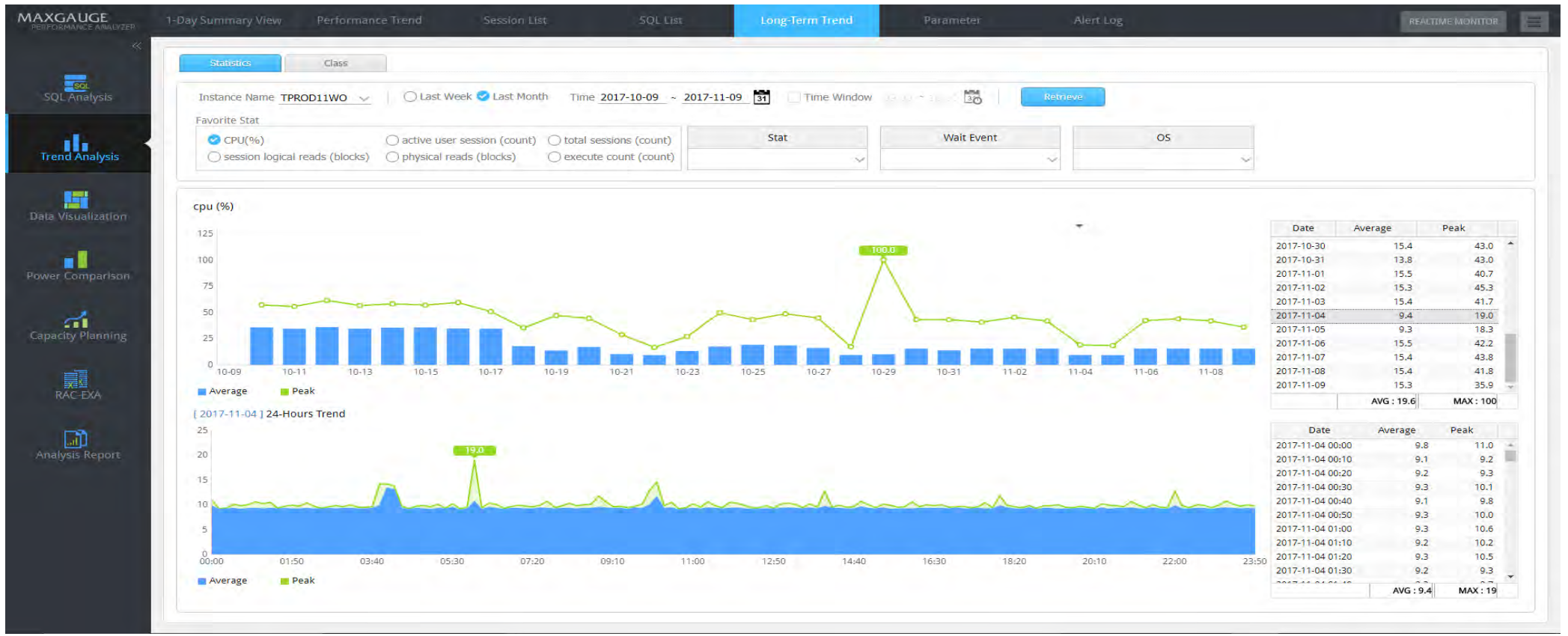
RAC Trend



Long-Term Trend ?

- 선택한 지표의 일별 성능 추이가 제공되어, 장기간의 **자원 사용량 추이**를 확인할 경우 용이합니다. (주 단위, 월 단위 Trend 비교를 통해 대상 서버의 **자원 사용 파악**이 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 Trend** 에서 사용 할 수 있다.

Long-Term Trend



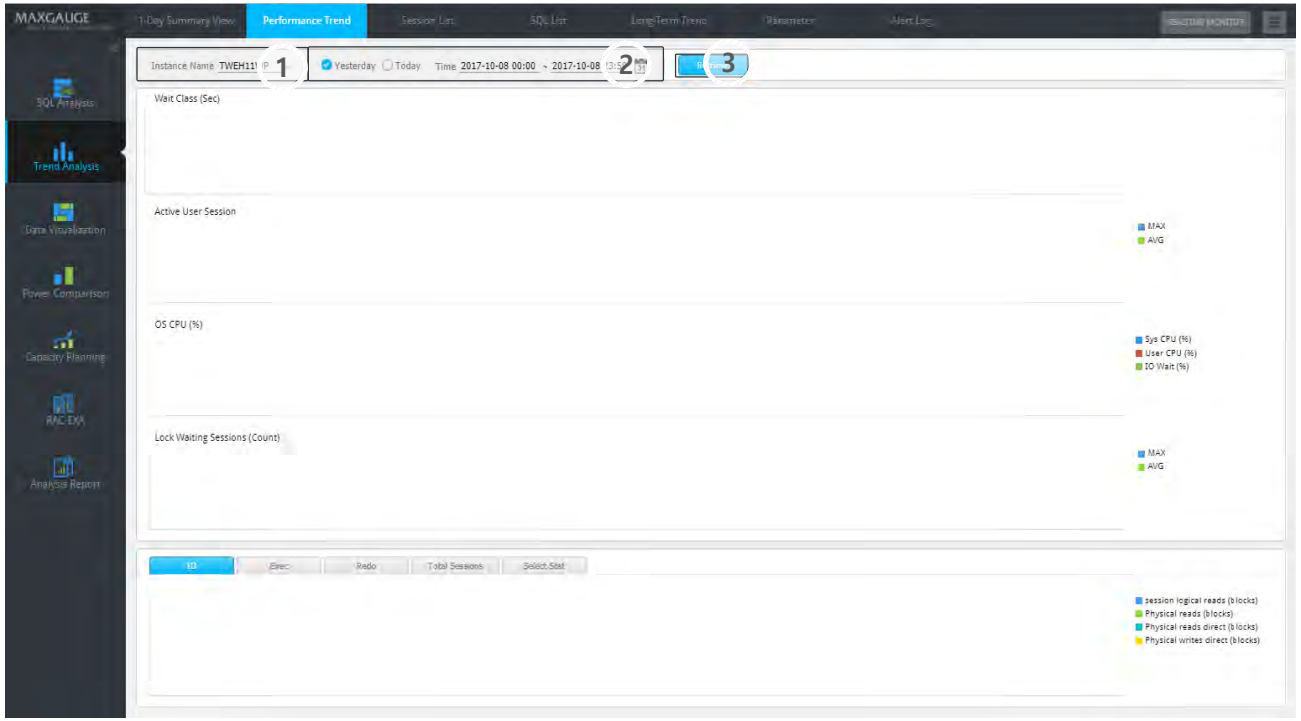
Trend 비교를 통해 DB 성능 분석을 할 수 있습니다!

Top-down 방식으로 DB에 대한 성능 분석이 이루어 집니다.

Trend Analysis의 활용

Performance Trend 의 사용 방법

» 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 Trend를 실행**합니다.

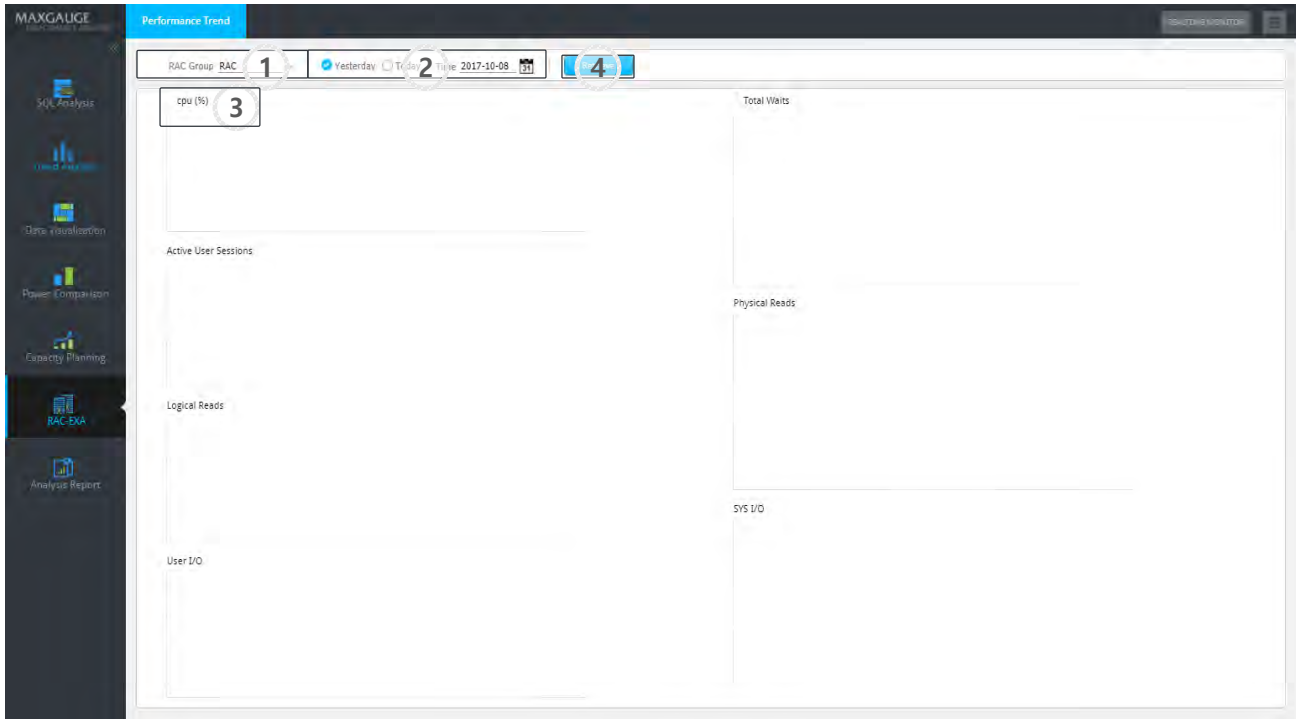
Performance Trend 의 사용 방법

Performance Trend 검색 화면

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



1 분석을 원하는 RAC 그룹을 선택합니다.

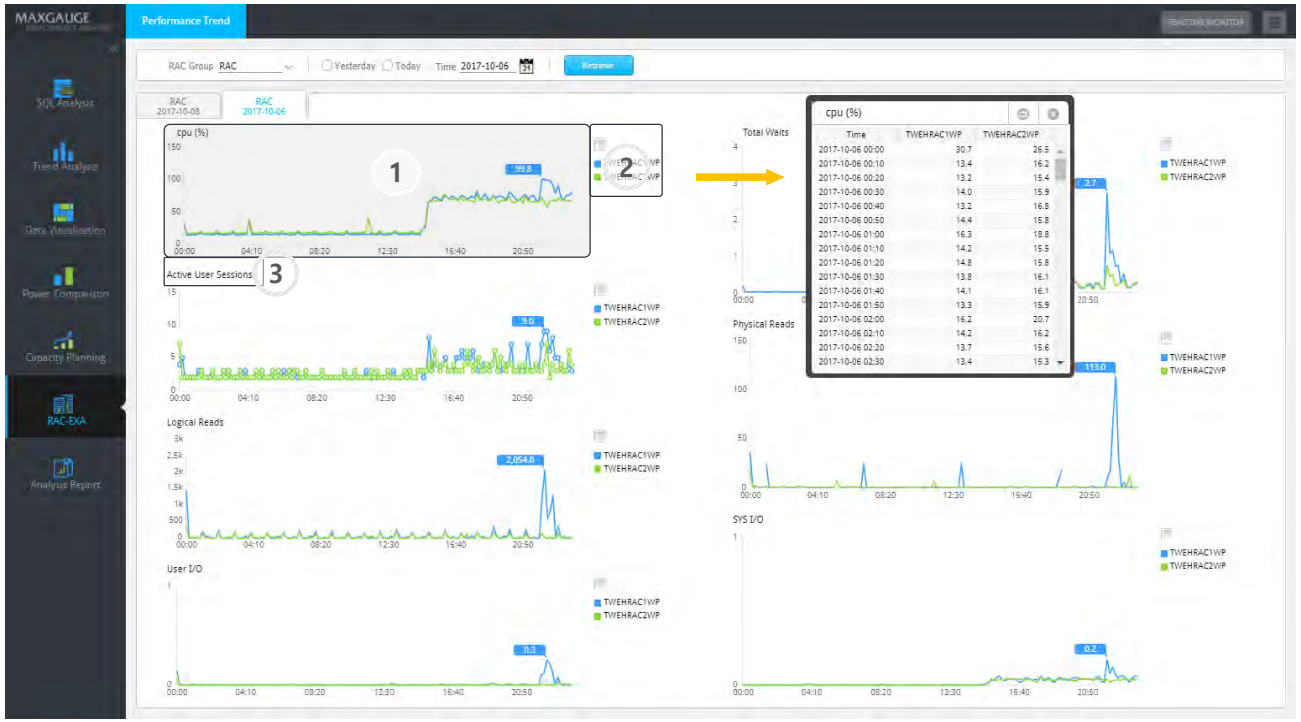
2 분석을 원하는 날짜를 선택 합니다. (**달력 선택**을 통해 데이터가 쌓이는 모든 날짜의 RAC 그룹 비교가 가능 합니다.)

3 성능 지표를 더블 클릭 시 (추천 지표와 DB stat , Wait Event, OS stat의 비교분석이 가능합니다.)

4 위의 설정한 조건으로 **RAC-EXA Trend를 실행**합니다.

RAC-EXA Trend 의 사용 방법

» 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.

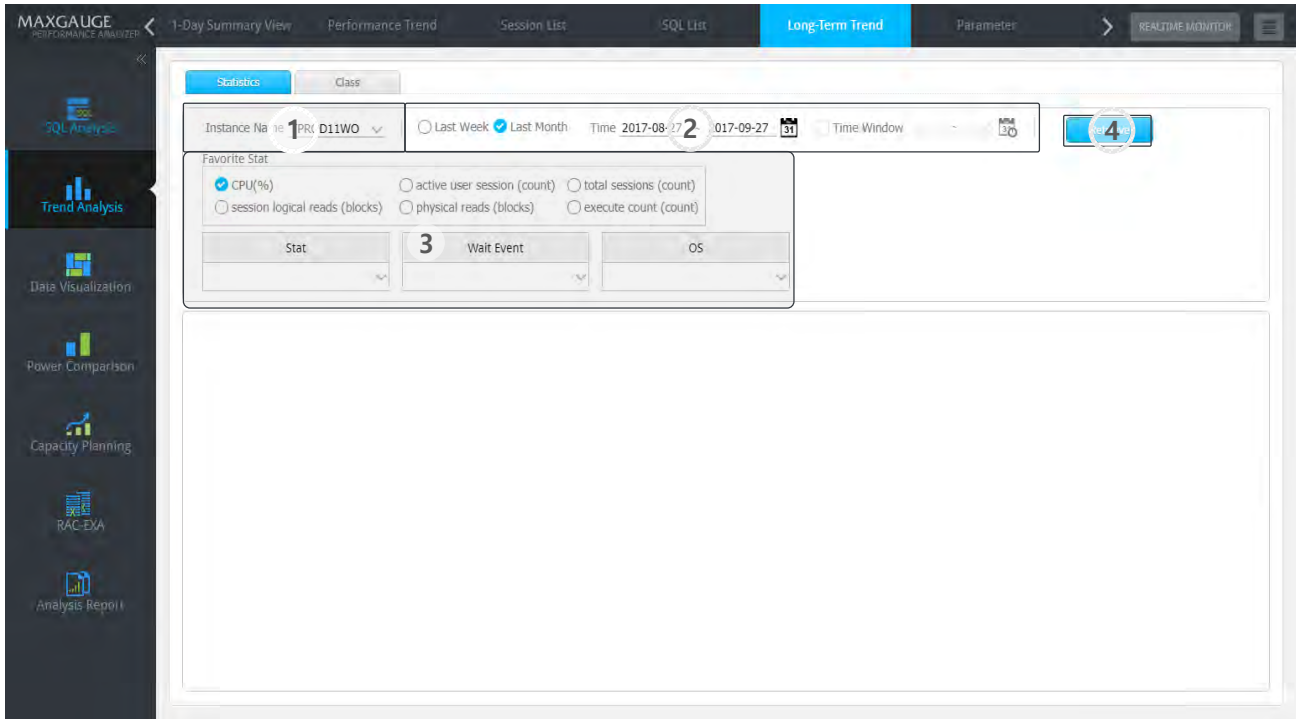
2 팝업 아이콘 선택 시 **성능 지표의 AVG 값**이 리스트 형태로 출력 됩니다.

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 화면



1 분석 대상 Instance를 선택합니다.

2 분석을 하고자 하는 기간 설정을 합니다. (**Time Window 설정**을 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 실행**합니다.

Long-Term Trend 의 사용 방법

» Long-Term Trend 조회 화면

1 분석을 원하는 지표의 일별 MAX, AVG 값이 **그래프 형태**로 출력 됩니다. (**최초 출력 일자**는 지표의 MAX값을 기록한 날짜가 출력 됩니다.)

2 분석을 원하는 지표의 일별 MAX, AVG 값이 **리스트 형태**로 출력 됩니다.

3 선택된 날짜의 일일 지표 추이가 **그래프 형태**로 제공 됩니다.

4 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.

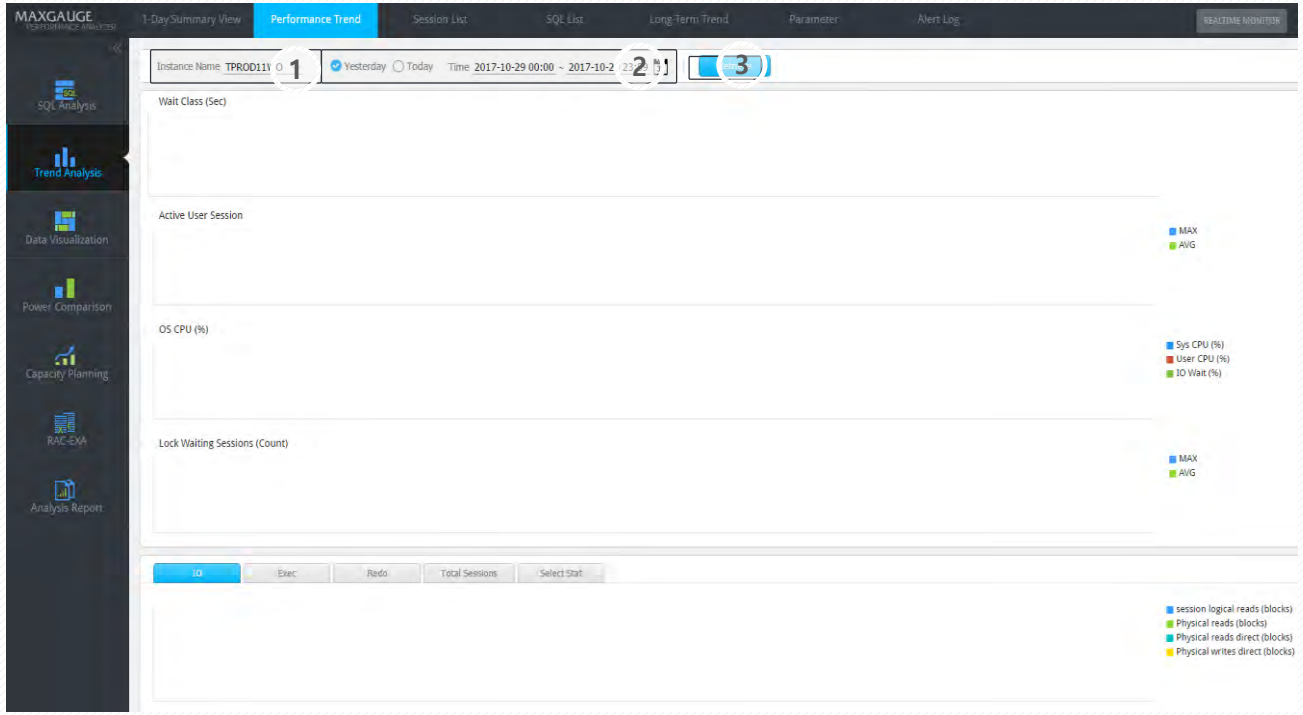
CPU가 급격히 증가한 원인을 알고 싶어요

STEP

1. 검색 조건 설정
2. 데이터 분석
   - 성능 추이 확인
   - "DB 장애 시점" 확인
   - "상세분석"
     - Trend Analysis 연계
     - SQL Detail 연계

" CPU가 급격히 증가한 원인을 알고 싶어요 "

Trend Analysis 검색 화면



✓ 시나리오

INSTANCE : TROD11WO

문제 발생일 : 2017년 10월 29일

✓ 목표

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

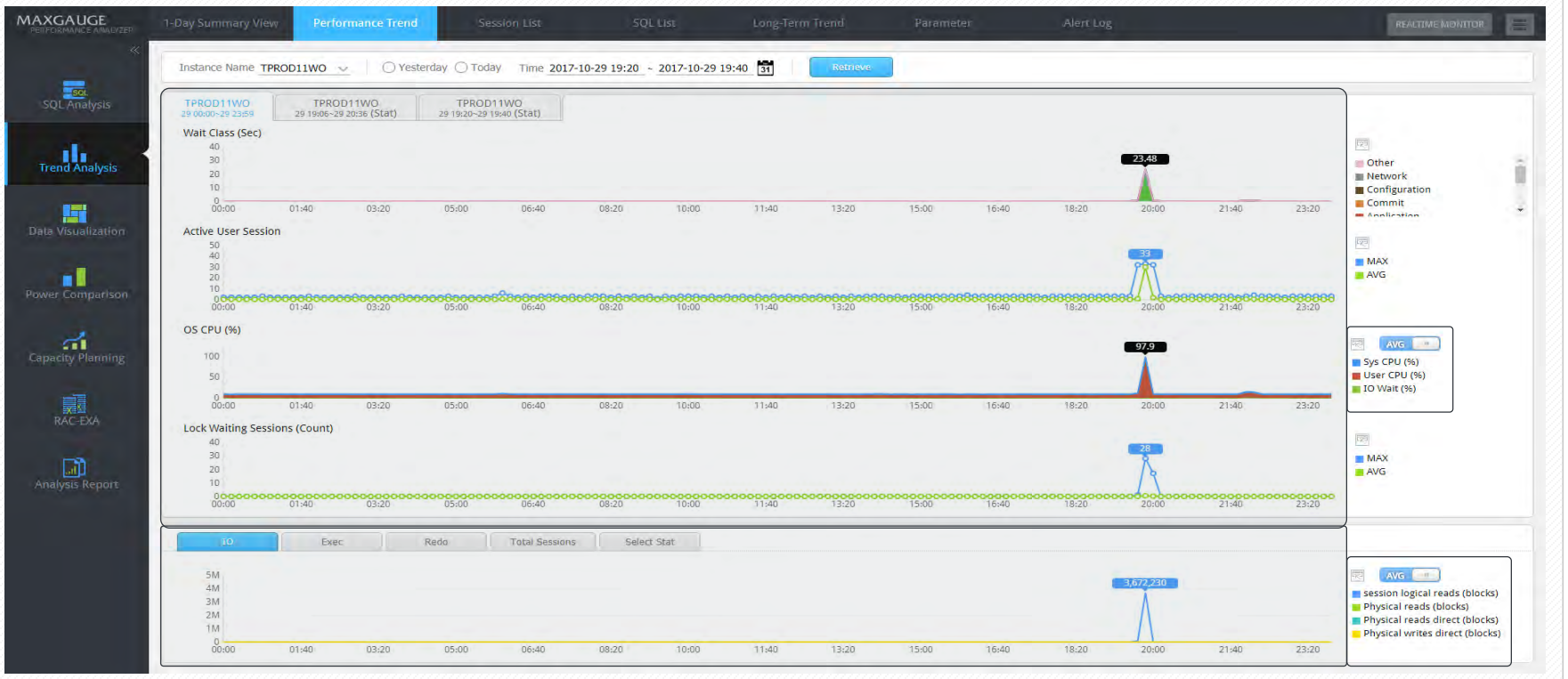
1. 성능 분석을 진행할 인스턴스를 선택 합니다.
2. 성능 분석을 진행할 날짜를(10월 29일) 선택합니다. (Performance Trend는 한 서버에 대해 **최대 2일**에 대한 성능 비교가 가능 합니다.)
3. 위의 설정한 조건으로 **Performance Trend를 실행**합니다.

"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



✓ 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.

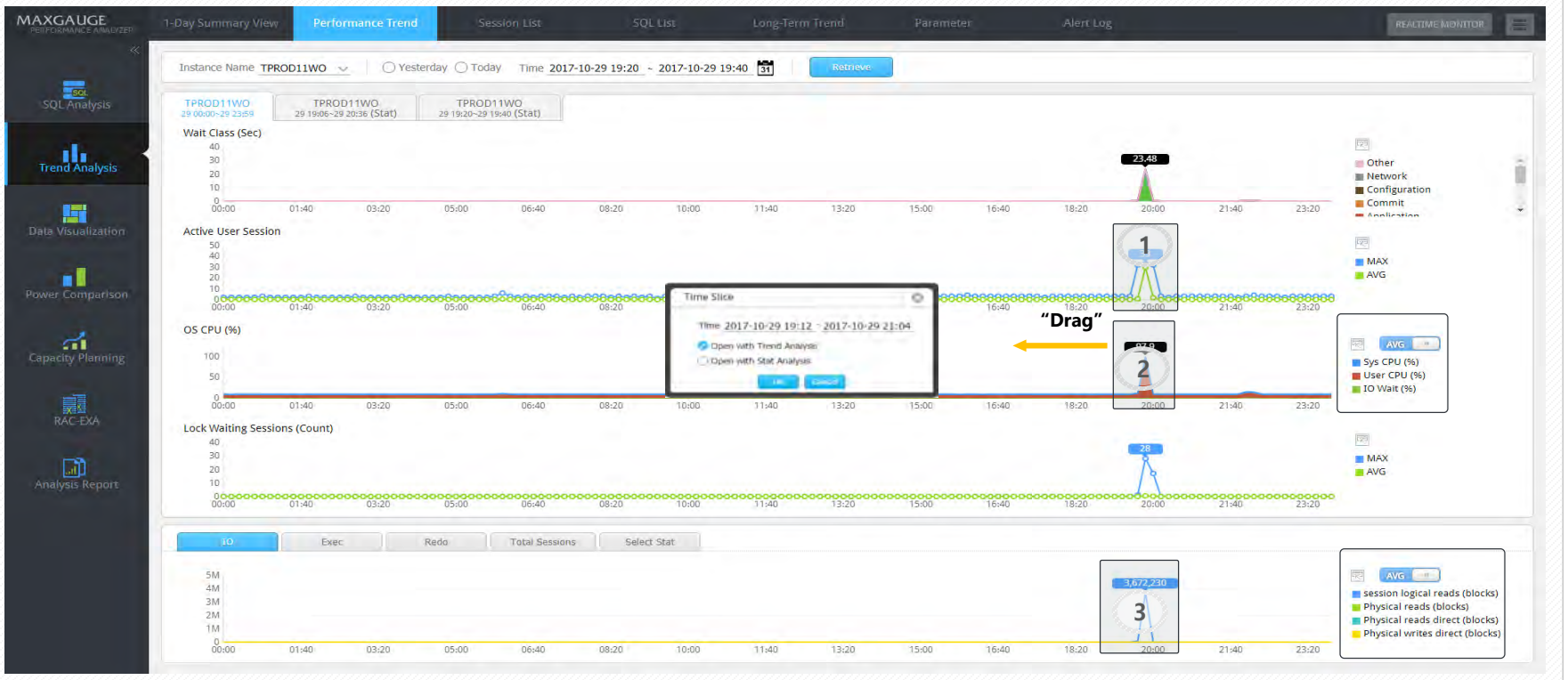
✓ 19시 40분경 **OS CPU 지표**와 **DB 성능지표**의 추이가 급격히 증가 된 것으로 보아, 해당 시점에 어떤 문제가 있다는 것을 **직관적**으로 예상할 수 있습니다.

"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

» Trend Analysis 조회 화면



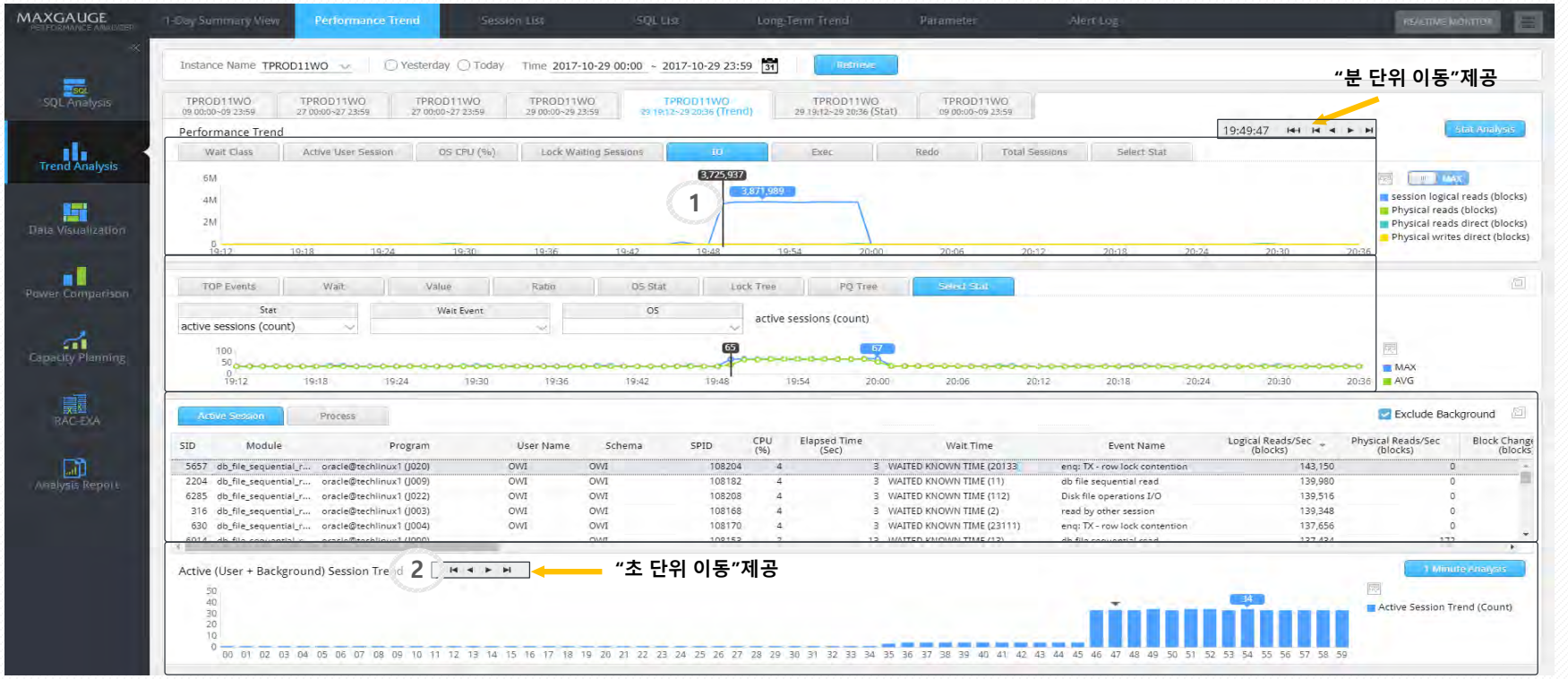
✓ **10월19일 19시40분 부터, CPU, Active Session, Session Logical Reads 등 부하의 척도가 되는 주요 지표 3가지가 동시에** 증가 된 것이 확인 됩니다.

✓ 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"

» Trend Analysis 조회 화면



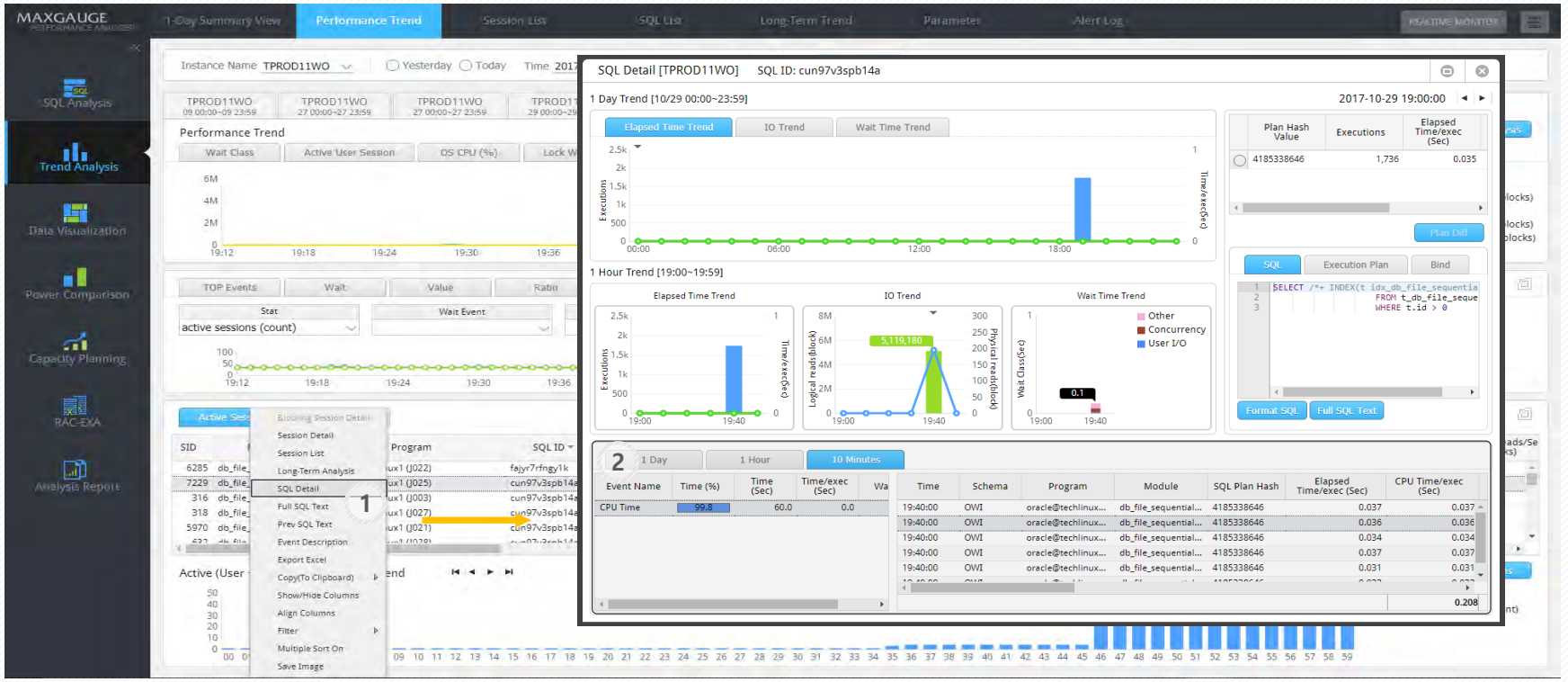
1 Time Slice는 Drag 한 구간까지의 Trend만 보여집니다. 그리고 하단에는 당시의 수행된 Active Session 정보가 보여집니다. 우선 CPU가 급격히 증가한 시점 이동한 후 Active Session Trend에서 Active Session이 급격하게 증가한 초를 확인한 후, 막대 바를 클릭해 해당 시점으로 이동합니다.

2 **성능 이슈 발생직전의 초(19 시49분 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"

» Trend Analysis 조회 화면



1 I/O량이 높아, 부하의 원인으로 의심이 되는 Active Session의 경우 , 우 클릭 후 **SQL Detail로 연계**하여, 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 time**에 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.)

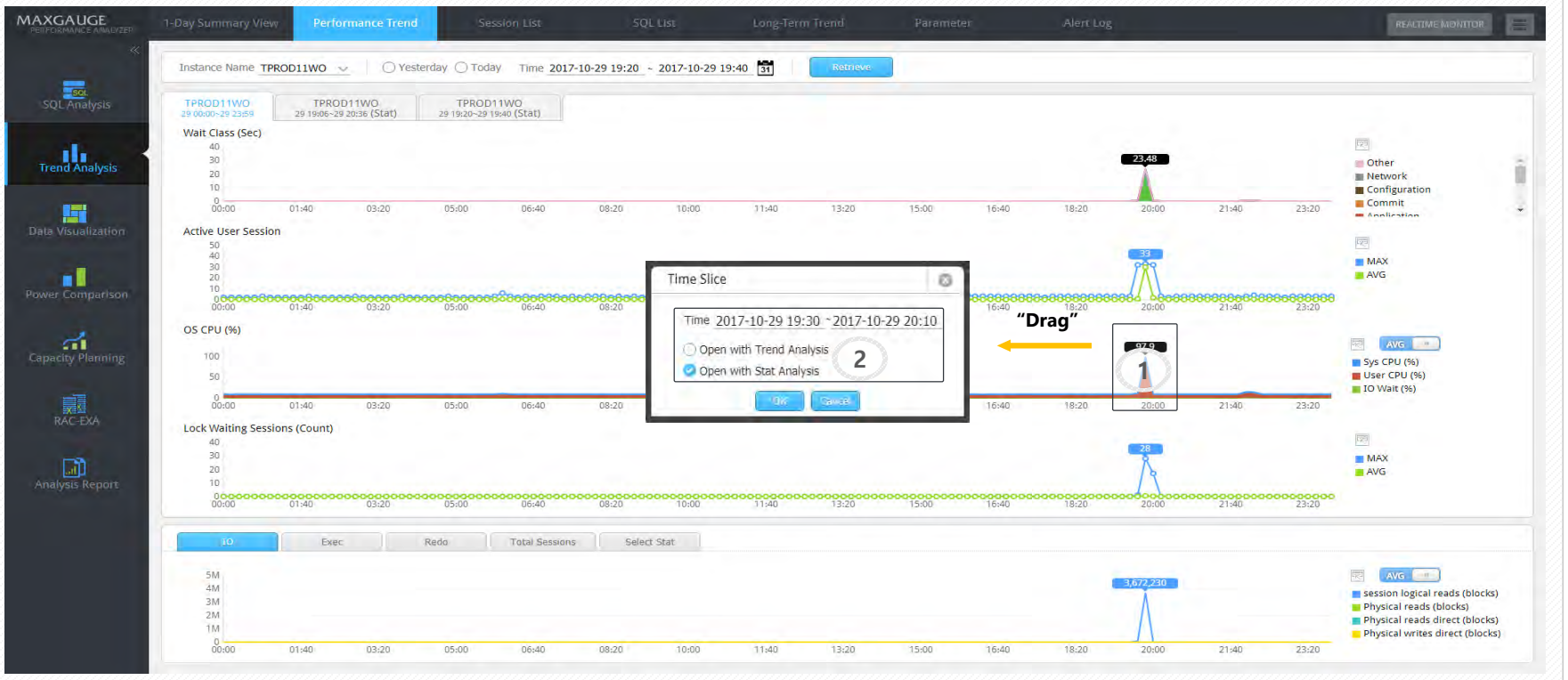
✓ 문제 SQL은 1회 수행당 부하는 작았으나 Execution이 높았습니다. 즉 부하 시점에 해당 SQL이 많이 수행된 것이 CPU가 급격히 증가한 원인임을 알 수 있습니다

"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

» Trend Analysis 조회 화면



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 SQL정보를 비교함으로써, 부하가 되는 SQL을 손쉽게 찾을 수 있습니다.

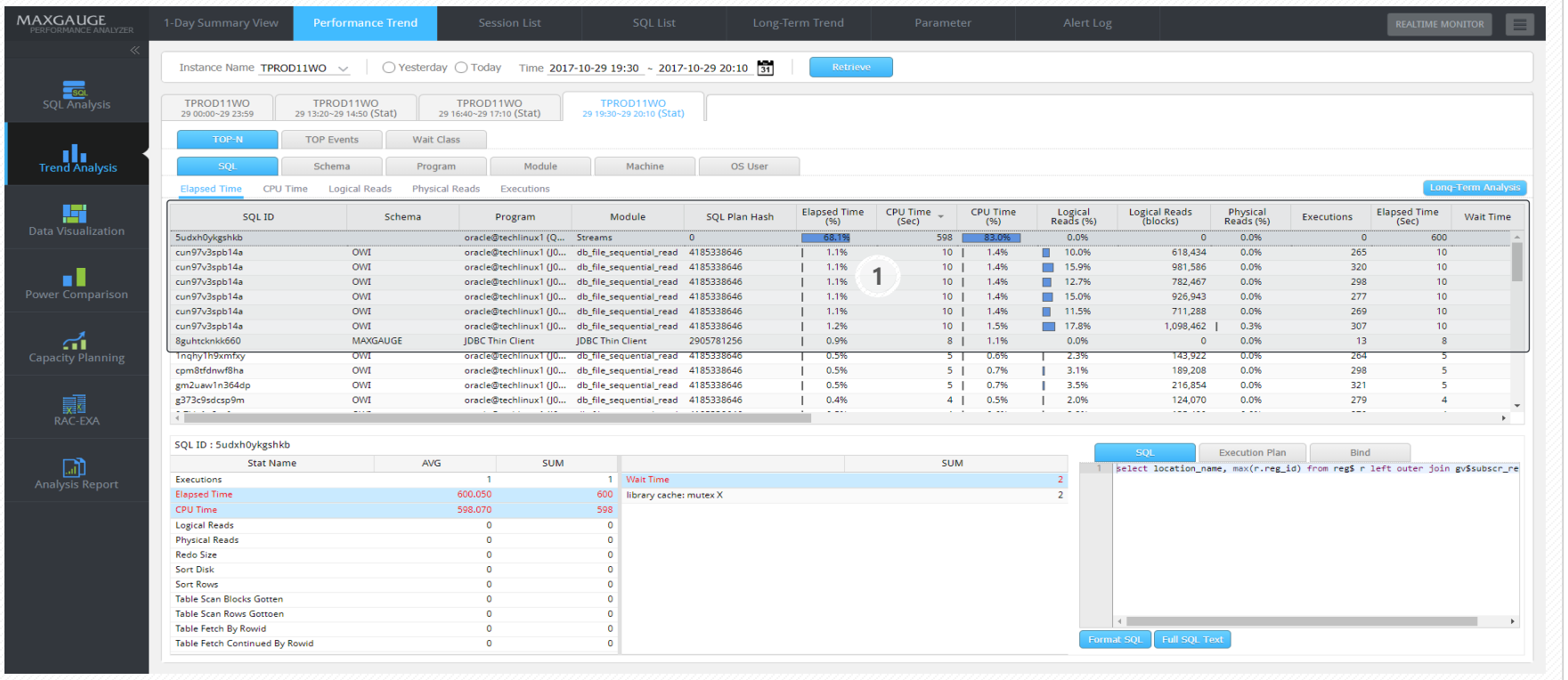
2 우선 성능 지표가 급격히 증가한 20시 부근을 **"Drag"한 후 팝업 창에서는 Top SQL을 바로 볼 수 있게** Stat Analysis을 수행할 수 있도록 합니다.

"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



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 것**을 확인 할 수 있습니다.

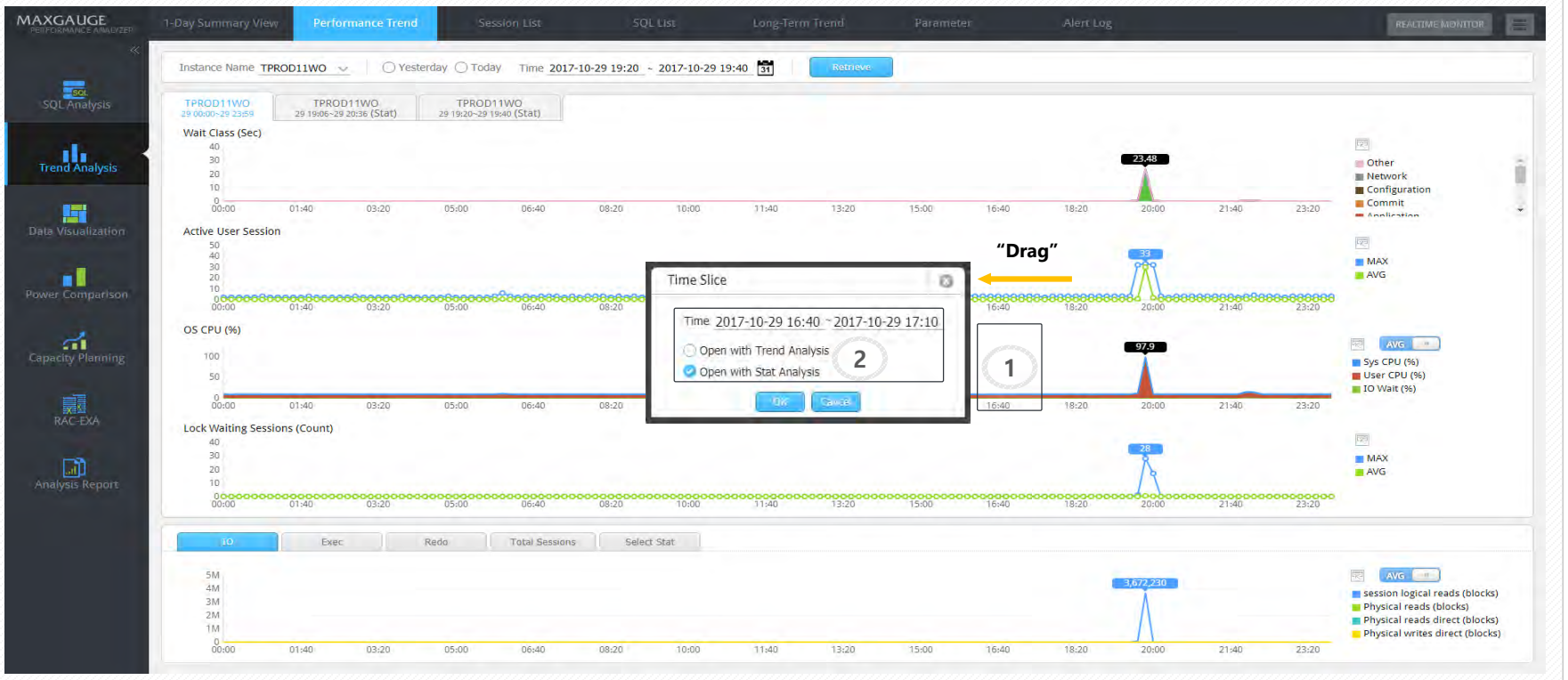
(Session Logical Reads가 높았다는 것은 실제 DB에서 Memory I/O가 높은 것을 의미하는데 이는 작업량이 높은 것을 의미합니다.)

"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**

» Trend Analysis 조회 화면



1 부하가 된 시점을 확인했다면, 이제는 **부하가 없던 시점을 확인**하여, 평상시에는 수행되지 않았던 SQL들이 존재하는지 파악하면 됩니다.

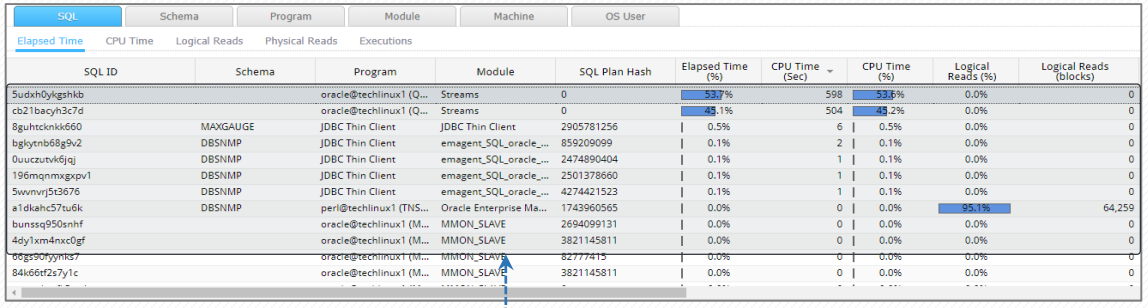
2 그럼 **"Drag"를** 통해 **정상적으로 DB가 운영된 시점 시점의 Stat Analysis** 를 확인해 보도록 하겠습니다.

"CPU가 급격히 증가한 원인을 알고 싶어요"

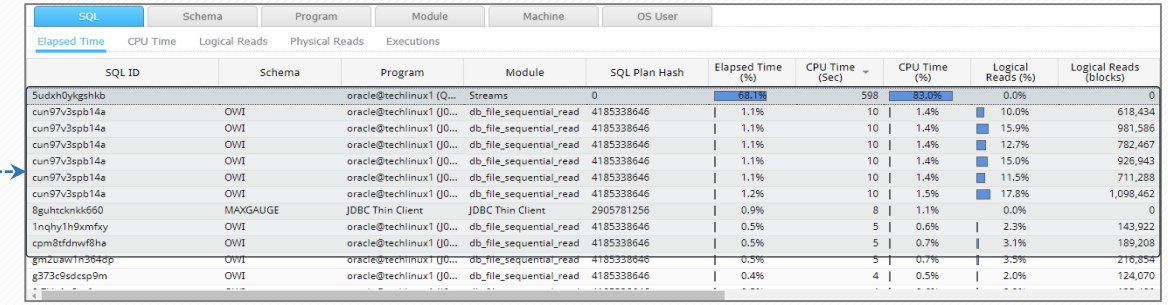
STEP

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



10월29일 DB 정상 시점 SQL(16시40~17시10분)



10월29일 CPU 부하 시점SQL(19시40~20시10분)

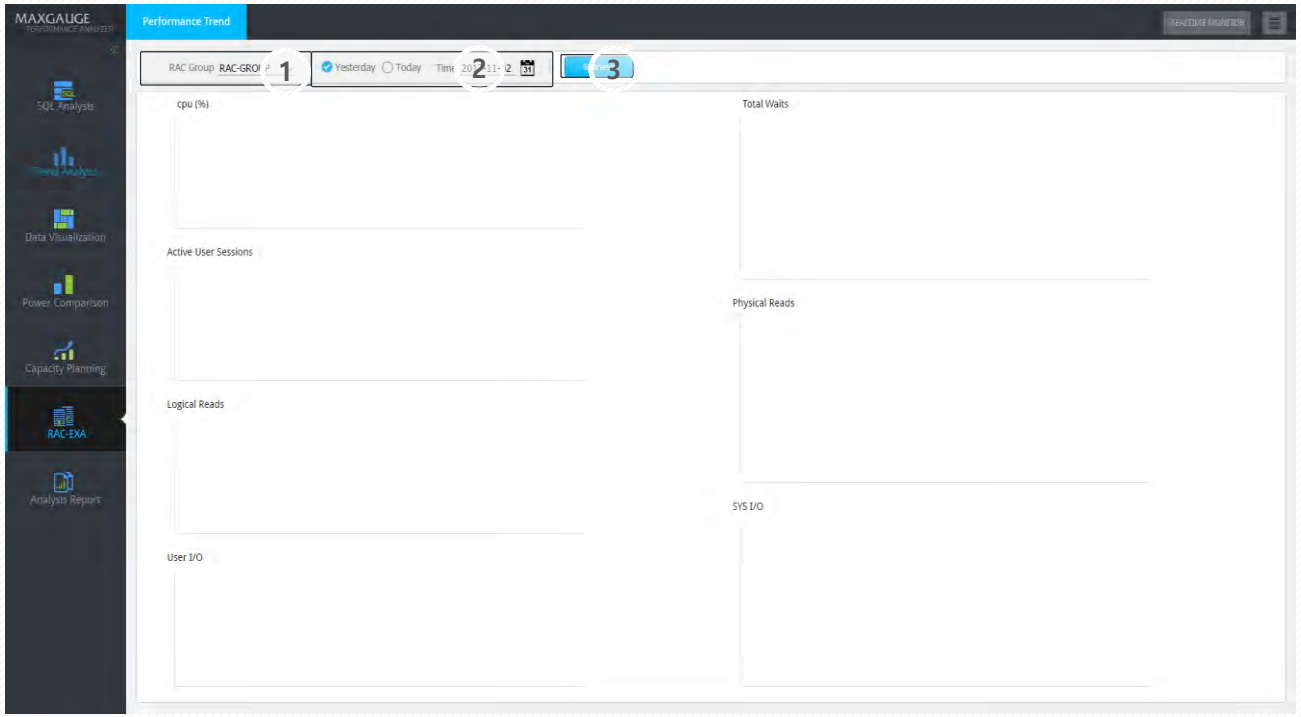
- ✓ 정상적 운영 시점과 CPU 부하 시점에 대해 **CPU Time으로 정렬을 하고, 두 시점에 대해 SQL문**을 비교를 할 수 있습니다.
- ✓ 부하시점과 정상시점의 Stat 비교하면, **문제가 되는 이벤트, 부하를 일으킨 SQL문**을 **쉽고 정확하게** 파악할 수 있습니다.
- ✓ 확인 결과 정상적인 구간에서는 보이지 않던 **OWI Schema 에서 수행된 SQL이** CPU 부하 시점 전체 CPU Time의 83%를 점유하고 있는 것으로 보아 해당 SQL이 많이 수행된 것이 CPU증가의 원인임을 확인 할 수 있었습니다.

실전 분석 사례 Case 2.

RAC 간의 Load Balance 및 성능을 확인 하고 싶어요.

" RAC 간의 Load Balance 및 성능을 확인 하고 싶어요 "

RAC Performance Trend 검색 화면



✓ **시나리오**

INSTANCE : TRODRAC1WO, TRODRAC2WO
문제 발생일 : 2017년 11월 10일

✓ **목표**

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

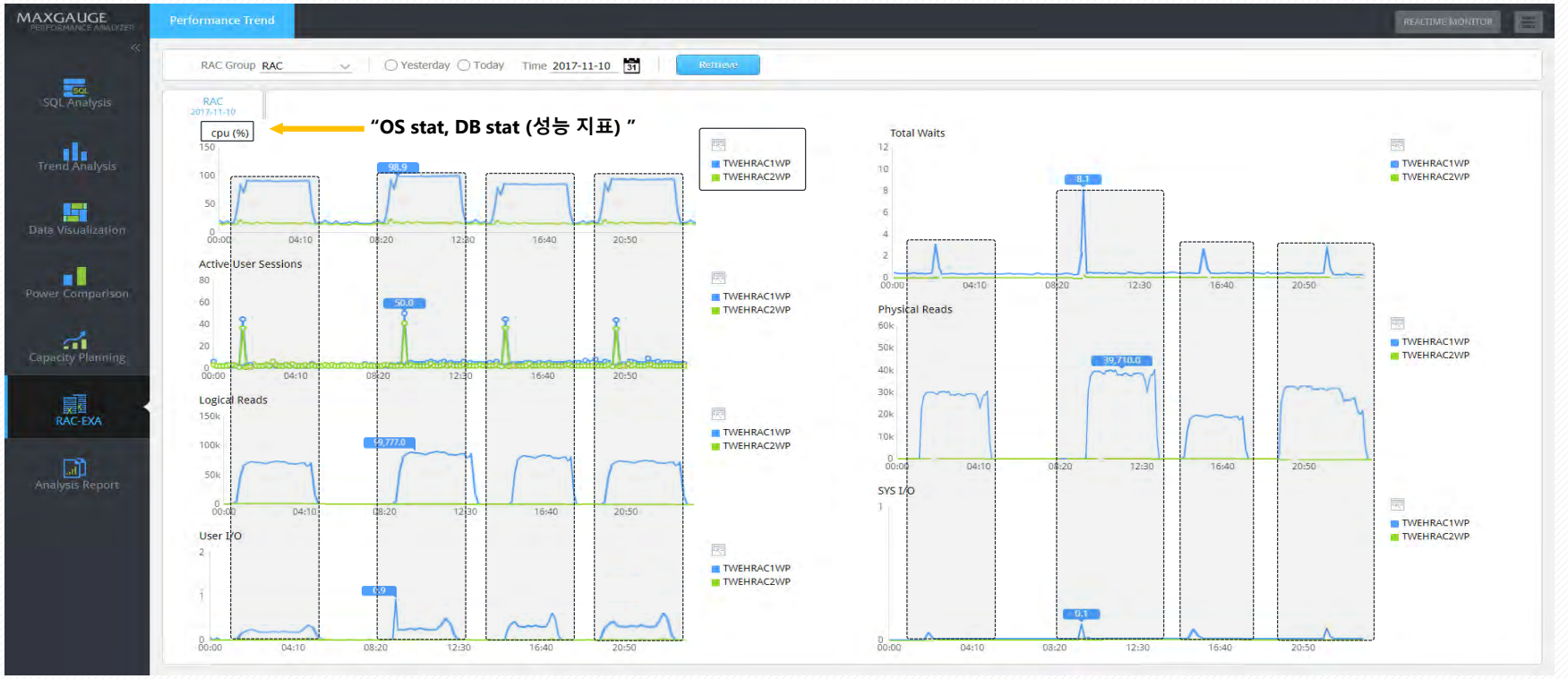
1 성능 분석을 원하는 **RAC 그룹을 설정** 합니다. (설치 시 RAC node간의 그룹 설정 가능)

2 분석을 원하는 날짜를 선택 합니다. (RAC 그룹에 속한 서버의 하루간의 성능 비교가 가능)

3 위의 설정한 조건으로 **Performance Trend를 실행**합니다.

" RAC 간의 Load Balance 및 성능을 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



✓ 위 화면은 RAC-Group에 속한 서버들의 OS stat , DB stat 의 성능 추이를 보여주고 있습니다.

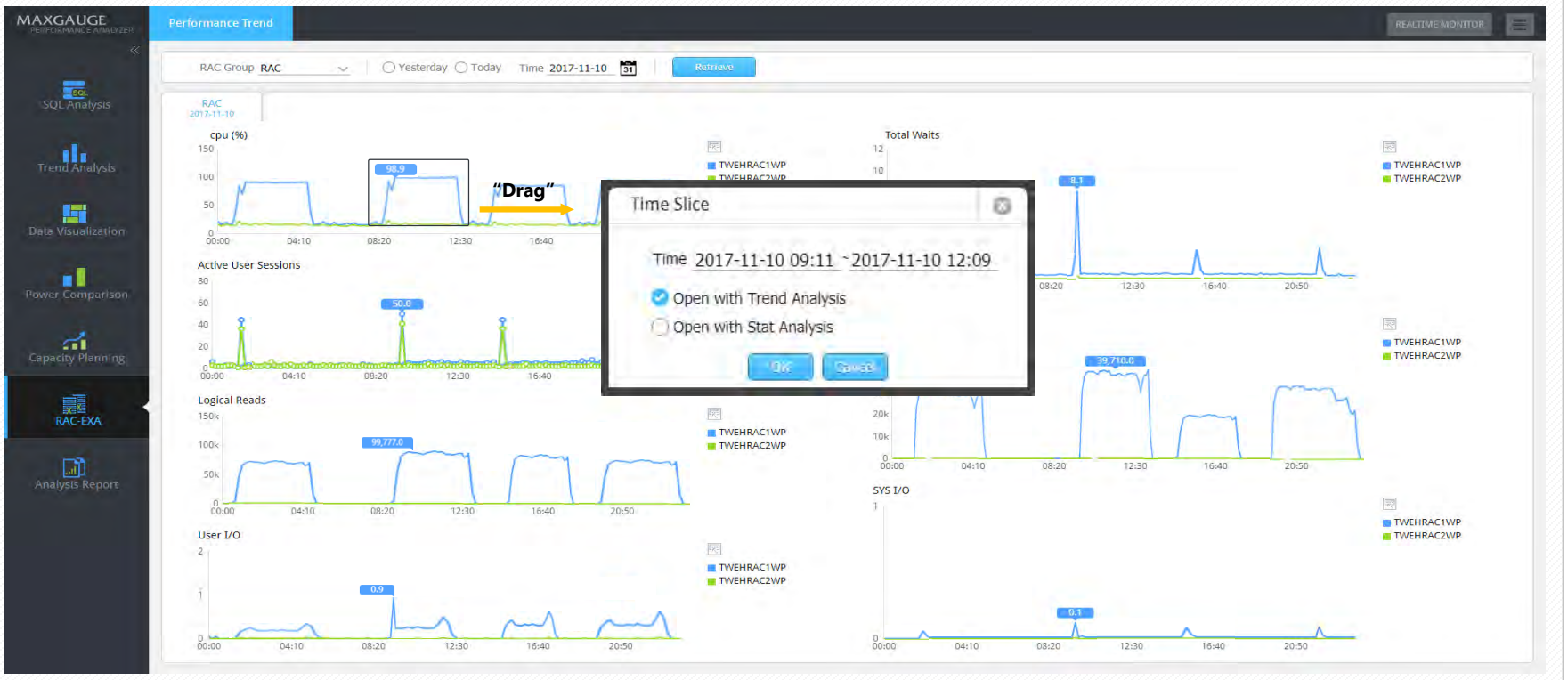
✓ 두 Node를 동시에 각기 다른 색으로 표기함으로써, Node 별 성능 추이를 빠르고 손쉽게 확인할 수 있습니다.

✓ **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 확인 하고 싶어요 "

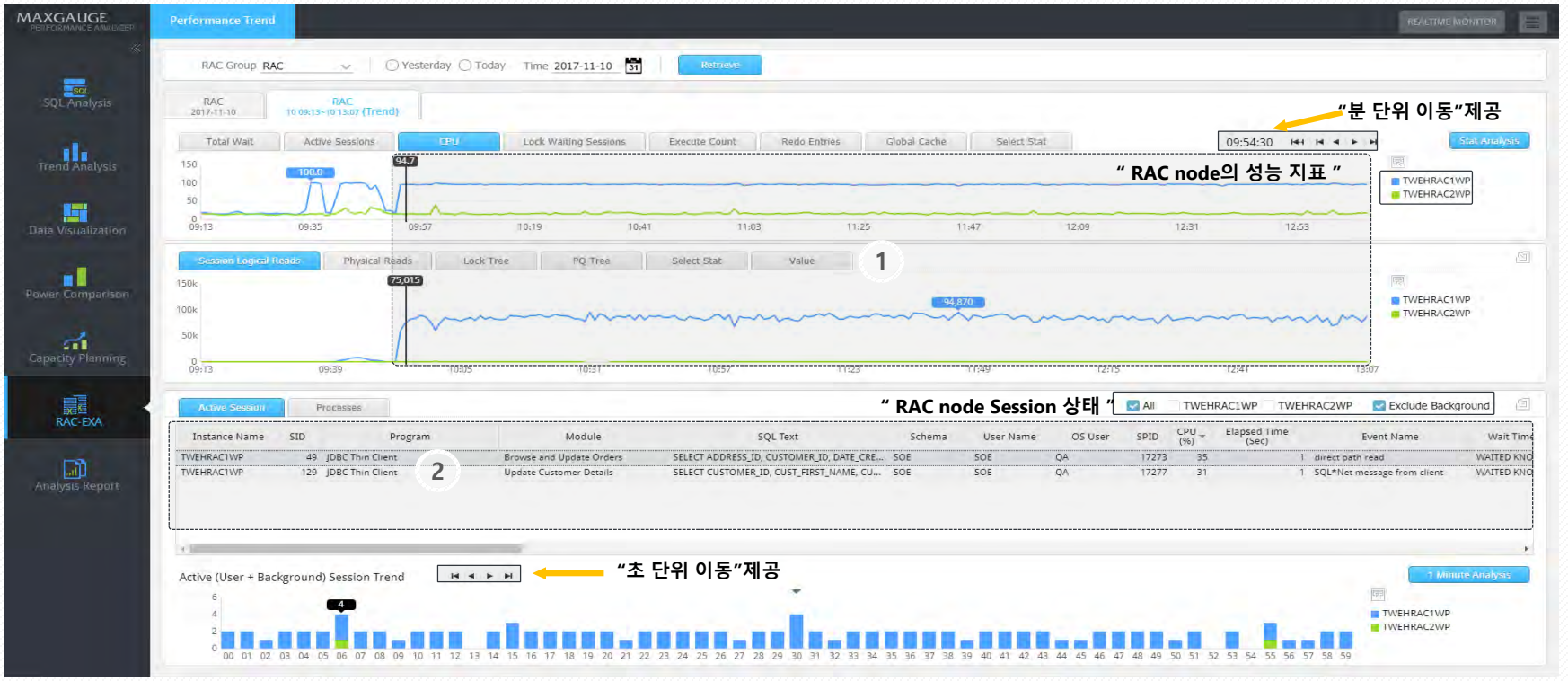
» RAC Performance Trend 조회 화면



✓ 부하가 되는 시점의 무슨 일이 벌어졌는지 좀더 자세히 확인하기 위해서 **"Drag"** 를 통해 Trend Analysis 연계 합니다.

" RAC 간의 Load Balance 및 성능을 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



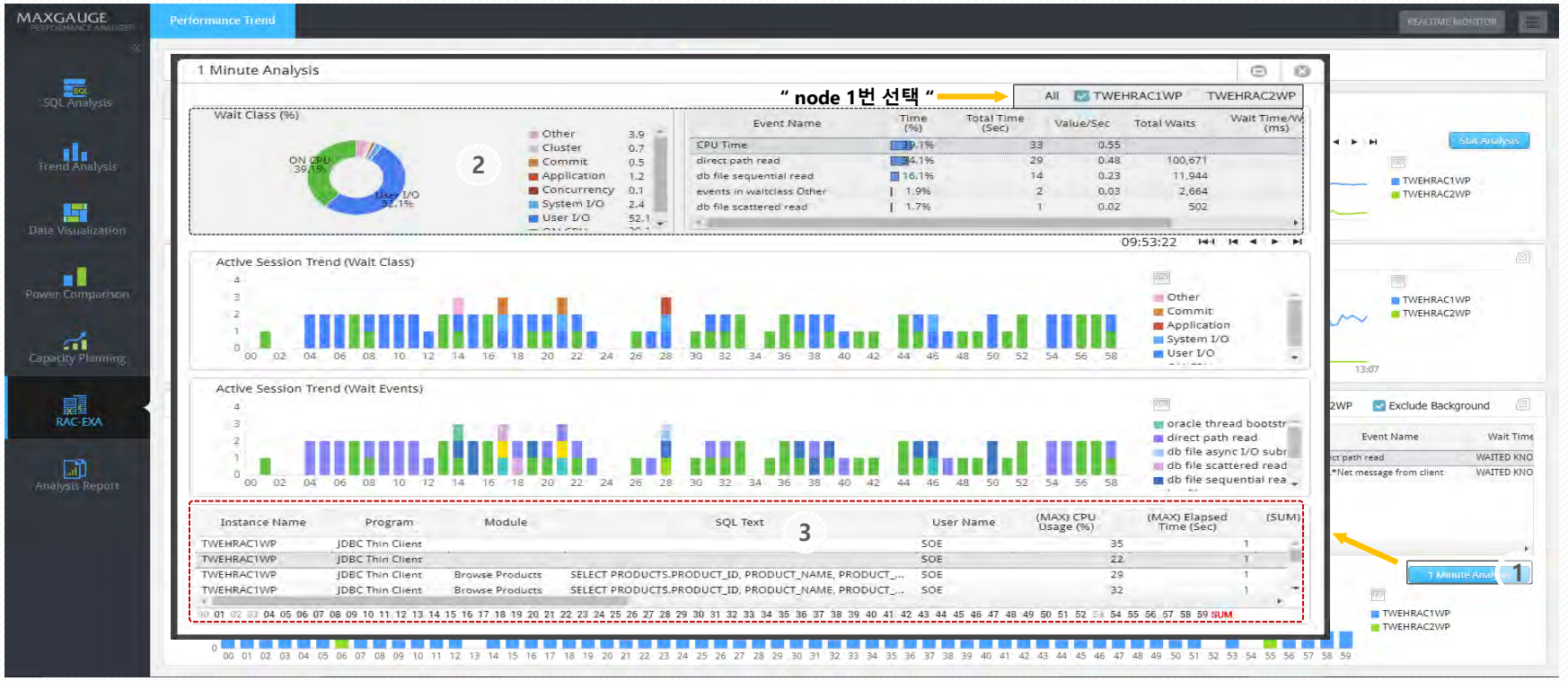
1 로그를 확인한 결과 항상 **1번 node만 CPU와 Logical Reads**가 급증 한 것을 확인 할 수 있습니다.

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 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



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

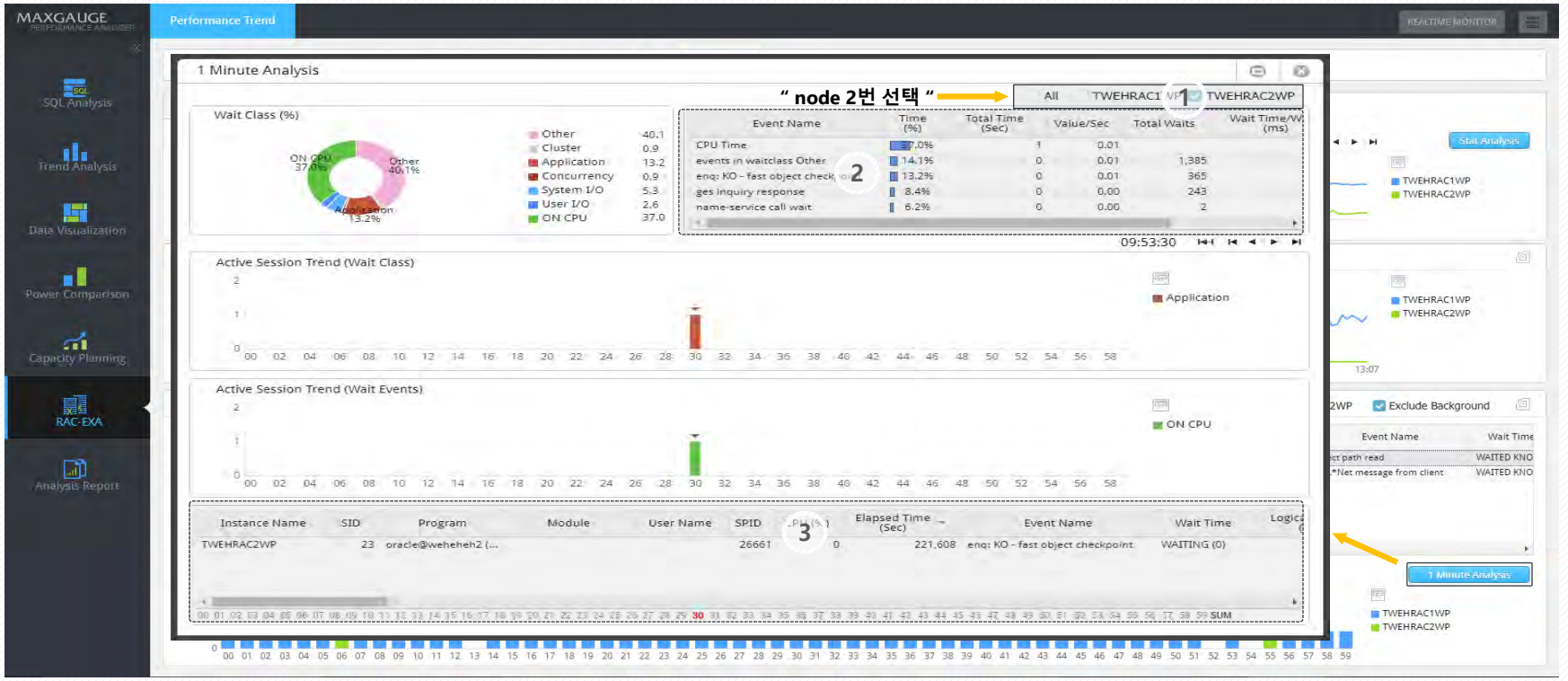
이 때 화면 하단의 **1minute Analysis 버튼**을 클릭하면 1분간 Stat / Wait에 대한 통계정보를 확인함으로써 빠르고 정확하게 분석할 수 있습니다.

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 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



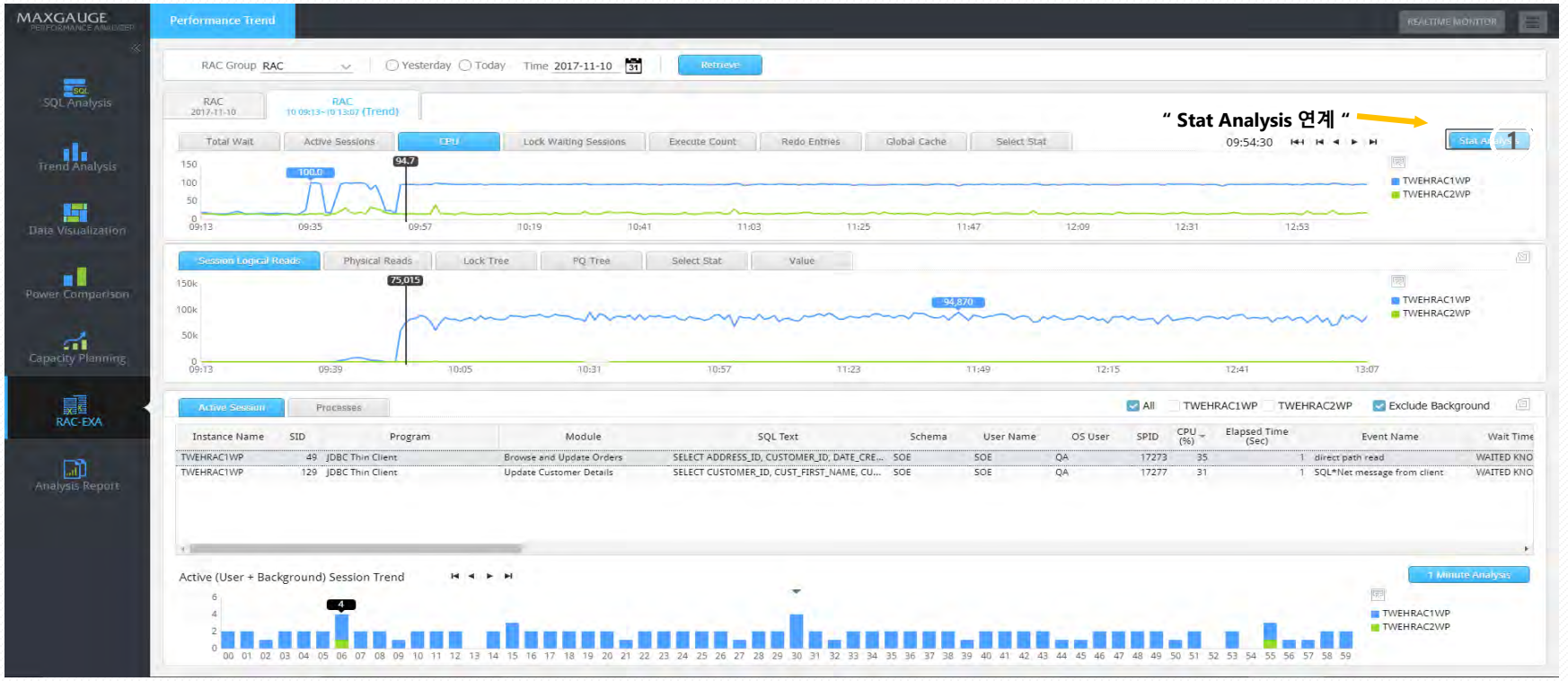
1 RAC 1min 분석의 **2번 node Session 정보**를 확인 합니다.

2 Event 확인 결과 동시간 때 **1번 node에 비해 Total Time이 수치**가 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 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



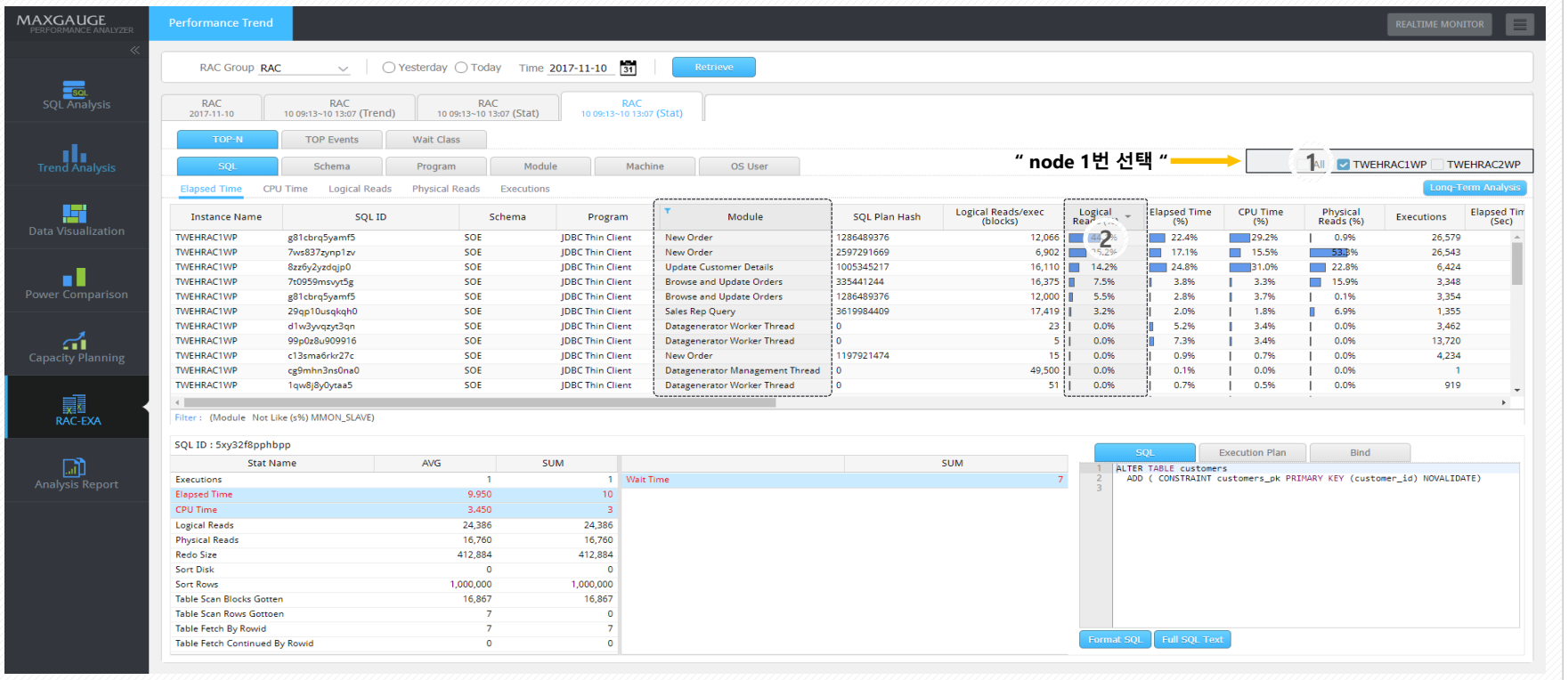
1 하지만 1분 구간만 확인할 경우, 전체에 걸쳐 발생한 것 마냥 오판하는 일반화 오류가 발생할 수도 있습니다.

이러한 우를 범하지 않기 위해서는 **Drag한 구간 전체에 대해 분석**하는 것입니다. 이는 화면 상단의 **Stat Analysis 버튼**을 클릭하여 분석이 가능합니다.

✓ **Stat Analysis**는 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 확인 하고 싶어요 "

RAC Performance Trend 조회 화면



1 Stat 분석의 1번 node를 선택 합니다.

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

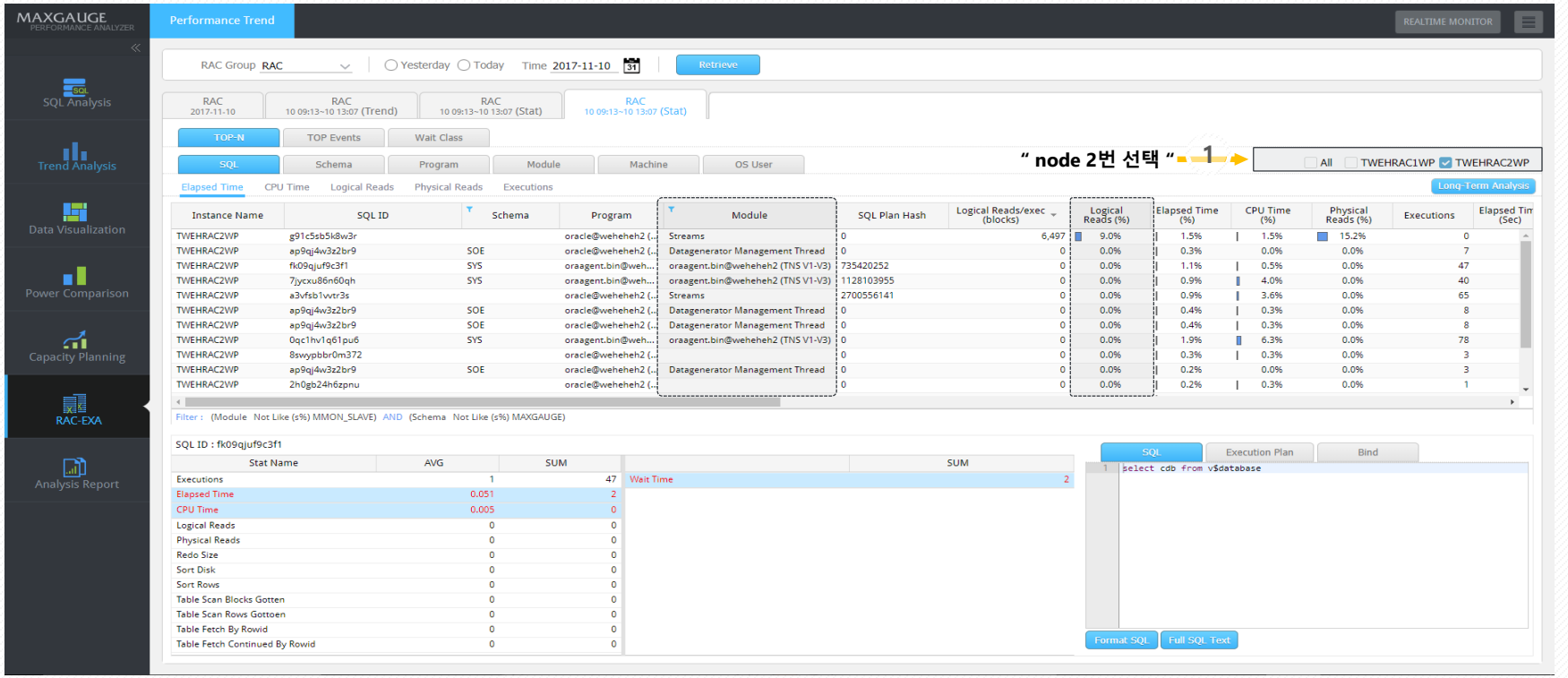
만약 Top 1, 2에 해당하는 **module명**이 이번에 추가된 업무 였다고 한다면, 예전과는 달리 성능 지표가 급격히 증가했다가 다시 내려가는 현상은

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 확인 하고 싶어요 "

» RAC Performance Trend 조회 화면



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

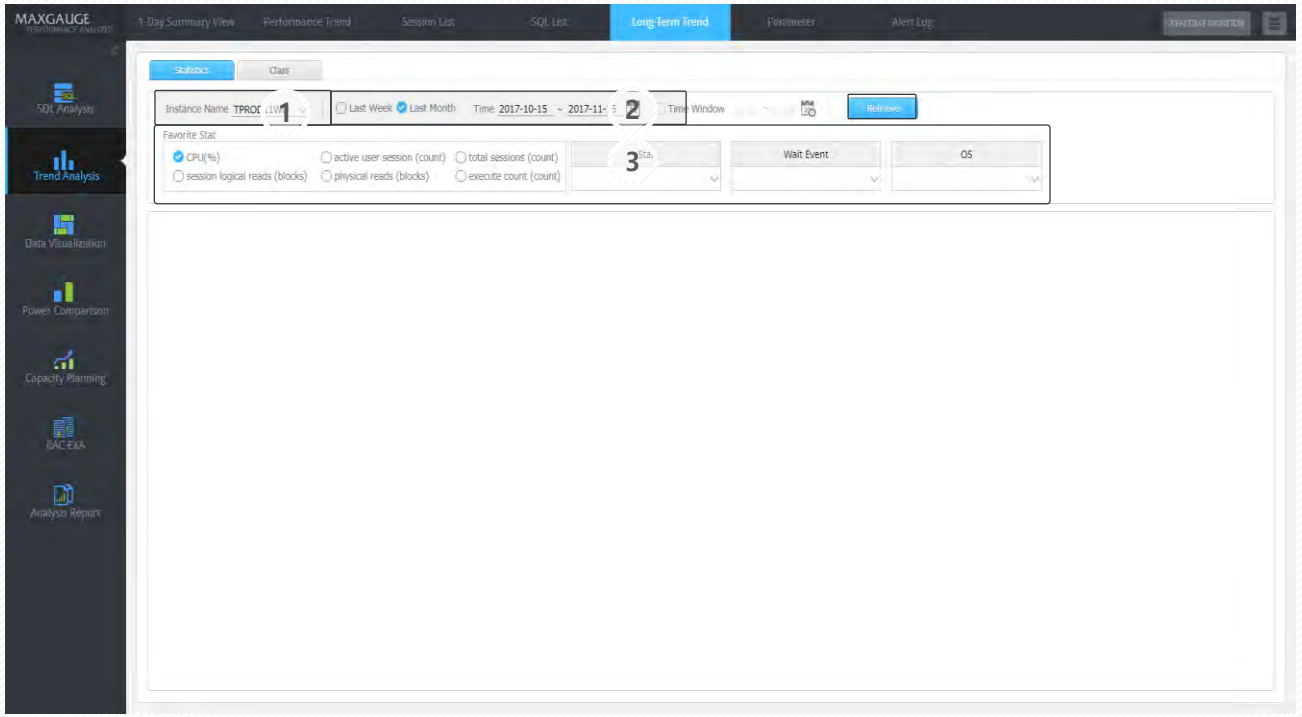
✓ 만약 신규 업무가 Node 1, Node 2에서 나누어서 실행되기를 원했던 경우라면, RAC 구성임에도 불구하고 1번 node로만 Session 접속이 이루어져 1번 Node에서만 부하가 발생 한 것입니다. 즉 RAC의 **" TNS 설정 정보 확인, 업무 구분을 통한 분산 작업 "** 에 대한 정밀한 검토가 필요할 것입니다.

실전 분석 사례 Case 3.

한 달간 가장 큰 부하의 원인을 알고 싶어요.

" 한 달간 가장 큰 부하의 원인을 알고 싶어요 "

Long-Term Trend Analysis 검색 화면



✓ 시나리오

INSTANCE : TROD11WO

분석 기간 : 2017.10.15~2017.11.15

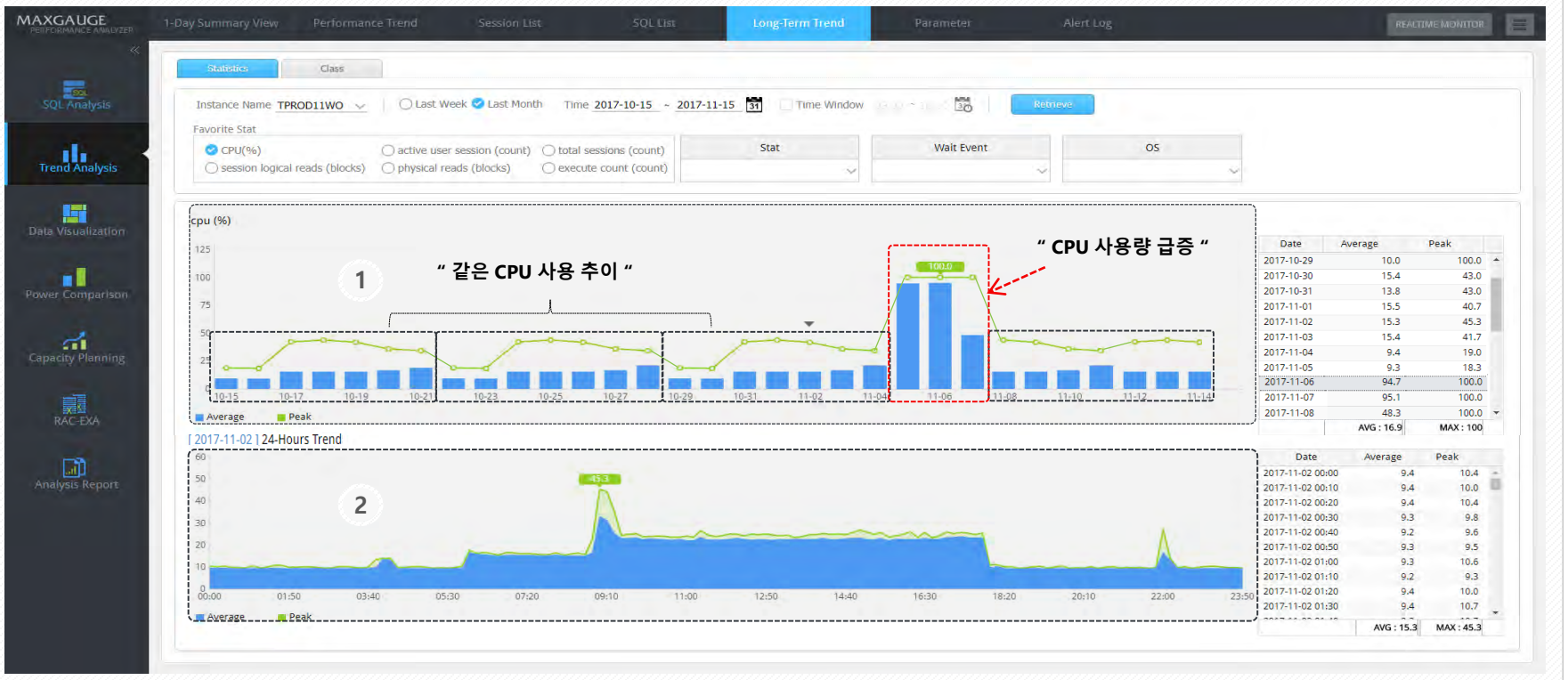
✓ 목표

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

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

" 한 달간 가장 큰 부하의 원인을 알고 싶어요 "

» Long-Term Trend Analysis 조회 화면



1 Long Term Analysis 결과를 분석해 보겠습니다. 화면 상단의 **성능 지표 그래프**를 통해 한 달간의 추이를 확인 할 수 있습니다. 확인 결과 일주일 동안 **9~15퍼센트**의 CPU 사용을 보이고 있습니다.

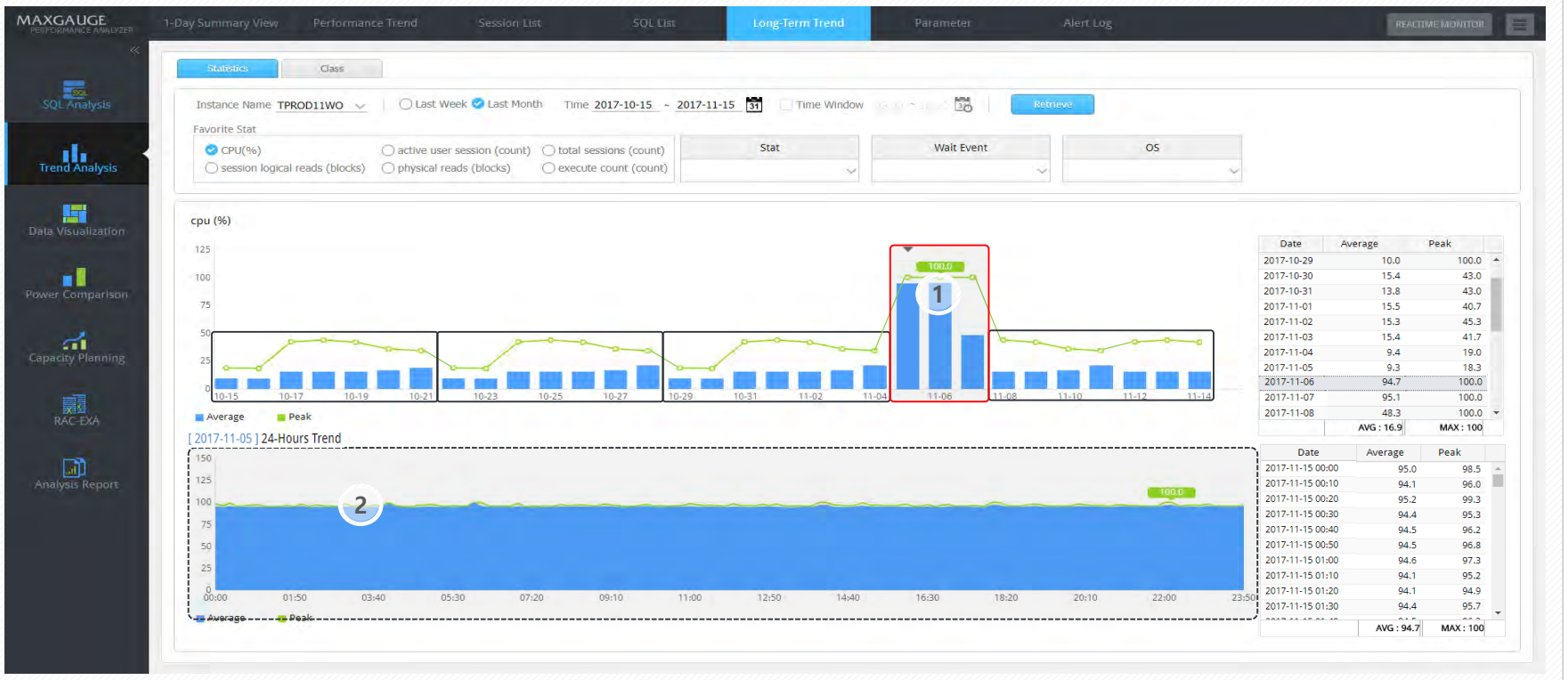
2 하단의 **24-Hours Trend 확인 시** 업무 시간에 CPU사용량 증가 후 줄어드는 일정한 형태를 보이고 있습니다. 그러나 **11월5일~11월7일** CPU 사용량이 급증한 것을 확인 할 수가 있습니다. 평소와 다른 부하가 있었음을 예상할 수 있습니다.

" 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

» Long-Term Trend Analysis 조회 화면



1 문제가 예상 되는 **11월 5일**의 막대 그래프를 선택해 보겠습니다.

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

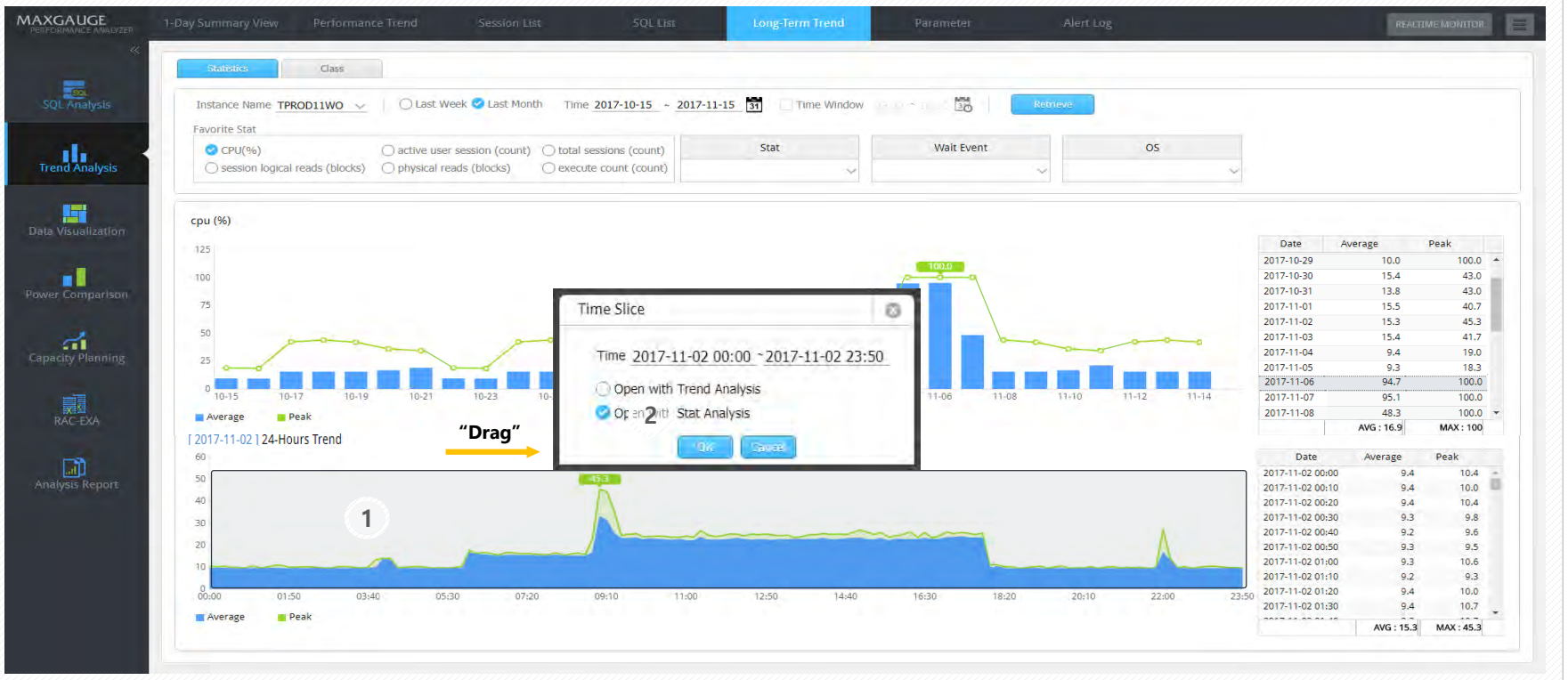
✓ 정확한 부하의 원인 파악을 위해 정상 시점과 부하시점의 **Stat Analysis**를 진행 해 보겠습니다.

" 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

» Long-Term Trend Analysis 조회 화면



1 **11월 2일**의 하루 동안의 Stat 정보를 확인 하기 위해 24-Hour-Trend를 **Drag** 합니다.

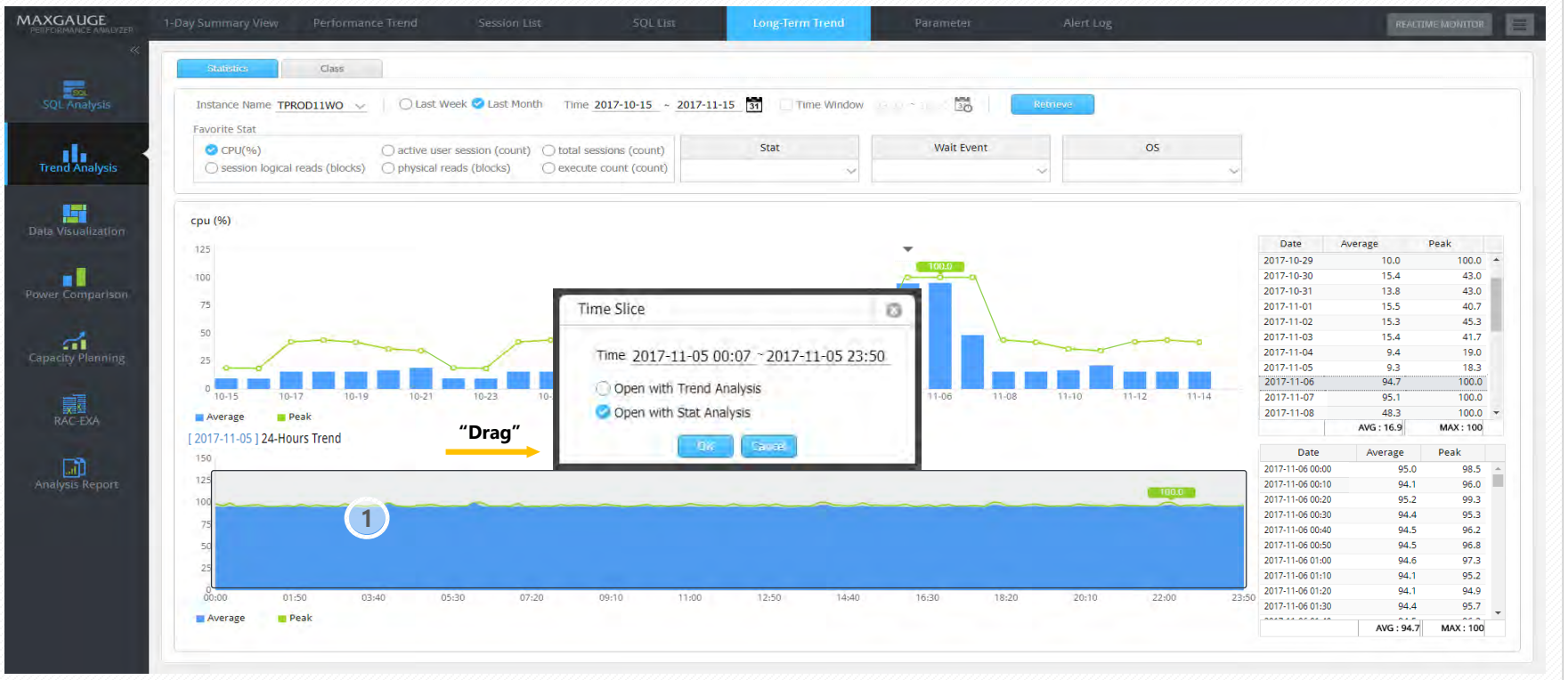
2 **Open with Stat Analysis**를 선택합니다.

" 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

» Long-Term Trend Analysis 조회 화면



1 **최초 부하 시점인 11월 5일**의 하루 동안의 Stat 정보를 확인 하기 위해 24-Hour-Trend를 **Drag** 합니다.

" 한 달간 가장 큰 부하의 원인을 알고 싶어요 "

» 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
5vaq3kqnwpsv5	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
12ajd4qcu7bba	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
7t0959msvyt5g	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
dvbv42b3hfyru	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.

Practical Guide

1-Day Summary View

Contents

1-Day Summary View?

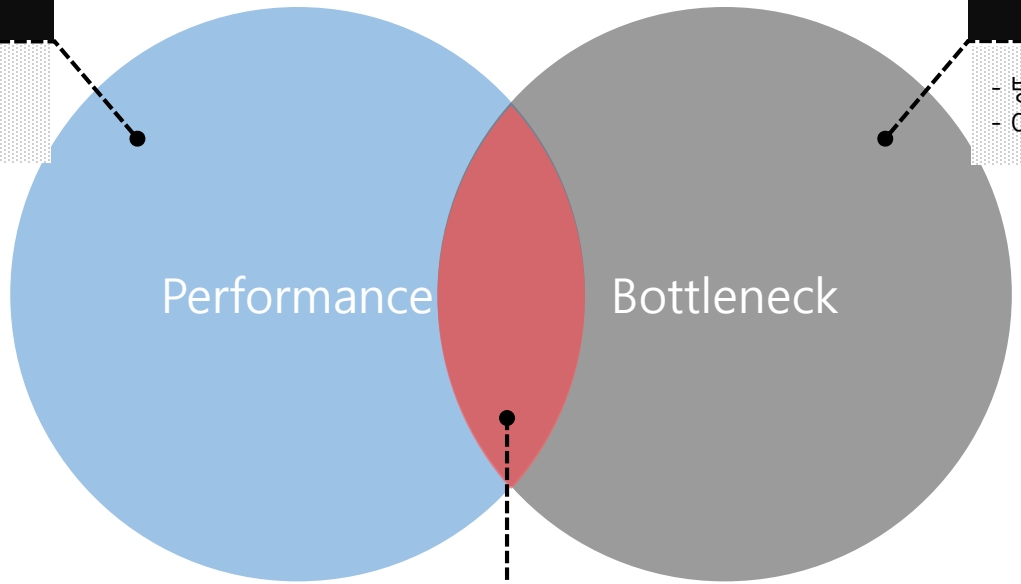
1-Day Summary View의 주요 기능은 무엇인가요?

주요 성능 지표 분석

- Peak 시간대 쉽게 파악
- Schema, Program, Module, SQL 별 Top-N 정보 제공

대기 클래스 분석

- 병목 시간대 쉽게 파악
- 이벤트 별 수행 SQL 정보 제공



1-Day Summary View

정확한 원인 분석 가능

1-Day Summary View은 언제 쓰나요?

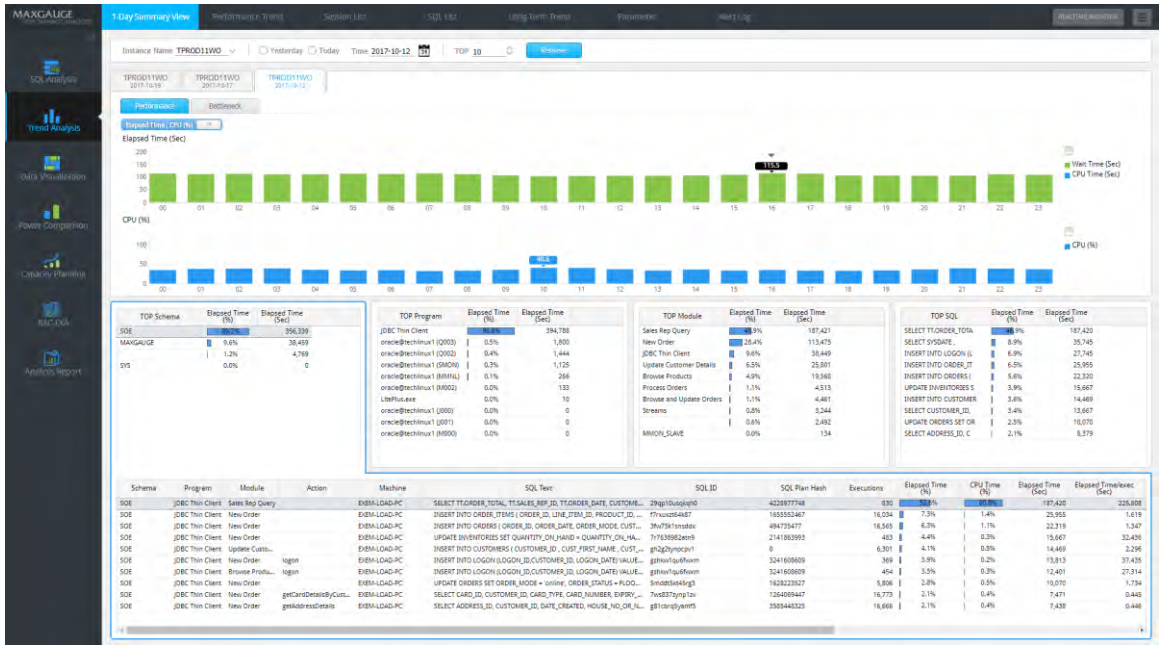
*** 1-Day Summary View**

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

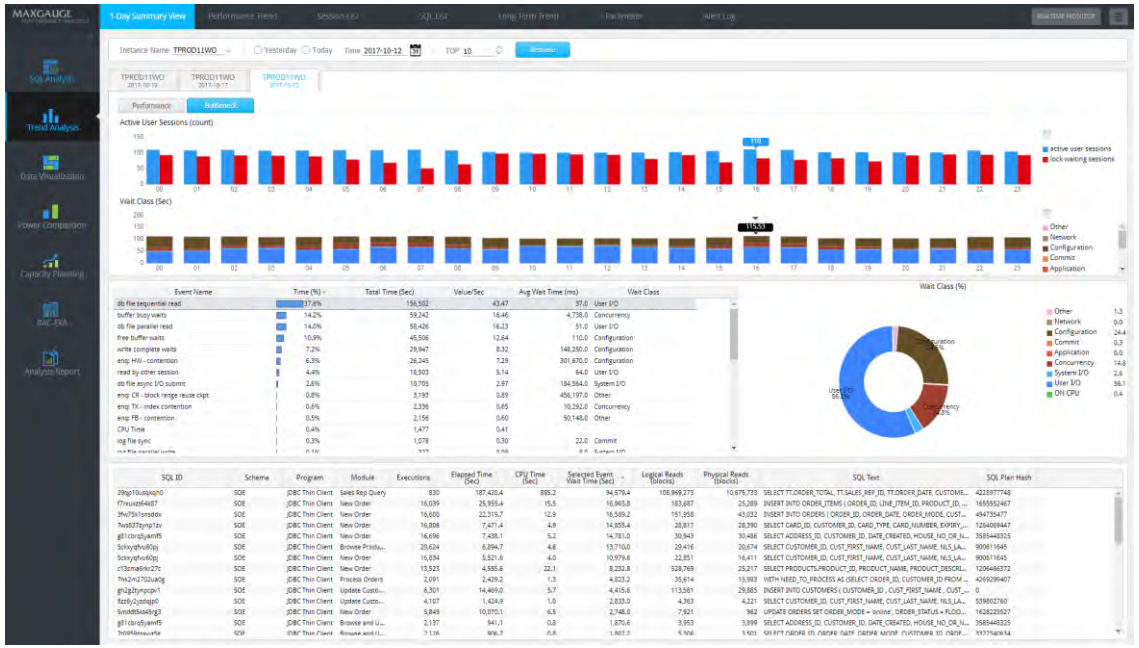
1-Day Summary View

- 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 SQL**을 **마우스 1-클릭**만으로 제공한다.
- Performance, Bottleneck의 두 가지 탭으로 구성되어 있다.
- **Performance Tab**은 주요 성능 지표에 대한 시간 별 성능 추이 및 **시간 대별 Top-N 정보**를 제공한다.
- **Bottleneck Tab**은 Wait Class에 대한 시간 별 성능 추이 및 대기 **이벤트 별 수행 SQL 정보**를 제공한다.
- 해당 기능은 Performance Analyzer ▸ **Trend Analysis** ▸ **1-Day Summary View** 경로를 통해 사용할 수 있다.

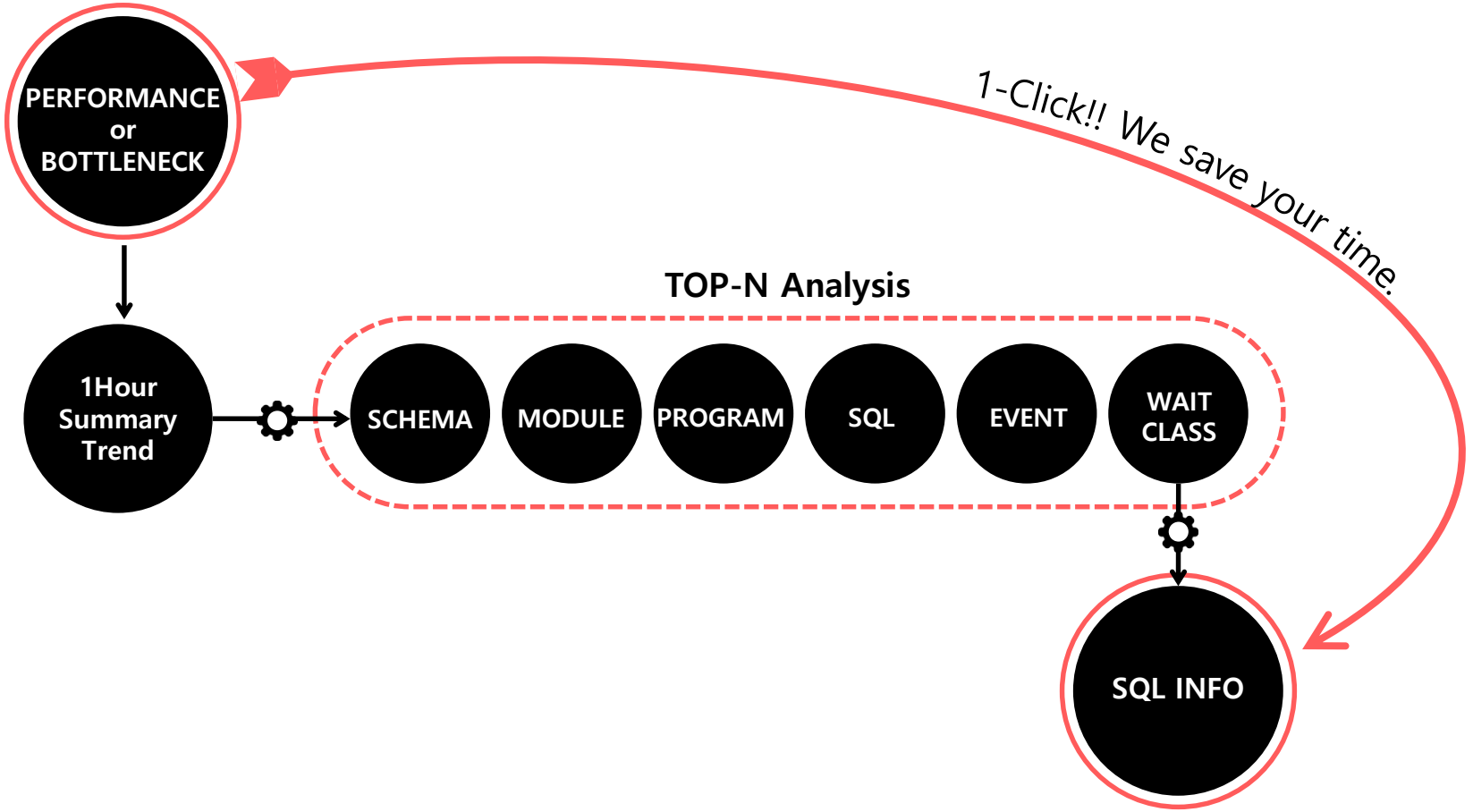
Performance



Bottleneck



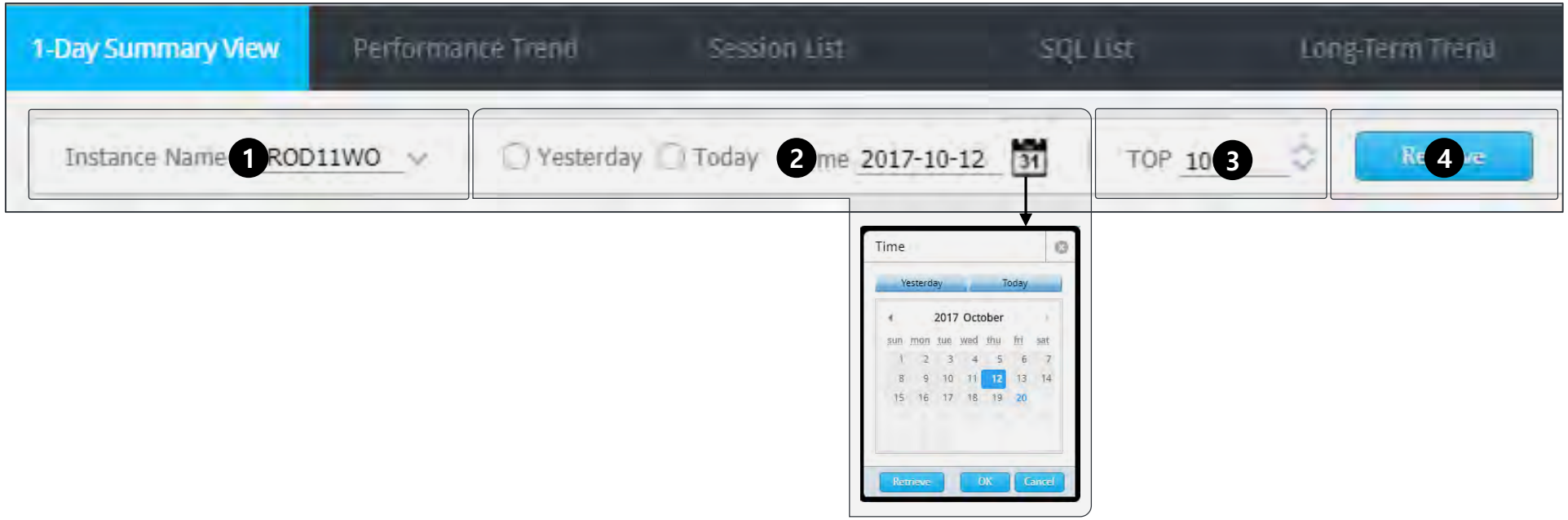
한 시간 단위 부하 SQL을 빠르고 쉽게 추적할 수 있습니다!



1-Day Summary View의 활용

1-Day Summary View의 사용 방법

» 1-Day Summary View 검색 화면



1 분석을 원하는 Instance를 선택합니다.

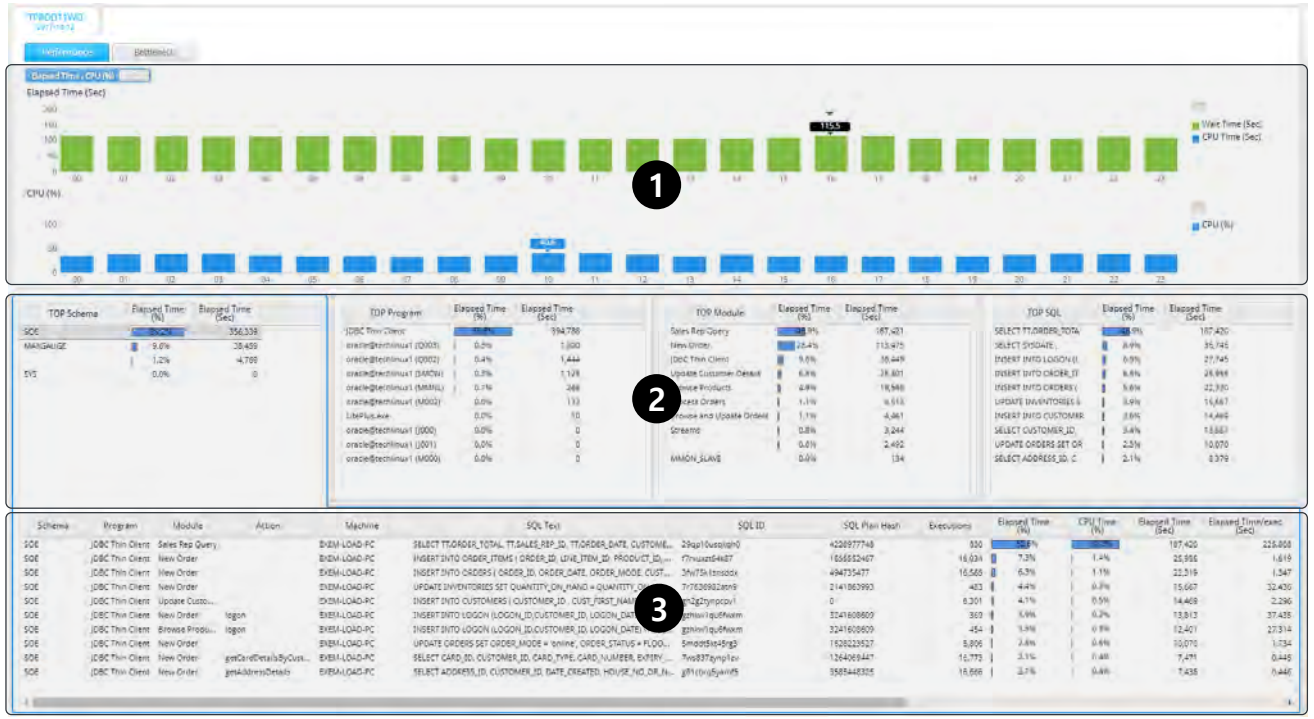
2 분석 일자를 Yesterday(기본 설정 값), Today를 선택하거나, 31 모양을 눌러서 원하는 날짜를 선택합니다.

3 TOP-N 구간에 출력되는 SQL 수를 조절합니다.

4 위의 설정한 조건으로 **1-Day Summary View를 실행**합니다.

1-Day Summary View의 사용 방법 (Performance 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, PYSICAL 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 사용 방법 (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.

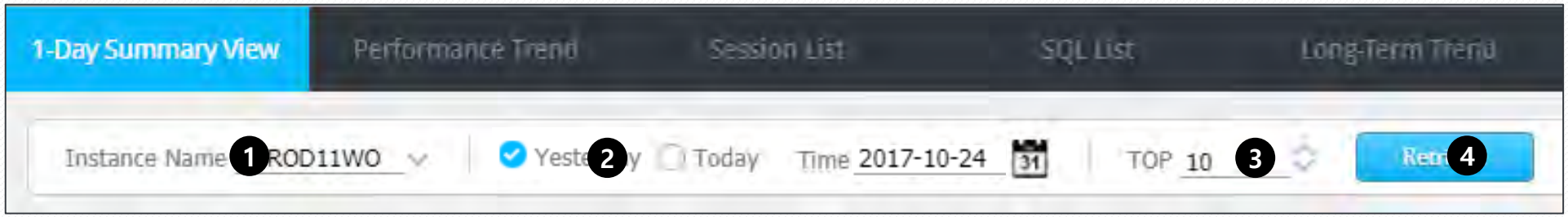
4 **SQL Info** : Top Event Area내에서 선택한 항목과 관련된 SQL 정보를 제공한다.

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)"

» 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. 캘린터 아이콘을 통해 특정 날짜 선택도 가능합니다.

3 출력되는 Top-N 의 개수**(TOP 10)**를 설정합니다.

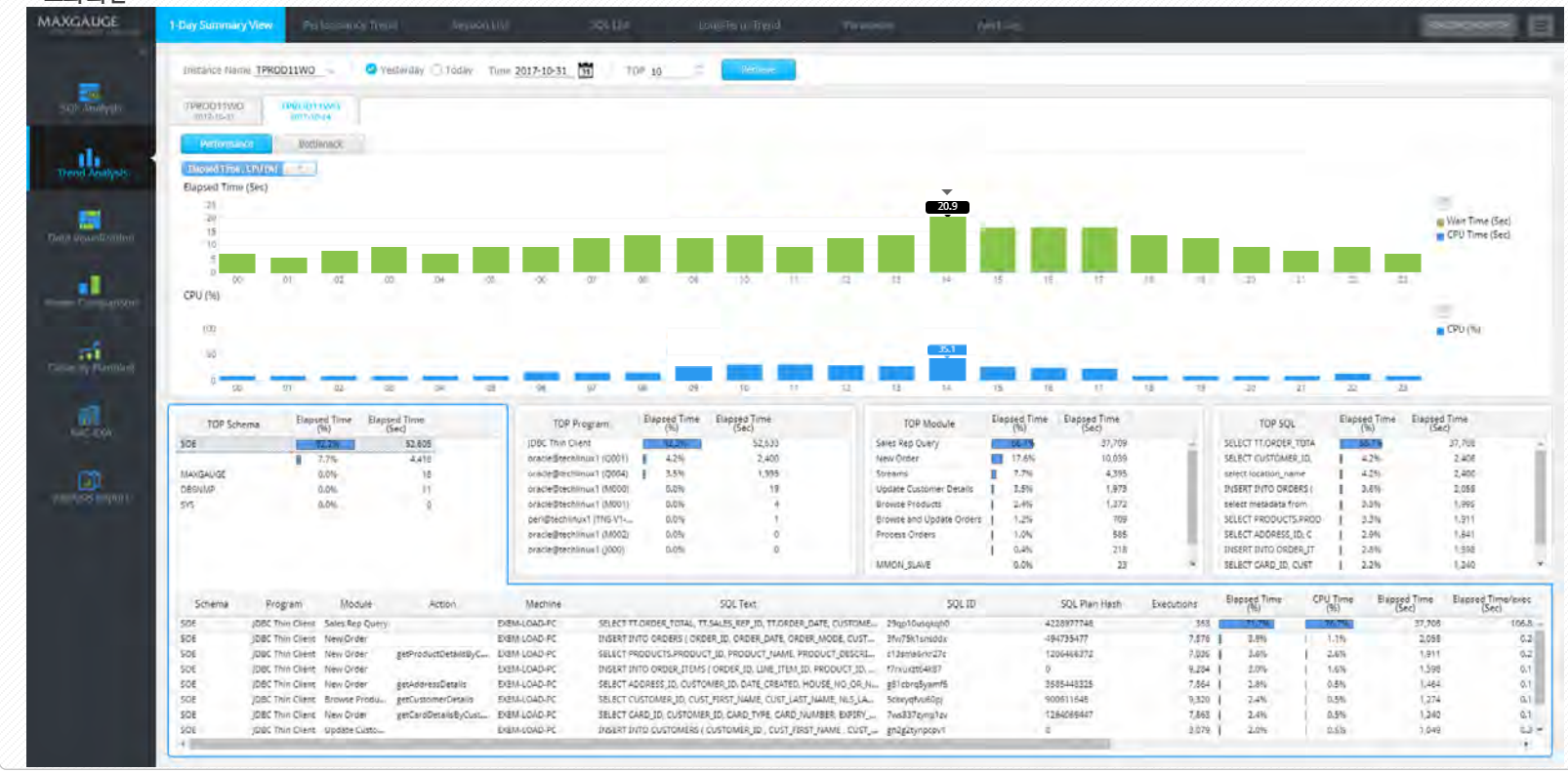
4 위의 설정한 조건으로 **1-Day Summary View 실행합니다.**

"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 (Performance) 조회 화면



✓ 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.

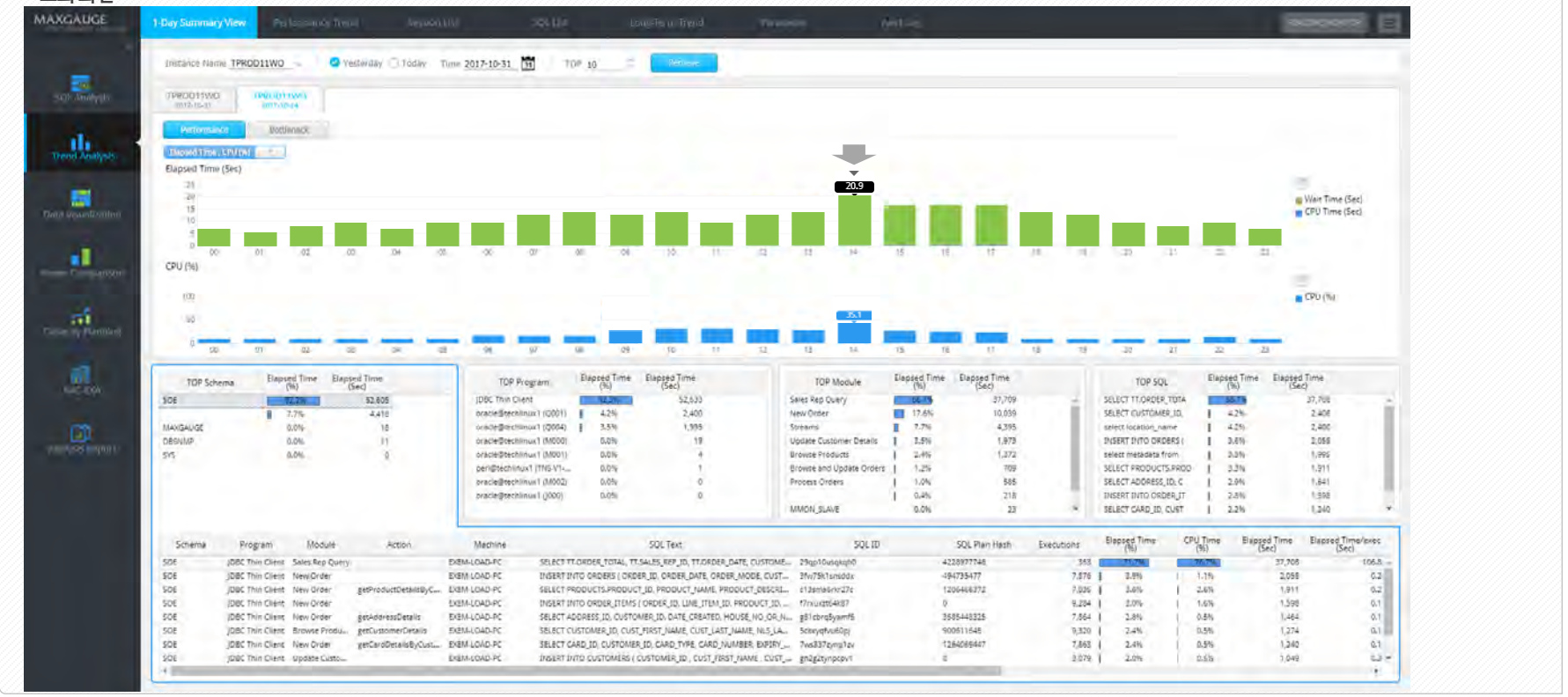
✓ 가장 하단 표를 통해 Top-N 데이터 별 수행 SQL 정보를 확인할 수 있습니다.

"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 (Performance) 조회 화면



✓ Elapsed Time 과 CPU 성능 지표의 시간 별 성능 추이를 확인힙니다.

✓ 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.

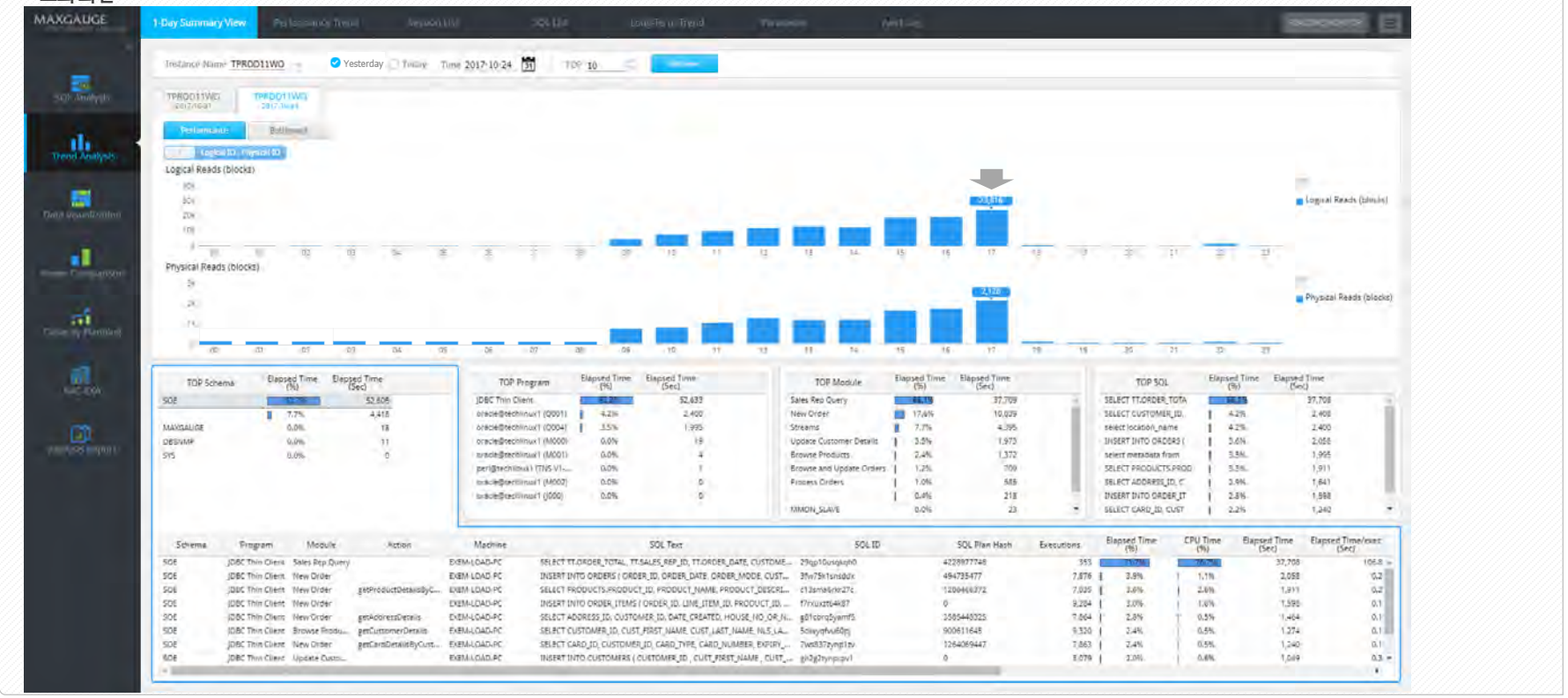
✓ Wait Time 20.8(Sec), CPU Time 0.1(Sec) 로 Elapsed Time 지표에 대한 Peak 시간대는 14:00~14:59 임을 확인할 수 있습니다.

"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 (Performance) 조회 화면



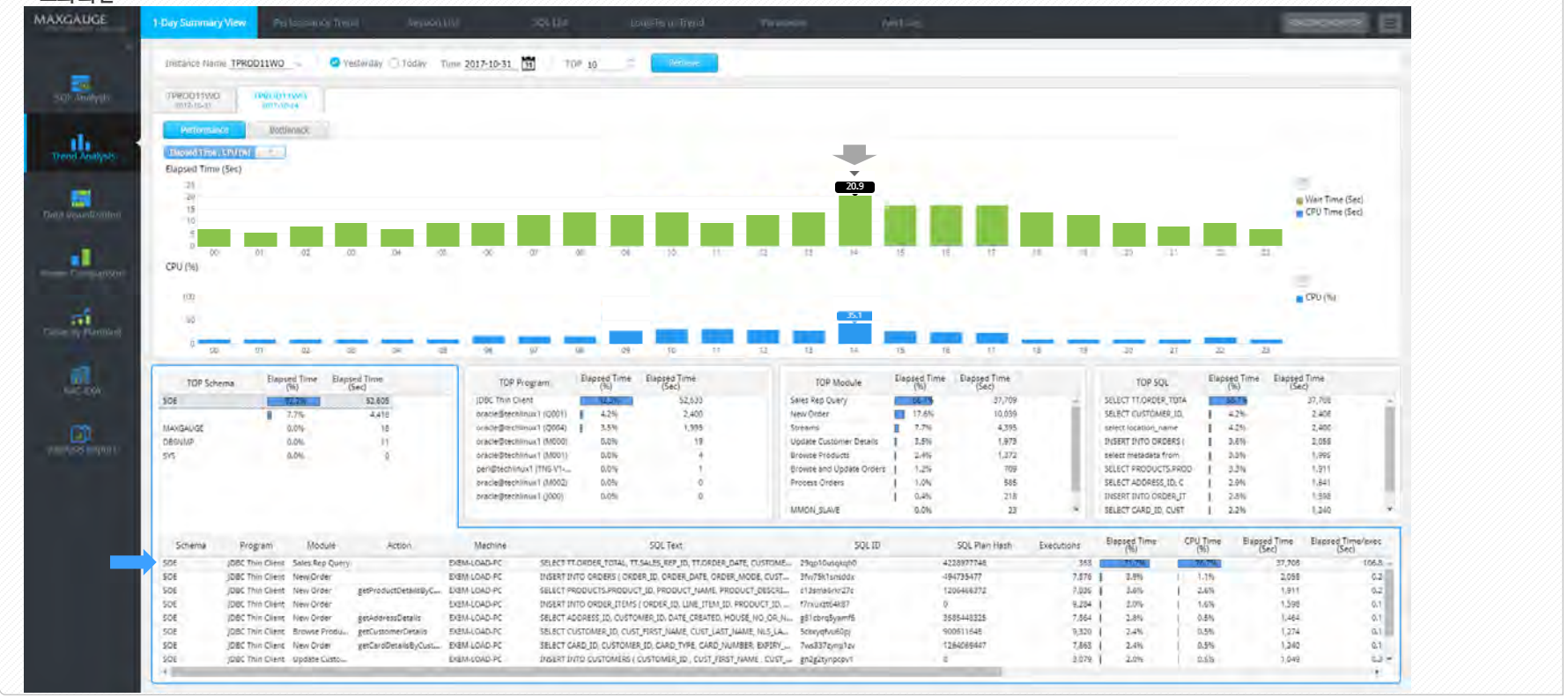
- ✓ Toggle Button 을 클릭하여 Elapsed Time, CPU 성능 지표에서 Logical IO, Physical IO 성능 지표로 변경 조회 가능합니다.
- ✓ Logical IO 와 Physical IO 의 성능 지표에 대한 시간 별 성능 추이를 확인합니다.
- ✓ Logical/Physical IO 모두 오전 9시부터 오후 6시 이전까지 계속해서 수치가 증가하는 것을 확인할 수 있습니다.
- ✓ Logical IO 에 대해 23,816(blocks) 로 Peak 시간대는 17:00~17:59 임을 확인할 수 있습니다.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

"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 (Performance) 조회 화면



✓ Elapsed Time 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 Elapsed Time 의 절반 이상의 비율인 66.1(%) 차지하는 것을 확인할 수 있습니다.

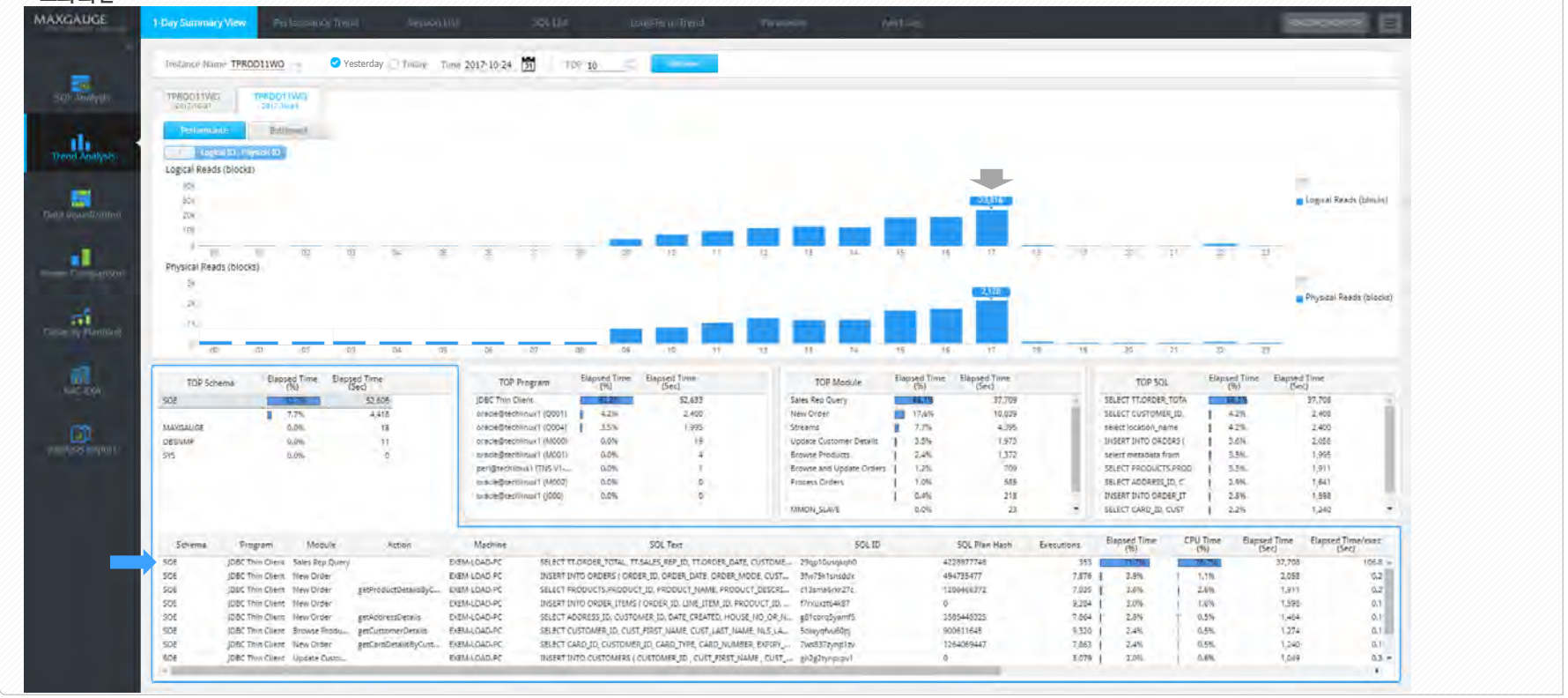
✓ 이 SQL 은 Top-N 에 해당하는 Schema, Program, Module 에 해당하는 SOE, JDBC Thin Client, Sale Rep Query 이며, "SQL ID : 29qp10usqkqh0" 입니다.

"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 (Performance) 조회 화면



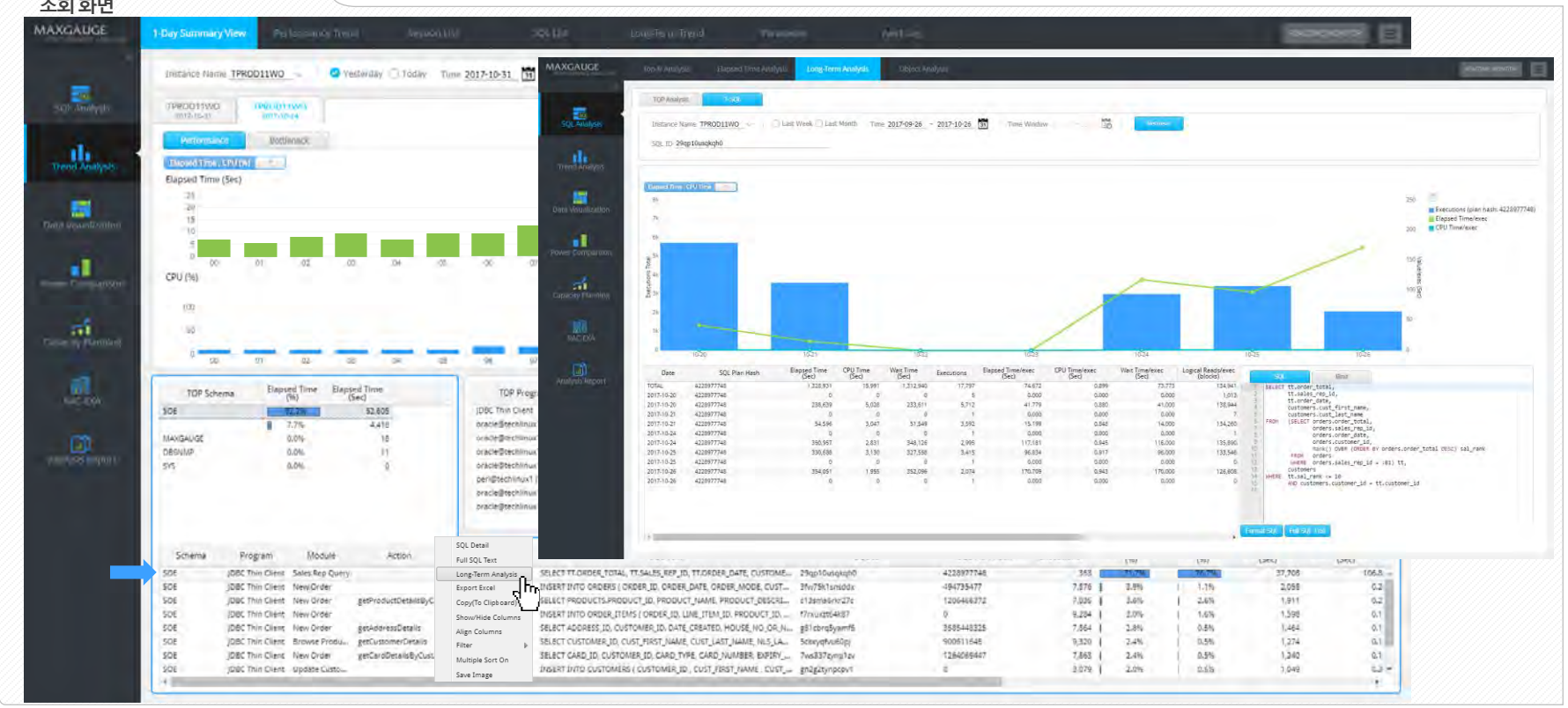
✓ Logical IO 와 Physical IO 에 대한 Peak 시간대의 Top Schema, Program, Module, SQL 은 각각 SOE, JDBC Thin Client, Sales Rep Query, SELECT TT.ORDER_TOTA~ 임을 확인할 수 있습니다.

✓ 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 View (Performance) 조회 화면



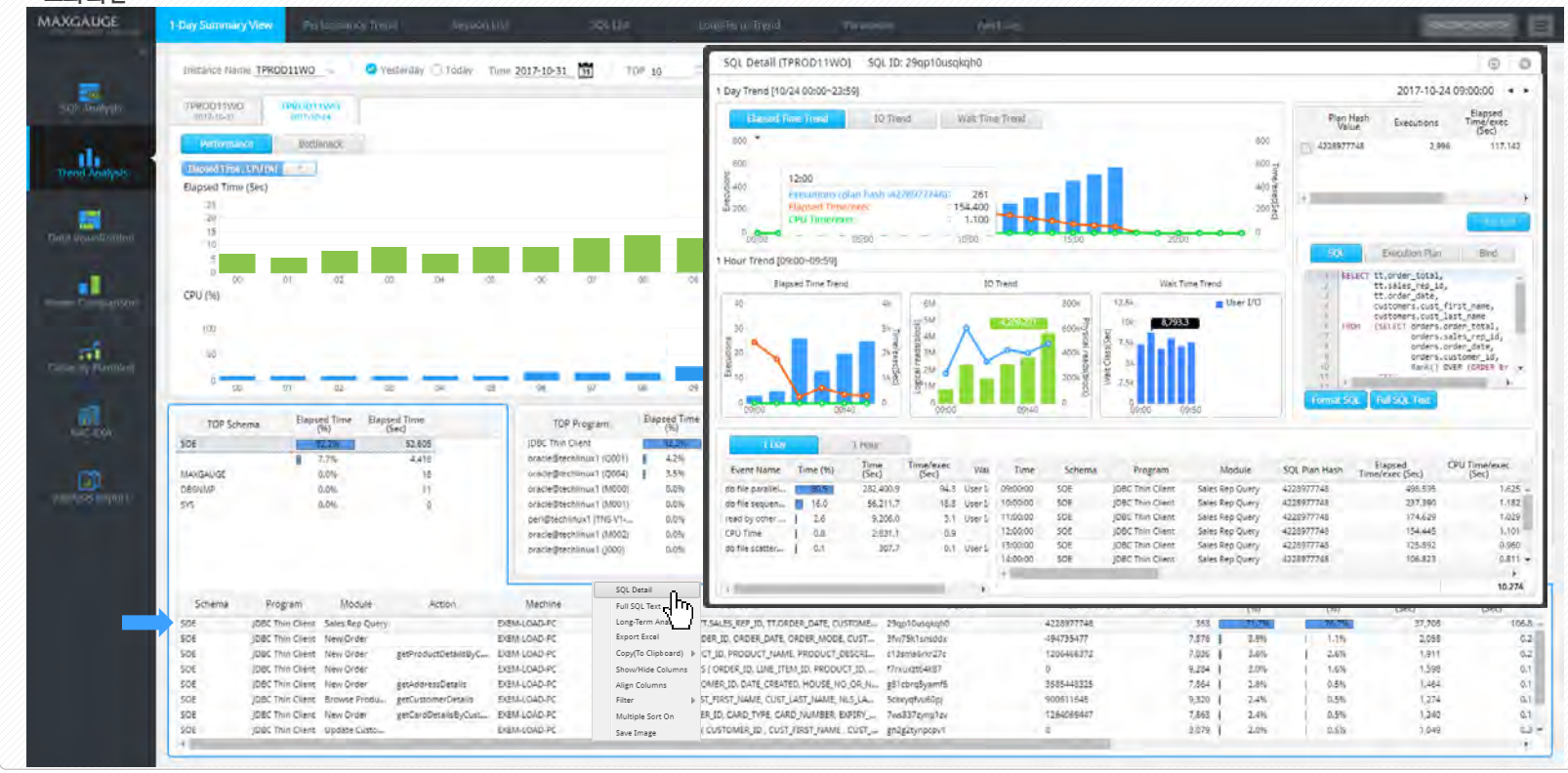
✓ 해당 SQL 을 그리드에서 선택 후 우클릭을 통해 Long-Term Analysis 를 선택하여 "SQL ID : 29qp10usqkqh0" 의 지난 일주일 동안의 일별 성능 추이를 확인할 수 있습니다.

✓ 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.

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

» 1-Day Summary View (Performance) 조회 화면



✓ "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.

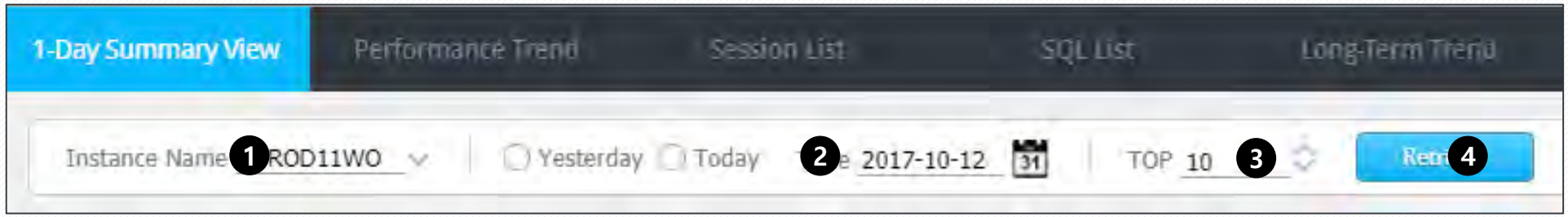
✓ SQL 수행 시 db file parallel read 대기 이벤트를 포함하여 User I/O 가 많이 발생하는 것도 확인할 수 있습니다.

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)"

1-Day Summary View (Bottleneck) 조회 화면



✓ **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 **2017-10-12** 를 선택합니다. 캘린더 아이콘을 통해 특정 날짜 선택도 가능합니다.

3 출력되는 Top-N 의 개수**(TOP 10)**를 설정합니다.

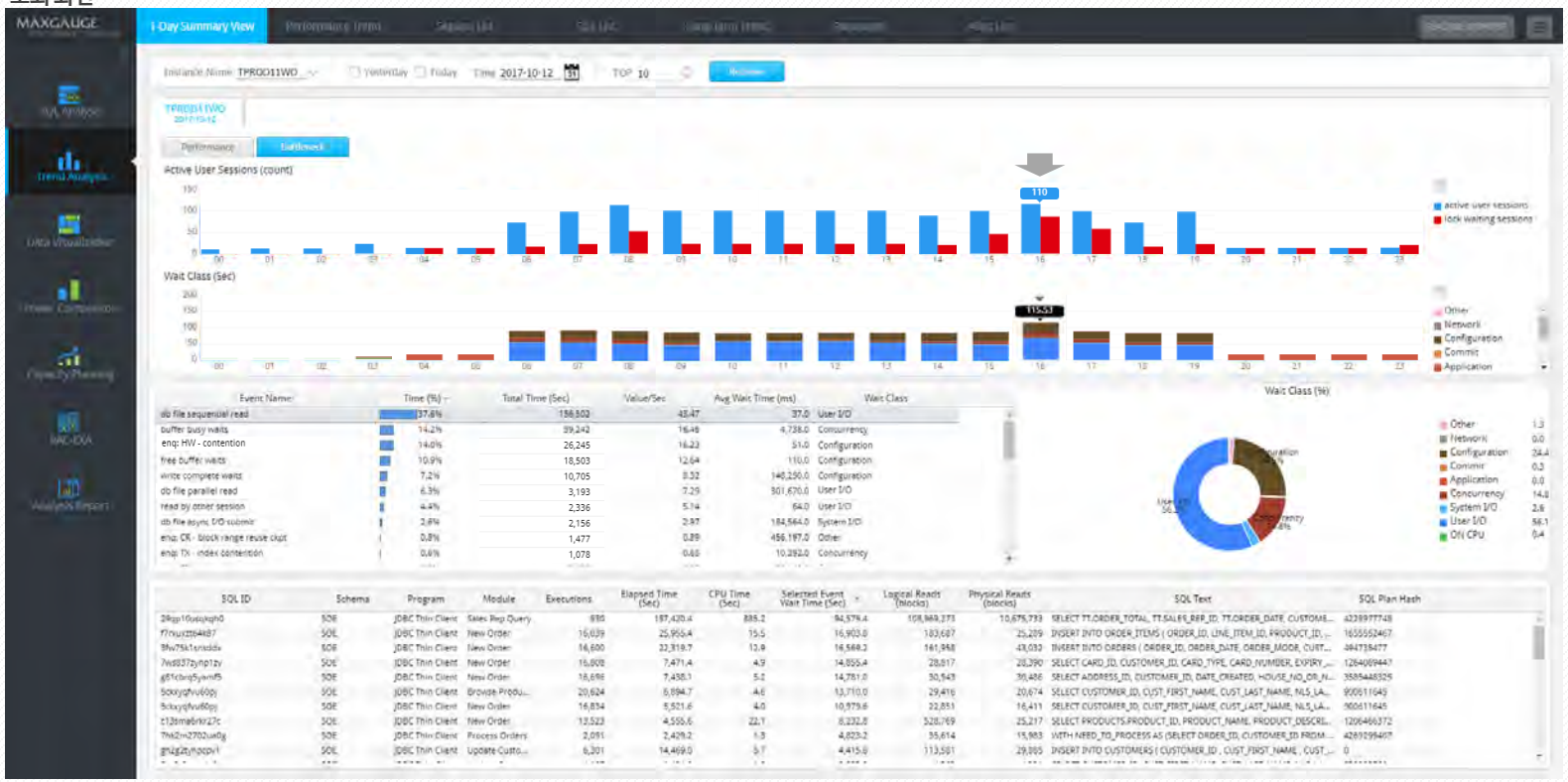
4 위의 설정한 조건으로 **1-Day Summary View 실행합니다.**

"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



✓ 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.

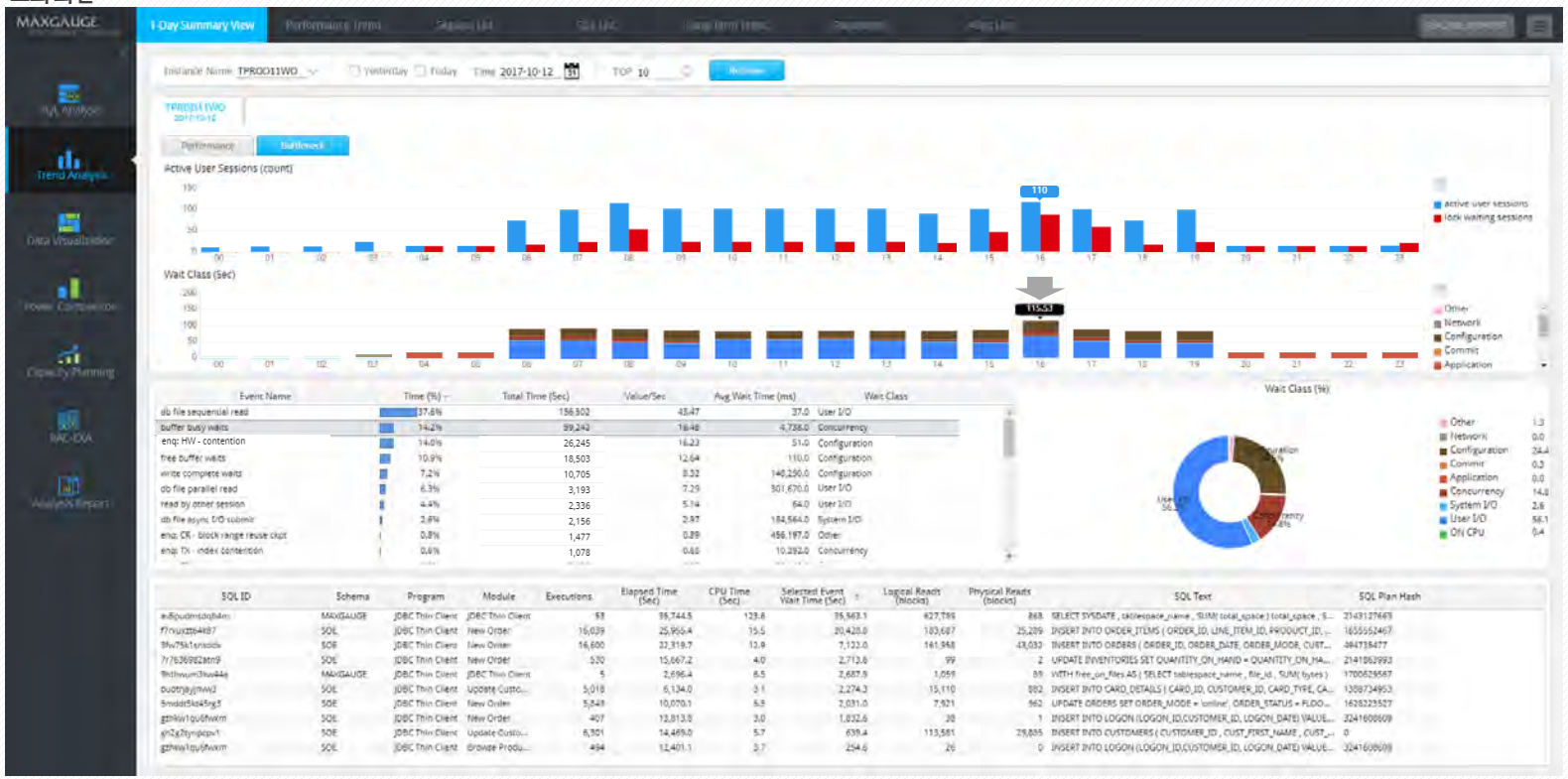
✓ Top Event 목록에서 이벤트 선택 시 해당 이벤트를 대기한 SQL 목록을 하단 SQL Info 영역에서 확인할 수 있습니다.

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)"

» 1-Day Summary View (Bottleneck) 조회 화면



✓ 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.

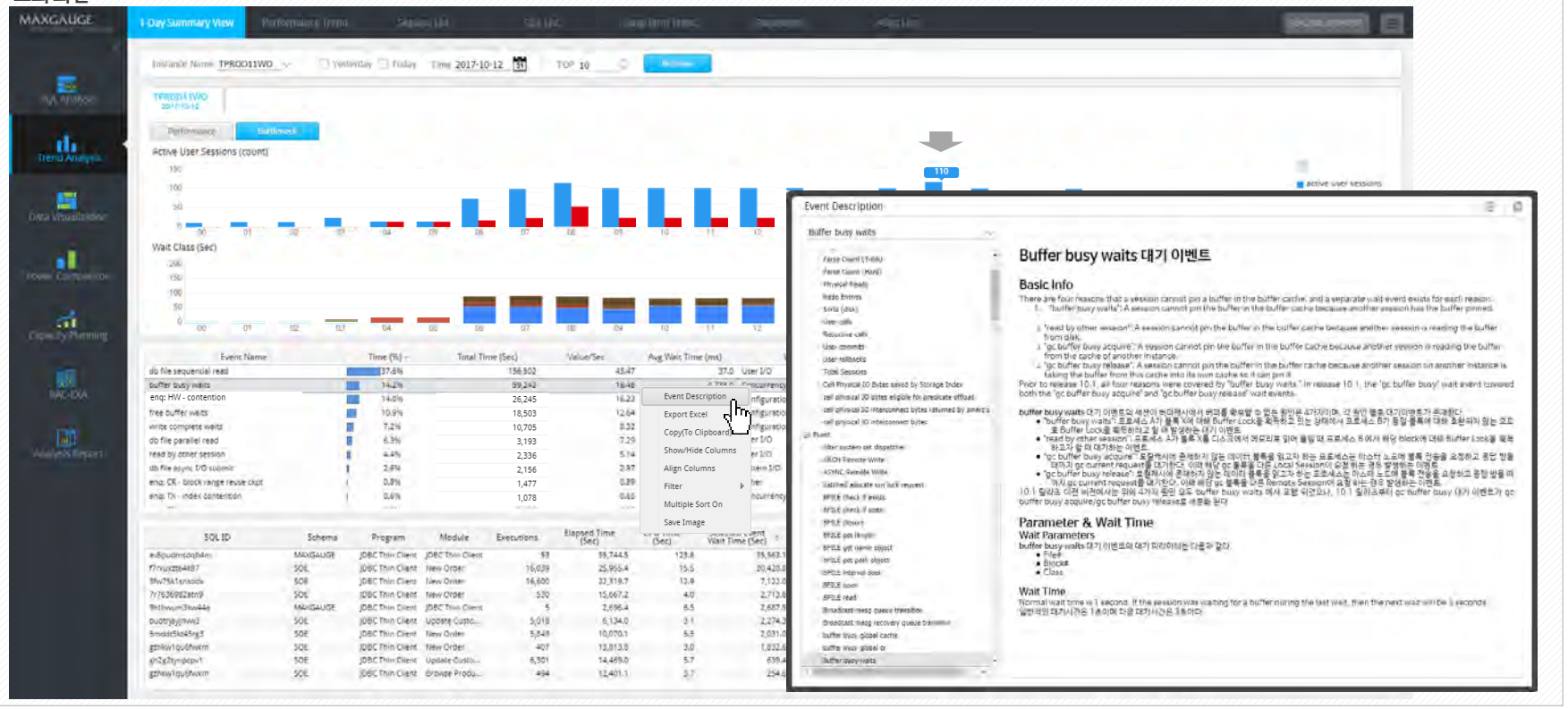
✓ 16:00~16:59 시간대에 Active User Sessions 은 110개, Lock Waiting Sessions 은 82 개로 락 세션 수가 급격히 늘어났으며 Wait Class 도 115.53 (Sec) 로 최대값으로 Peak 시간대임을 확인할 수 있습니다.

"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



✓ 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.

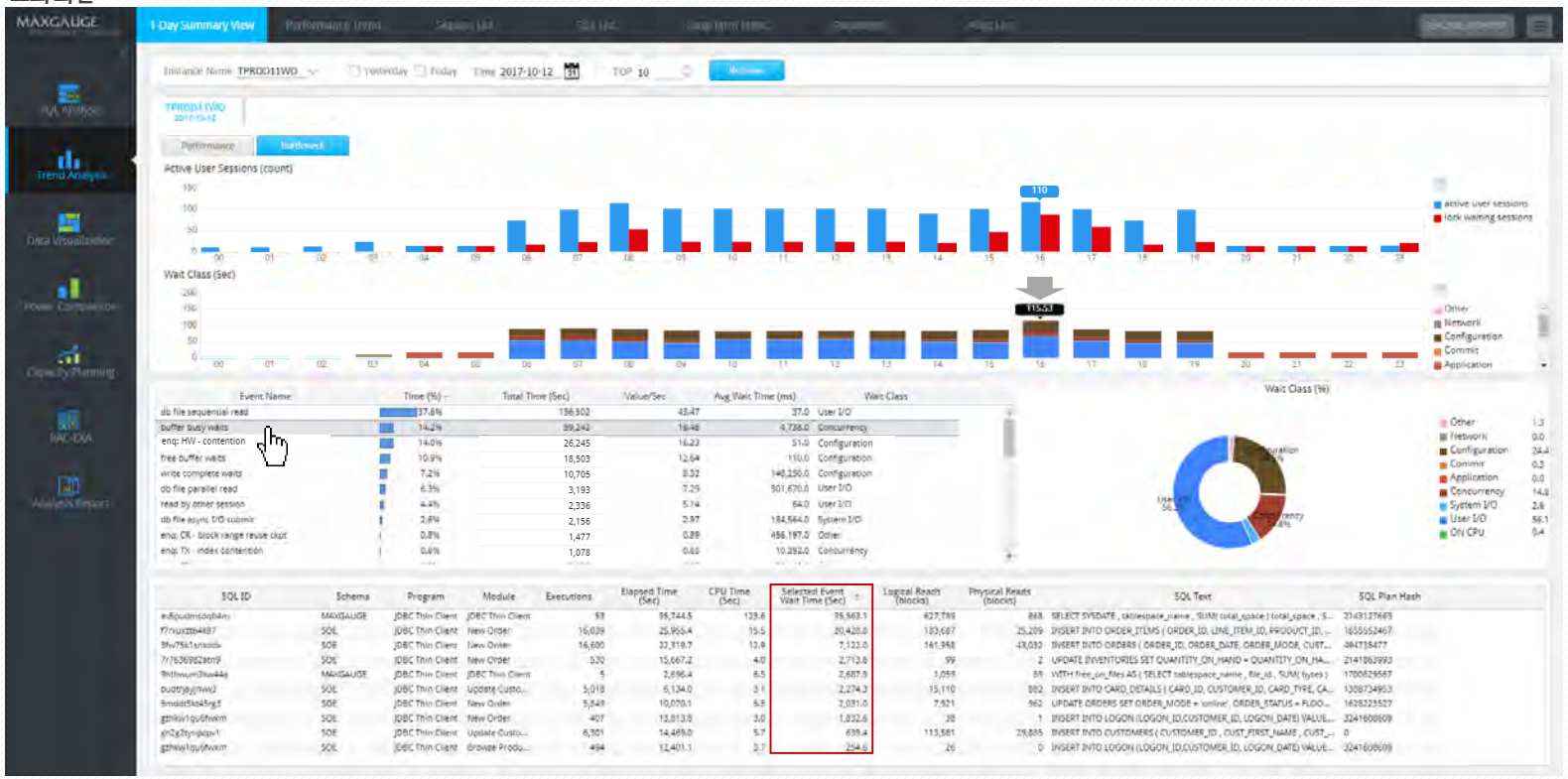
✓ Top Event 에서 우클릭 연계 기능(Event Description) 으로 해당 대기 이벤트에 대한 설명을 확인할 수 있습니다.

"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



✓ Top Event 중 하나를 선택하여 해당 이벤트를 대기한 SQL 목록을 확인할 수 있습니다.

✓ buffer busy waits 이벤트을 선택하여 이 이벤트를 대기한 SQL 의 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.

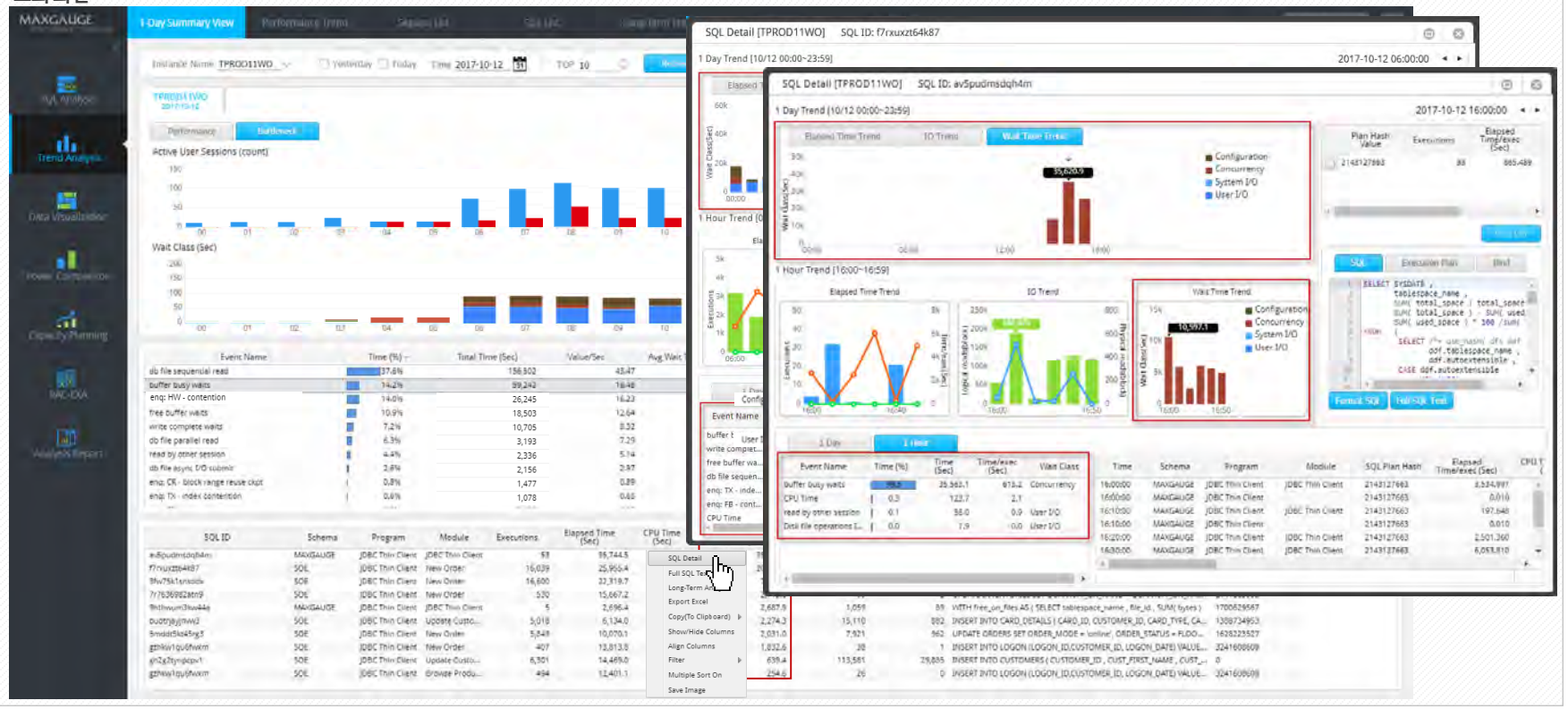
✓ Selected Event Wait Time (Sec) 수치가 높은 "SQL ID : av5pudmsdqh4m, f7rxuxzt64k87" 에 해당하는 2개의 SQL 이 특히 buffer busy wait 이벤트를 대기한 시간이 긴 것을 확인할 수 있습니다.

"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



✓ 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 확인할 수 있습니다.

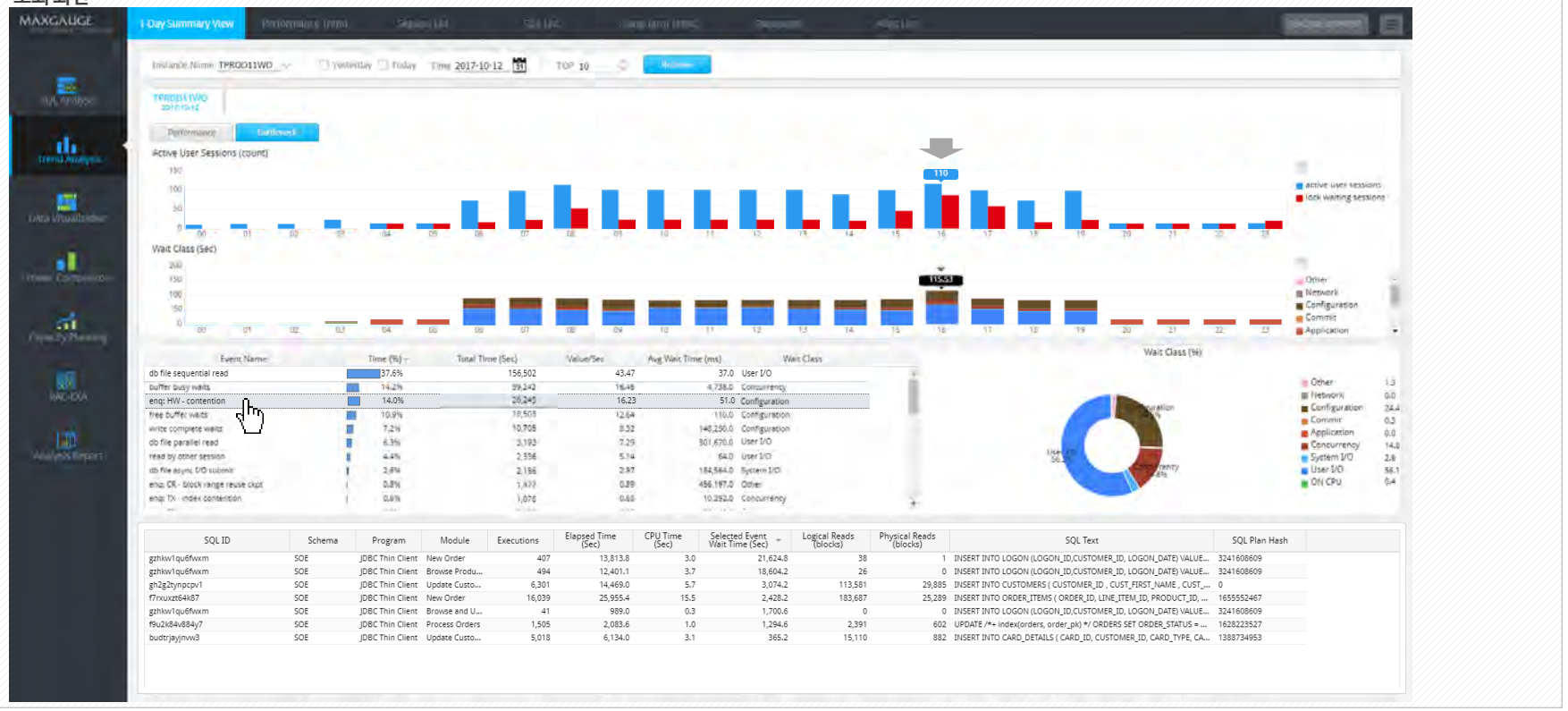
✓ Wait Time 정보 외에도 주요 성능 정보를 포함하여 SQL Text, Plan, Bind 등의 정보를 표시하므로 SQL 분석을 통하여 해당 SQL 의 문제를 파악할 수 있습니다.

"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



✓ 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.

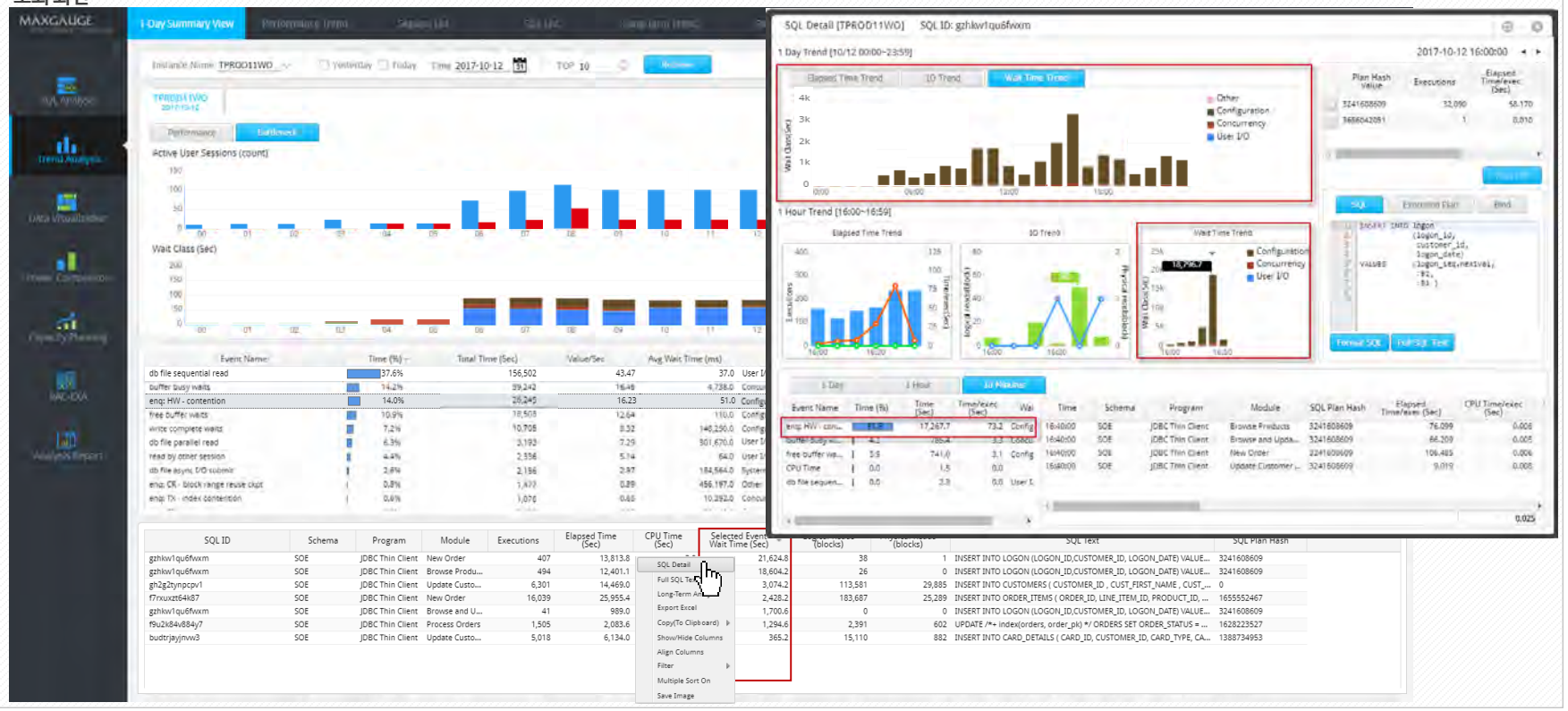
✓ 그 중 가장 높은 비율의 enq: HW – contention 이벤트를 클릭하였을 때, Selected Event Wait Time (Sec) 수치가 높은 "SQL ID : gzhkw1qu6fwxm" 에 해당하는 SQL 이 특히 대기 시간이 긴 것을 확인할 수 있습니다.

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)"

» 1-Day Summary View (Bottleneck) 조회 화면



✓ 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.

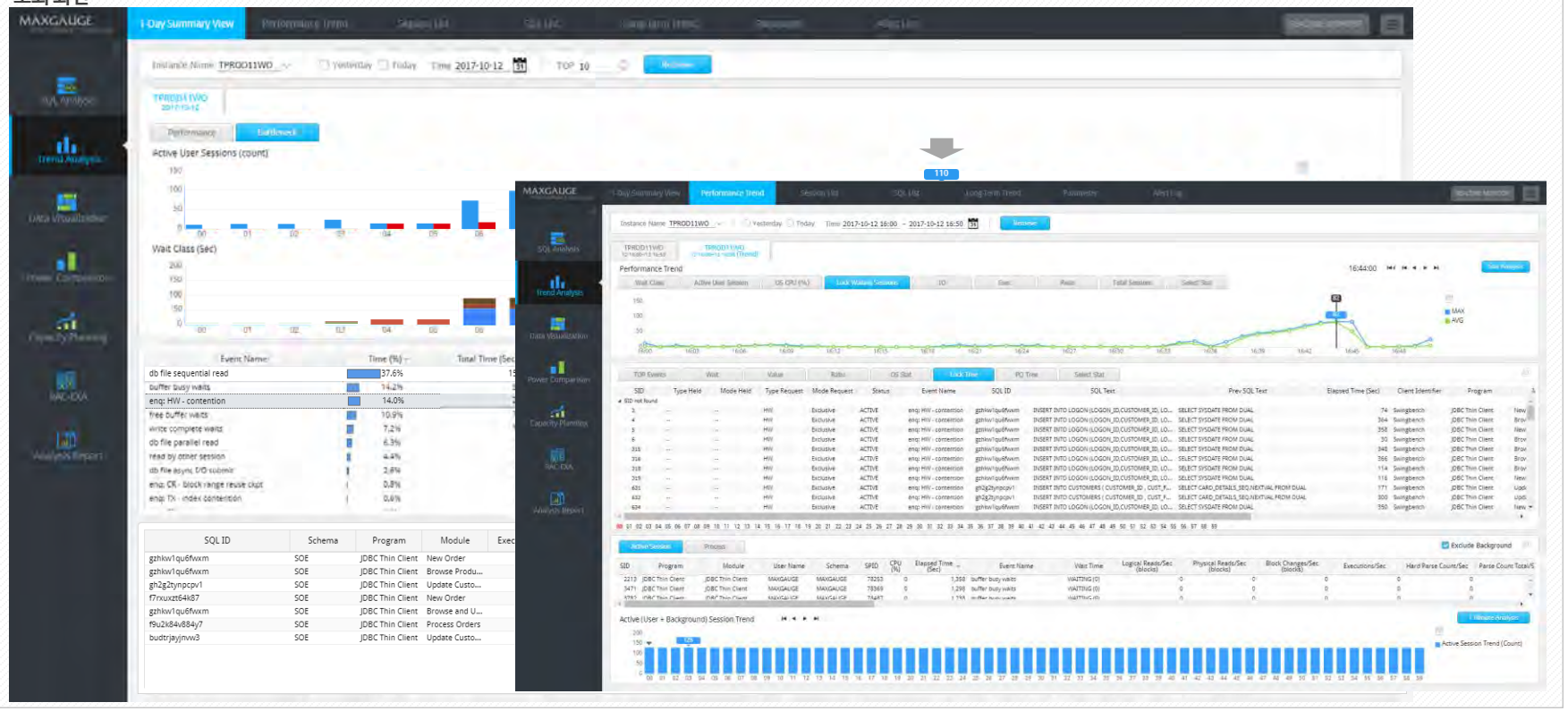
✓ Wait Time 정보 외에도 주요 성능 정보를 포함하여 SQL Text, Plan, Bind 등의 정보를 표시하므로 SQL 분석을 통하여 해당 SQL 의 문제를 파악할 수 있습니다.

"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



✓ Performance Trend 에서 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 Tree 에서 Holder Session 또는 Blocking Session 기능을 활용해서 락 발생의 원인을 찾을 수 있습니다.

MAXGAUGE Practical Guide

Parameter View [Performance Analyzer]

Practical Guide

Parameter View

Contents

Parameter View?

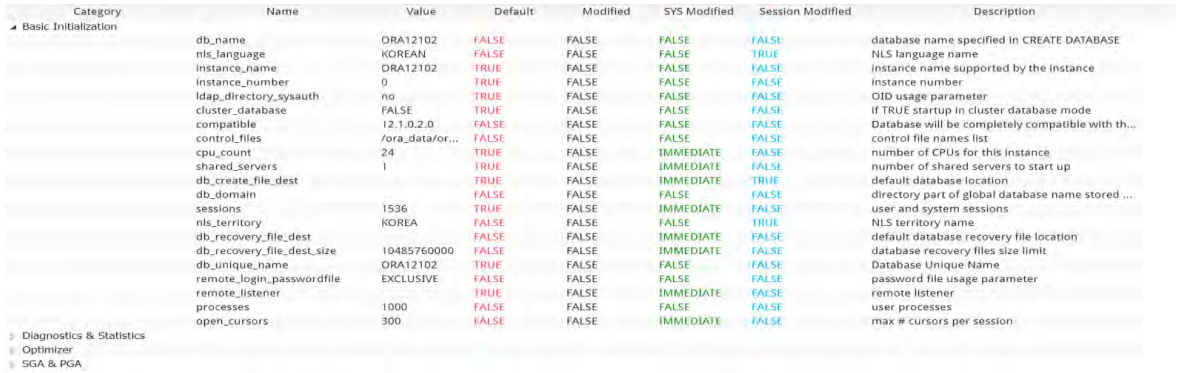
Parameter View의 활용방법은 무엇인가요?

» **파라미터 변경 이력을 통해**
성능 장애를 분석할 때

Parameter View

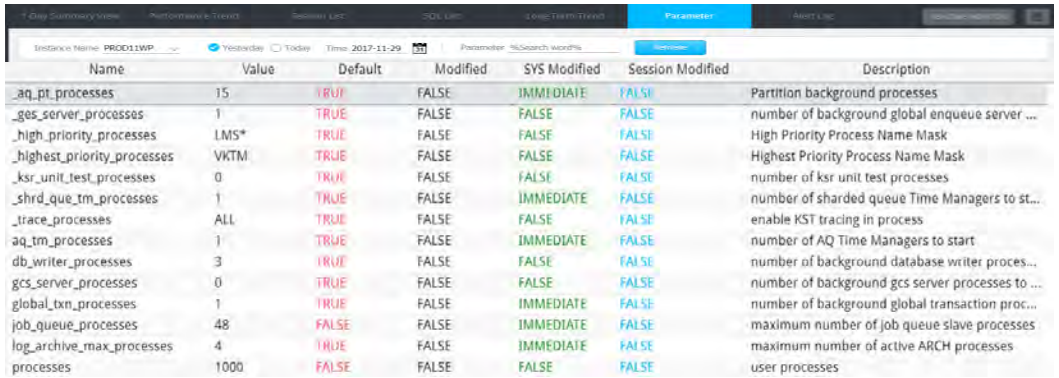
- 오라클에서 제공하는 모든 Parameter 정보를 제공하는 기능이며 Performance Analyzer ▸ **Tend Analysis** ▸ **Parameter** 경로를 통해 사용할 수 있다.
- Pre-Check 탭은 모니터링 DB에 설정된 현재 Parameter 에 대한 카테고리 별 현황을 보여준다.
    - Basic Initialization / Diagnostics & Statistics / Optimizer / SGA & PGA
- Search Result 탭은 Parameter 명을 %Search Word%로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

Pre-Check



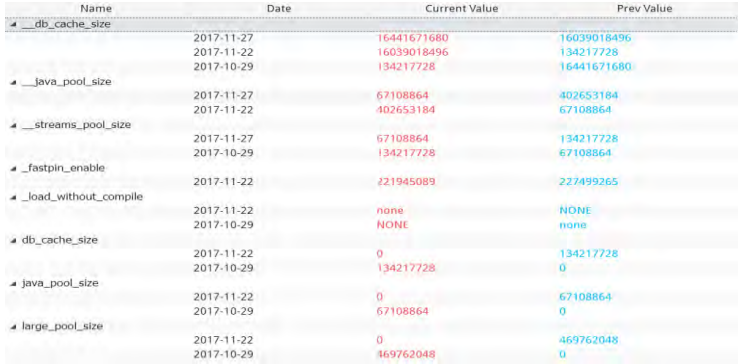
Category	Name	Value	Default	Modified	SYS Modified	Session Modified	Description
◢ Basic Initialization							
	db_name	ORA12102	FALSE	FALSE	FALSE	FALSE	database name specified in CREATE DATABASE
	nls_language	KOREAN	FALSE	FALSE	FALSE	TRUE	NLS language name
	instance_name	ORA12102	TRUE	FALSE	FALSE	FALSE	instance name supported by the instance
	instance_number	0	TRUE	FALSE	FALSE	FALSE	instance number
	ldap_directory_sysauth	no	TRUE	FALSE	FALSE	FALSE	OID 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	10485760000	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	FALSE	IMMEDIATE	FALSE	max # cursors per session
▸ Diagnostics & Statistics							
▸ Optimizer							
▸ SGA & PGA							

Search Result



Name	Value	Default	Modified	SYS Modified	Session Modified	Description
_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 ksr 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	FALSE	FALSE	user processes

Modification History



Name	Date	Current Value	Prev Value
◢ __db_cache_size			
	2017-11-27	16441671680	16039018496
	2017-11-22	16039018496	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			
	2017-11-22	221945089	227499265
◢ _load_without_compile			
	2017-11-22	none	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

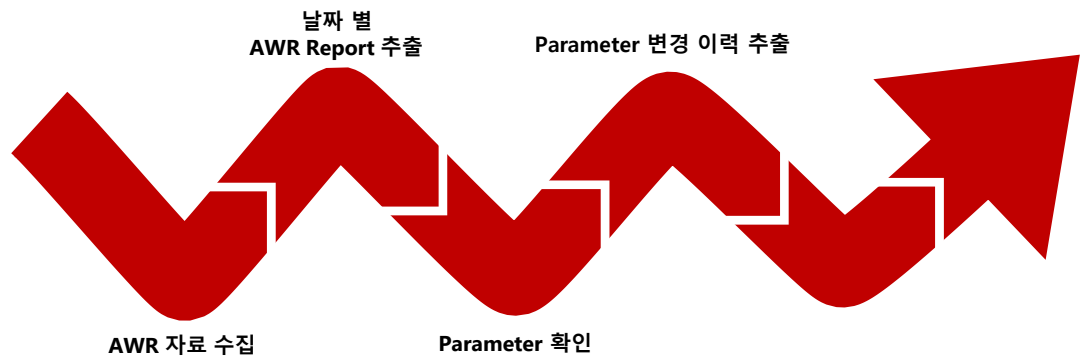
Parameter 성능 분석을 더욱 쉽게 할 수 있습니다!

성능 분석 방법

MaxGauge

Performance Analyzer → Parameter View → 성능 분석

Guide

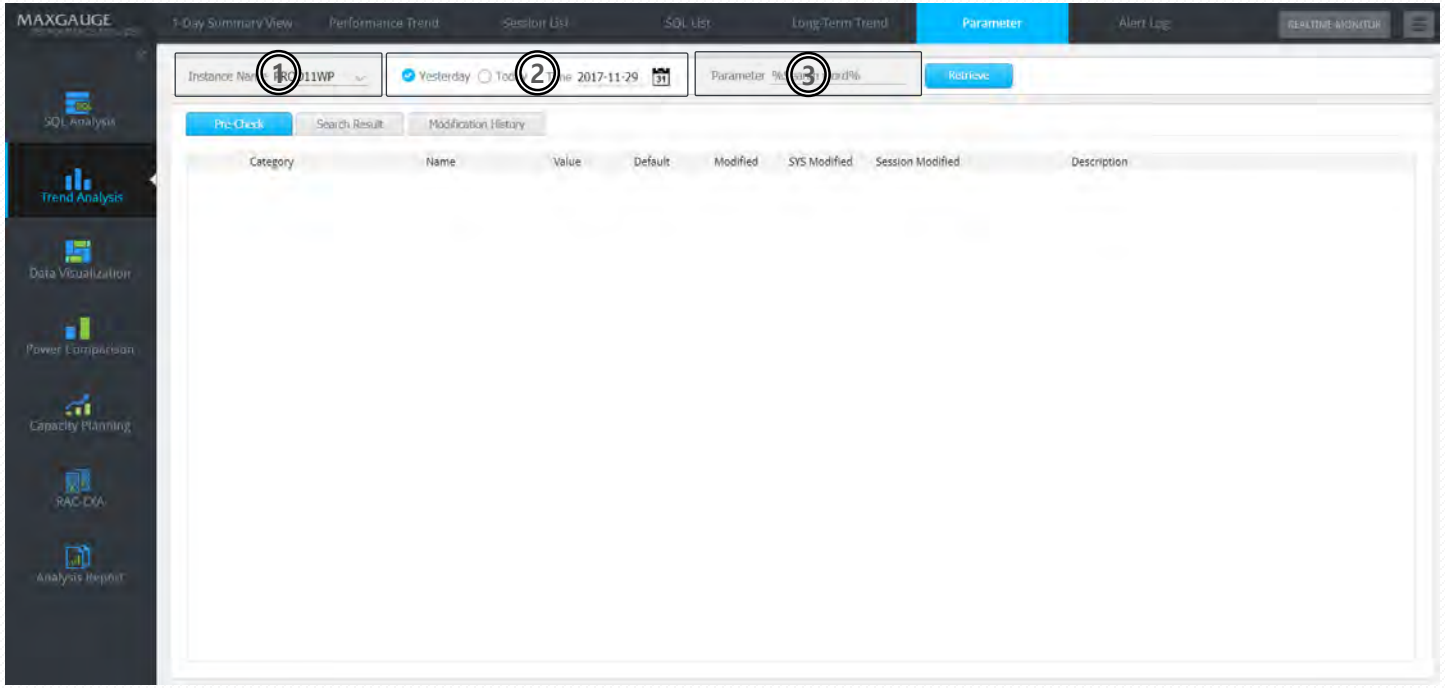


성능개선

Parameter View 의 활용

Parameter View 의 사용 방법

Parameter View 검색 화면



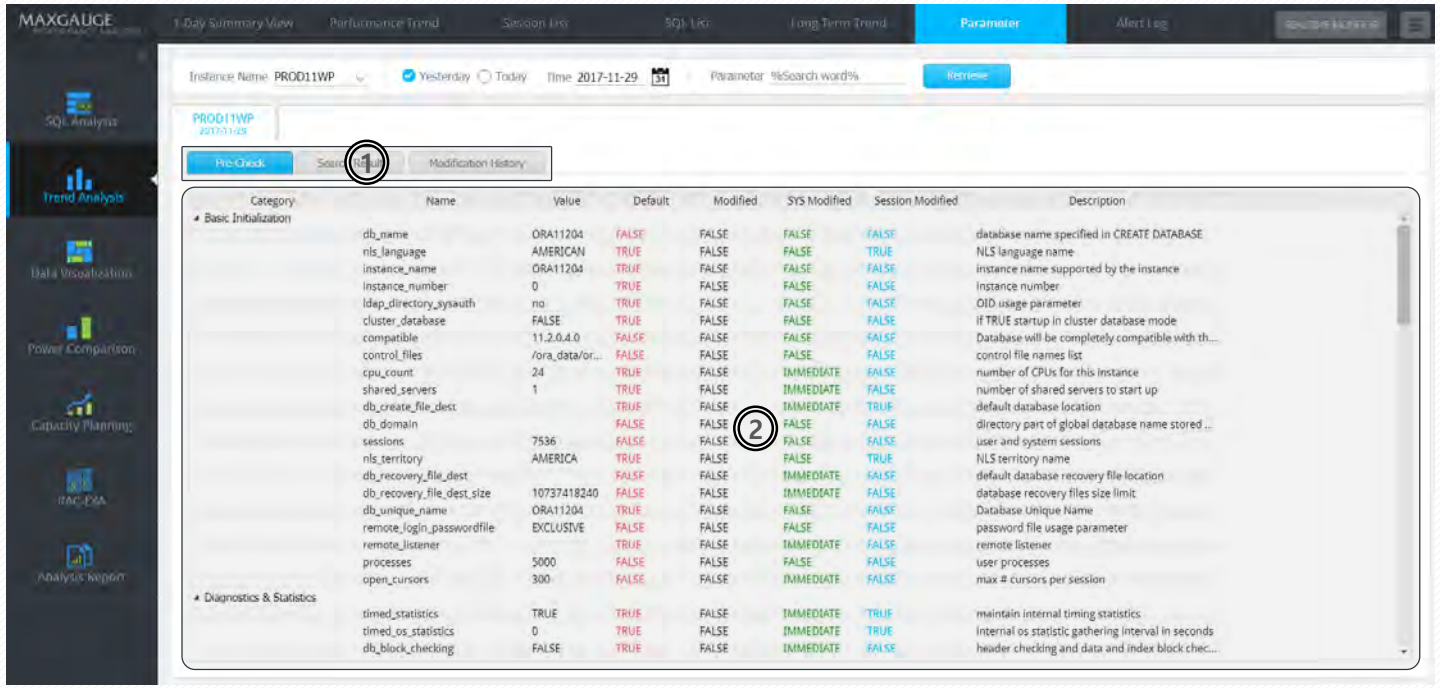
① 대상 Instance를 선택합니다.

② 일단위로 시간을 선택합니다.

③ Parameter 이름을 like 조건과 동일한 %Search Word% 를 통해 검색할 수 있습니다.

Parameter View 의 사용 방법

Parameter View 조회 화면



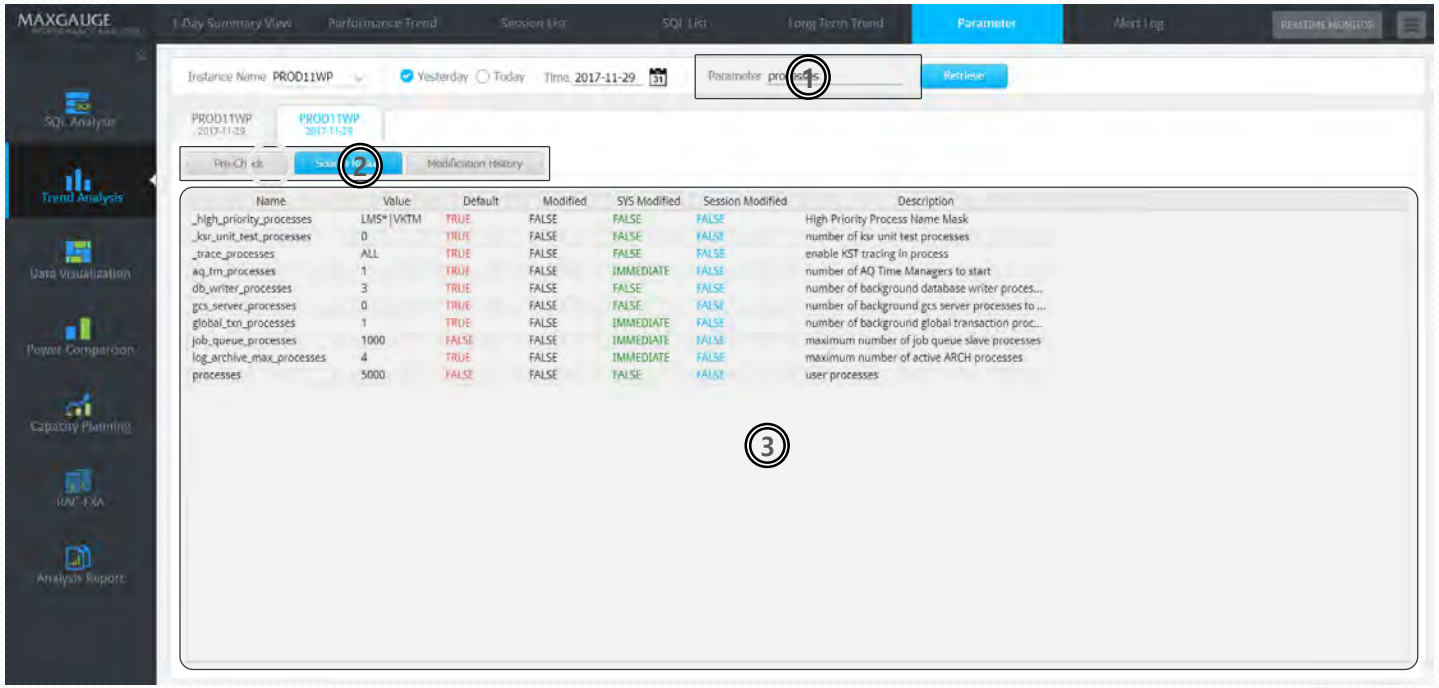
① 검색결과는 Pre-Check, Search Result, Modification History 으로 분류됩니다.

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 조회 화면



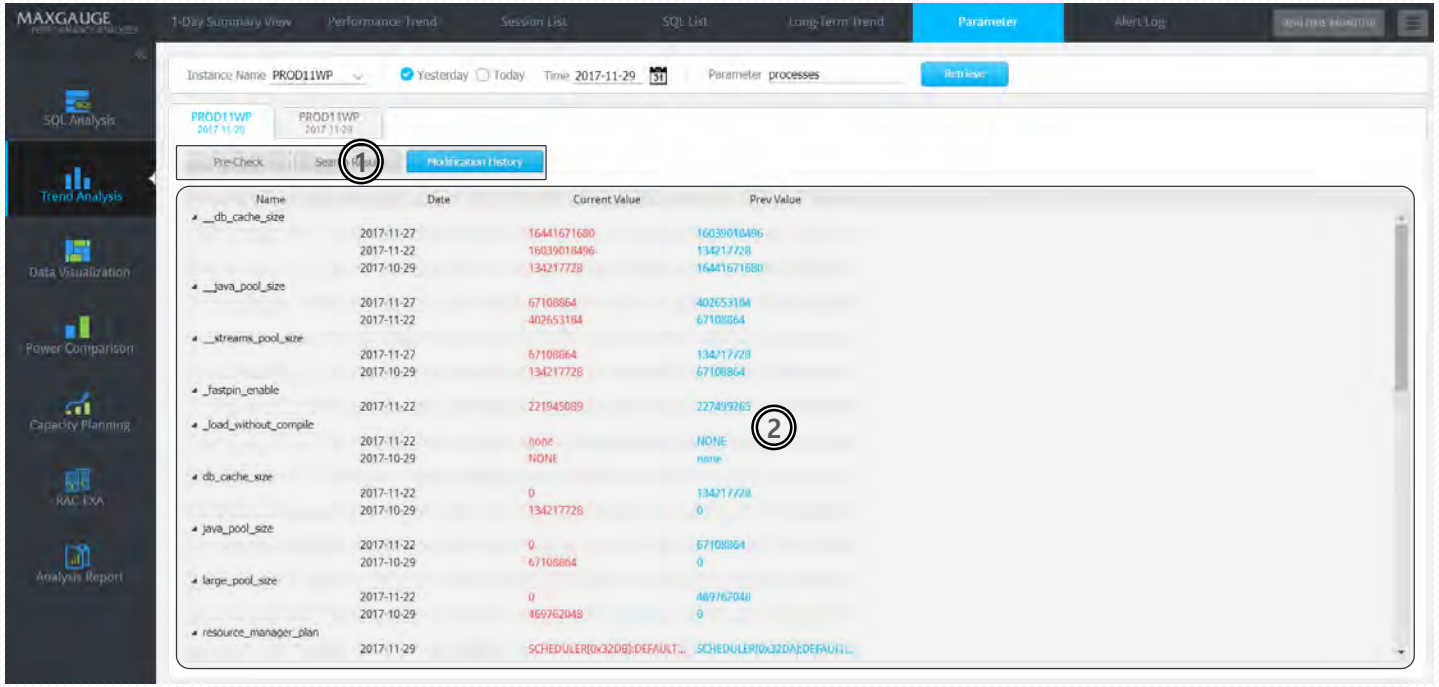
① Parameter 이름을 like 조건인 %Search Word% 로 검색합니다. (공백 입력 시 모든 Parameter 출력)

② 검색결과는 Pre-Check, Search Result, Modification History 으로 분류됩니다.

③ Search Result 탭은 검색조건에 따른 Parameter 정보를 제공합니다.

Parameter View 의 사용 방법

Parameter View 조회 화면



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

1. **문제발생**
2. 문제 분석
   - Parameter 확인
   - Parameter 변경이력 확인
3. 문제 해결

» MaxGauge 수집서버 운용 중 **문제발생**

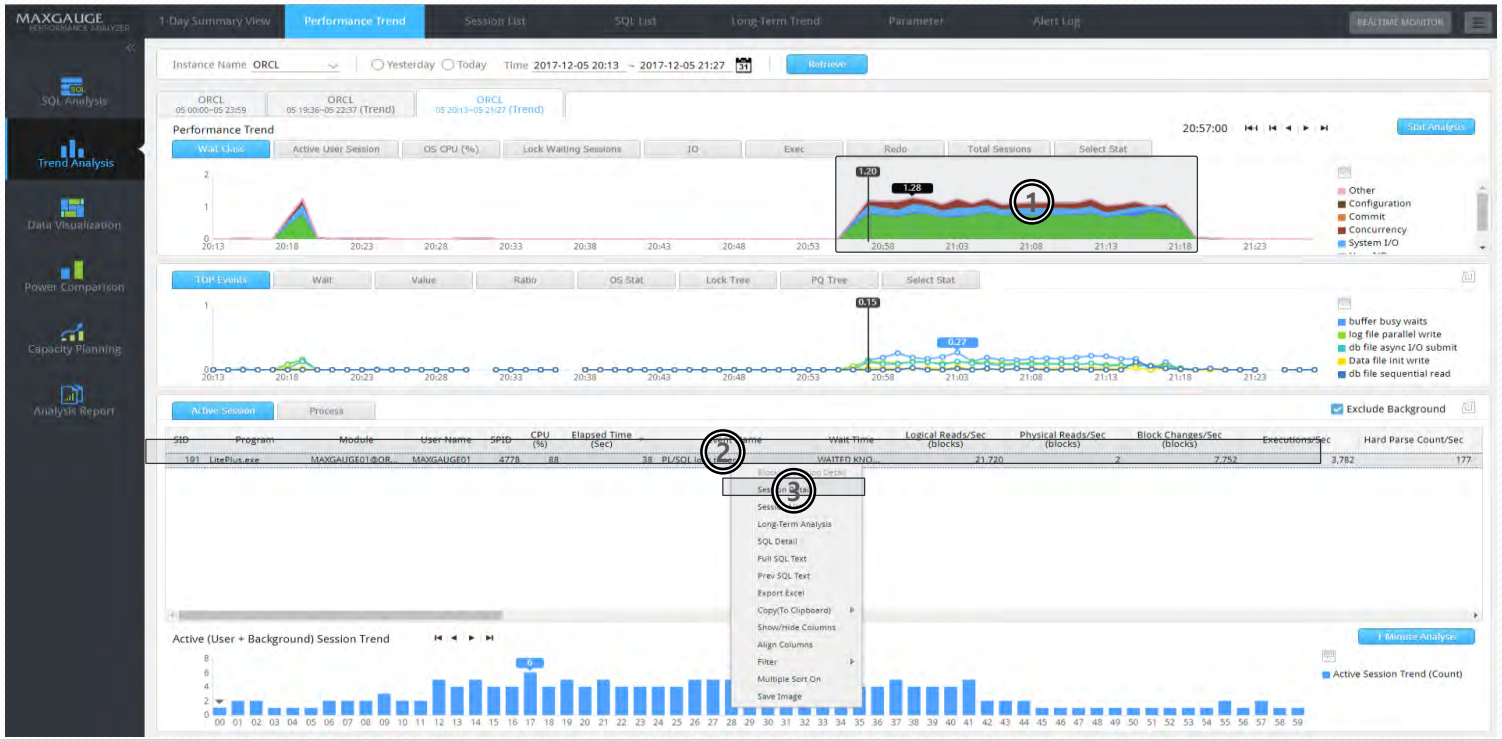
- ✓ 하루에 한번 수행되는 MaxGauge 파티션 생성 작업을 정기 PM 작업 시 6개월 치 생성으로 대체하여 수행하는 배치 업무
- ✓ 6개월 치의 파티션을 미리 생성하는 배치 작업 수행 시 오랜 시간 대기하는 현상 발생
- ✓ 이를 분석하고자 MaxGauge 의 Performance Trend, Session Detail, Parameter View 를 활용

"배치 업무가 느려진 이유를 알고 싶어요!"

STEP

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

파티션 생성 작업 지연



① Trend Analysis -> Performance Trend 에서 세션 20:56:00 ~ 21:02:00 시간대 7분 이상 파티션 생성 작업이 지연됨

② 20:57:00 시점에 대한 세션 리스트를 확인

② 해당 세션에 대한 상세정보를 확인하기 위해 마우스 오른쪽 버튼 -> "Session Detail" 클릭

STEP

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

"배치 업무가 느려진 이유를 알고 싶어요!"

파티션 생성 작업 지연

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

"배치 업무가 느려진 이유를 알고 싶어요!"

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

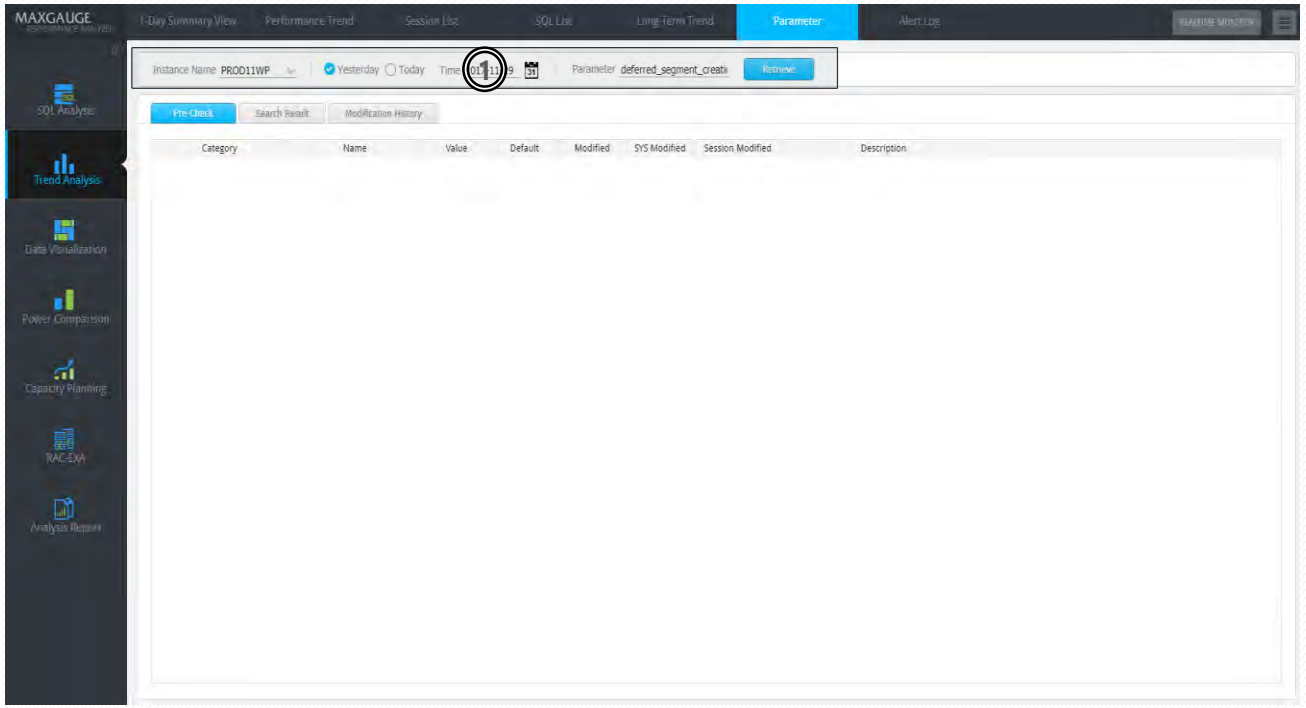
파라미터 전수 조사

STEP

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

"배치 업무가 느려진 이유를 알고 싶어요!"

Parameter View 검색 화면



✓ 시나리오

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

✓ 목표

Parameter 현재 값 확인 후 변경이력 파악

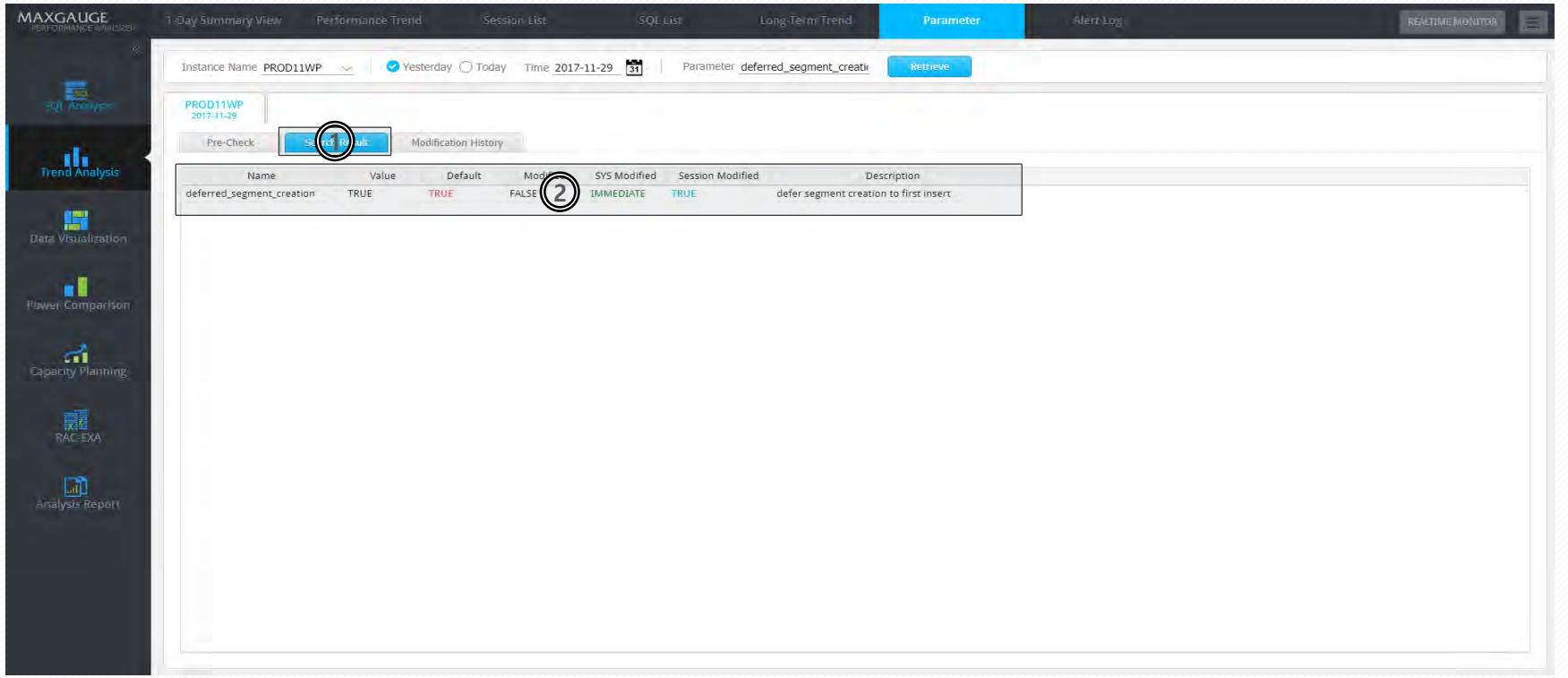
① 일별 Parameter 값을 확인하기 위한 검색조건 추가 **"deferred_segment_creation"**

"배치 업무가 느려진 이유를 알고 싶어요!"

STEP

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

» Parameter View 조회 화면



① 검색 조건에 부합하는 결과값을 보기 위해 Search Result 선택

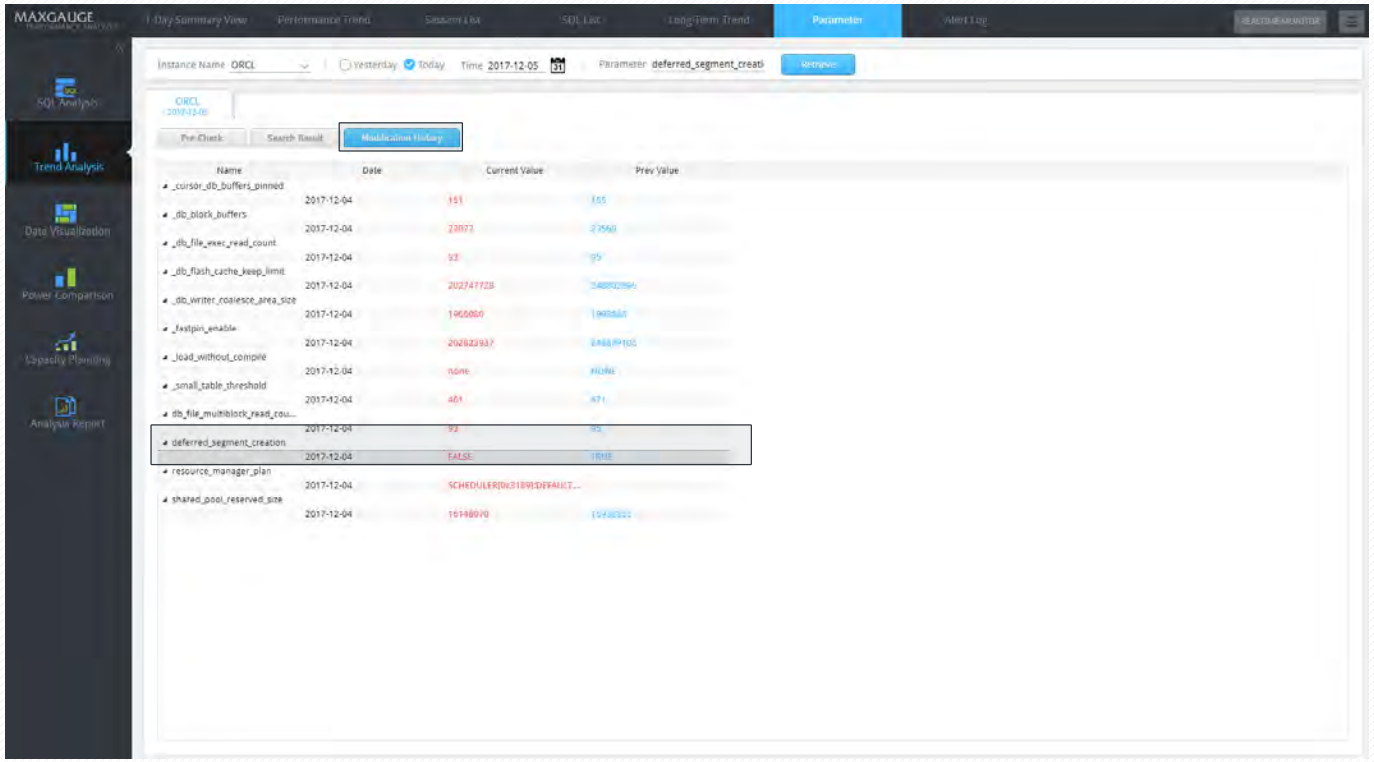
② "deferred_segment_creation" Parameter 현재 값을 확인

STEP

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

"배치 업무가 느려진 이유를 알고 싶어요!"

» Parameter View 조회 화면



✓ deferred_segment_creation 파라미터가 **TRUE -> FALSE 로 변경된 이력을 확인**

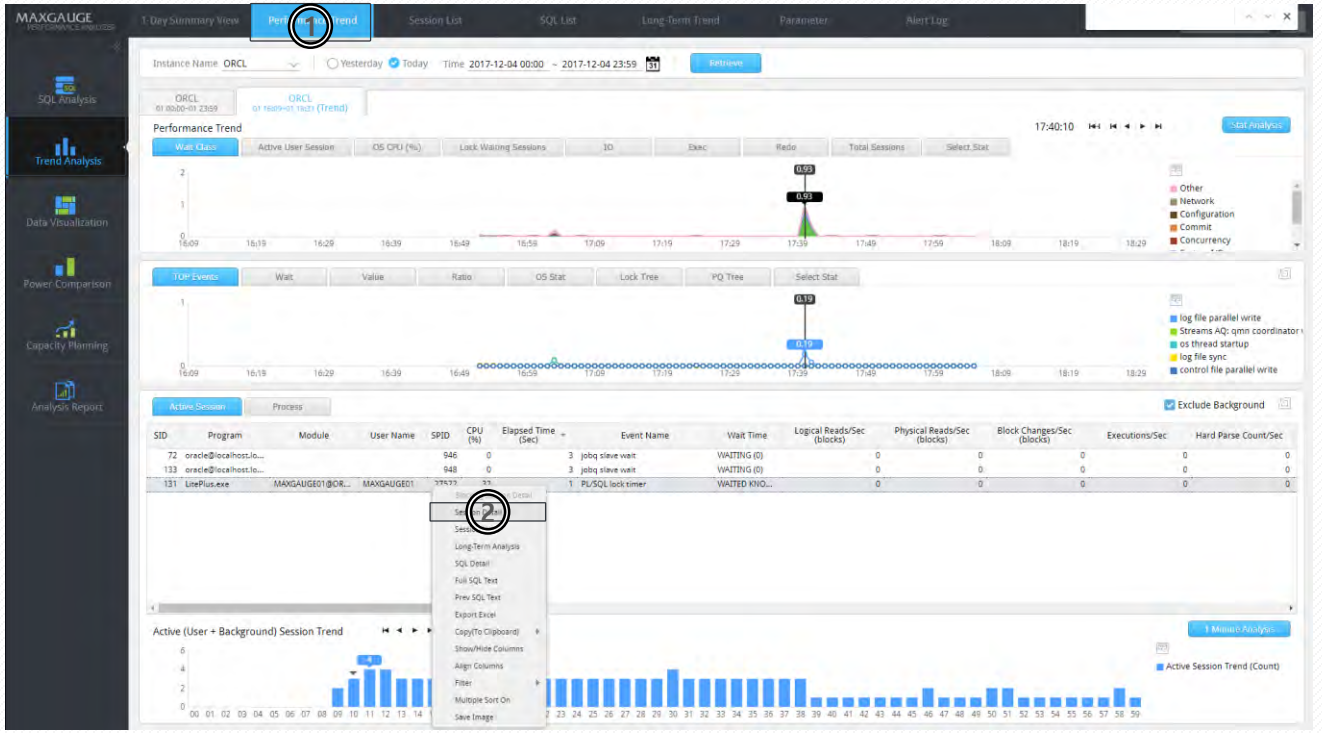
✓ 즉, 테이블(또는 파티션) 생성 시 데이터가 존재하지 않아도 SEGMENT 를 바로 생성

“배치 업무가 느려진 이유를 알고 싶어요!”

STEP

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

MaxGauge 수집서버 운용 정상화



① Trend Analysis -> Performance Trend 에서 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**

MaxGauge 수집서버 운용 정상화

① 17:40:10 ~ 17:40:14 약 1분 수행하며 빠르게 완료 됨

- 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]

PRODUCT
DOCUMENTATION

Practical Guide

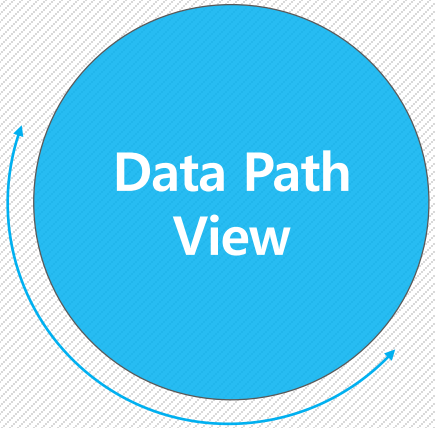
Data Visualization

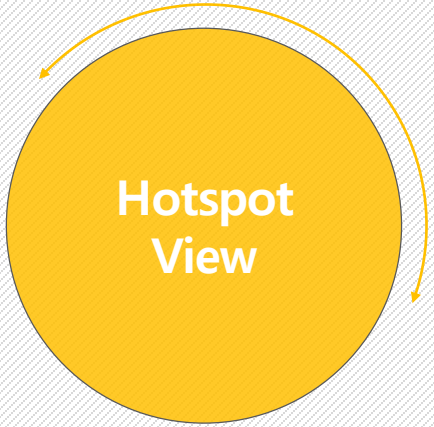
Contents

Data Visualization?

Data Visualization은 언제 쓰나요?

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





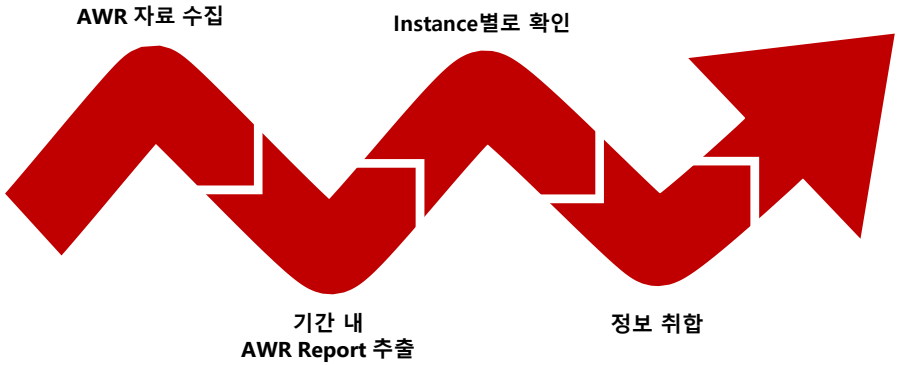
» 다수의 Instance 또는 여러 일자에 대한
CPU(%) hotspot 구간이 궁금할 때

Instance별 업무 비중도 를 더욱 쉽게 구분 할 수 있습니다!

Instance별 업무 비중도 확인 요청

MaxGauge

AWR

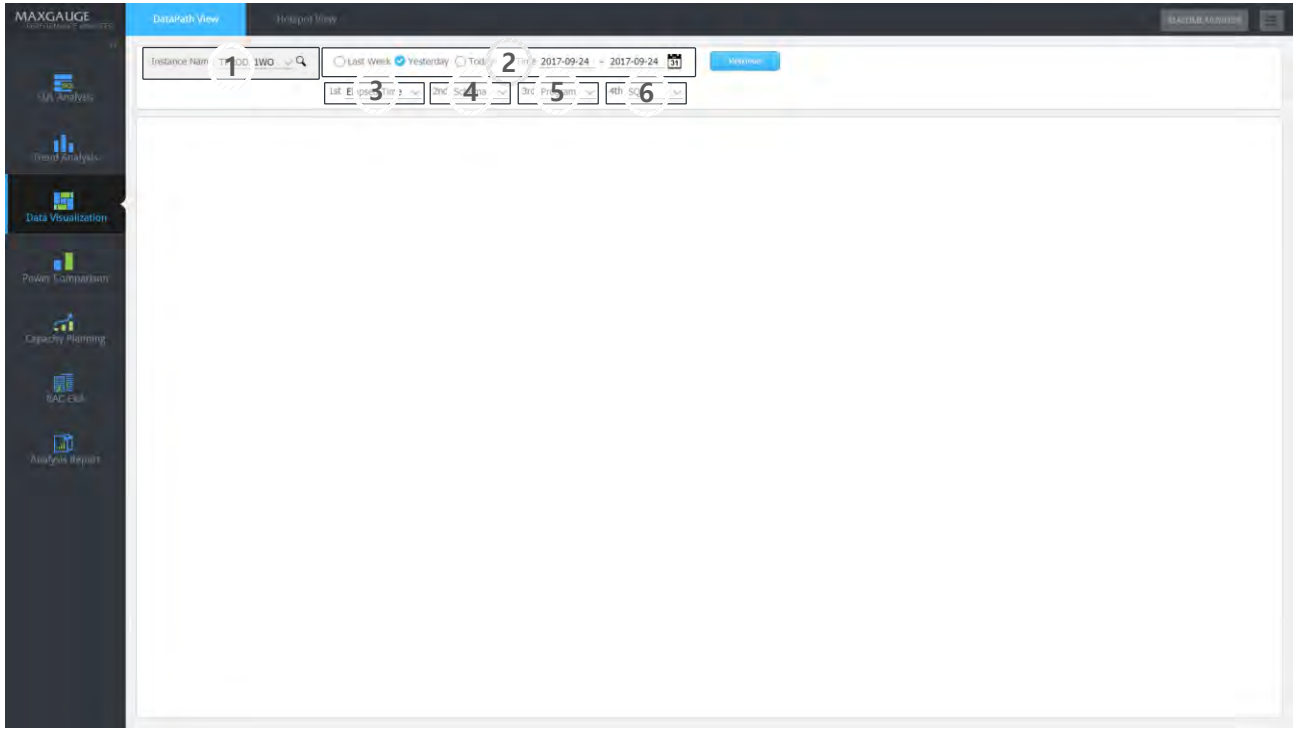


Instance별 업무 비중도 확인

DataPath View의 활용

DataPath View의 사용 방법

» DataPath view 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.

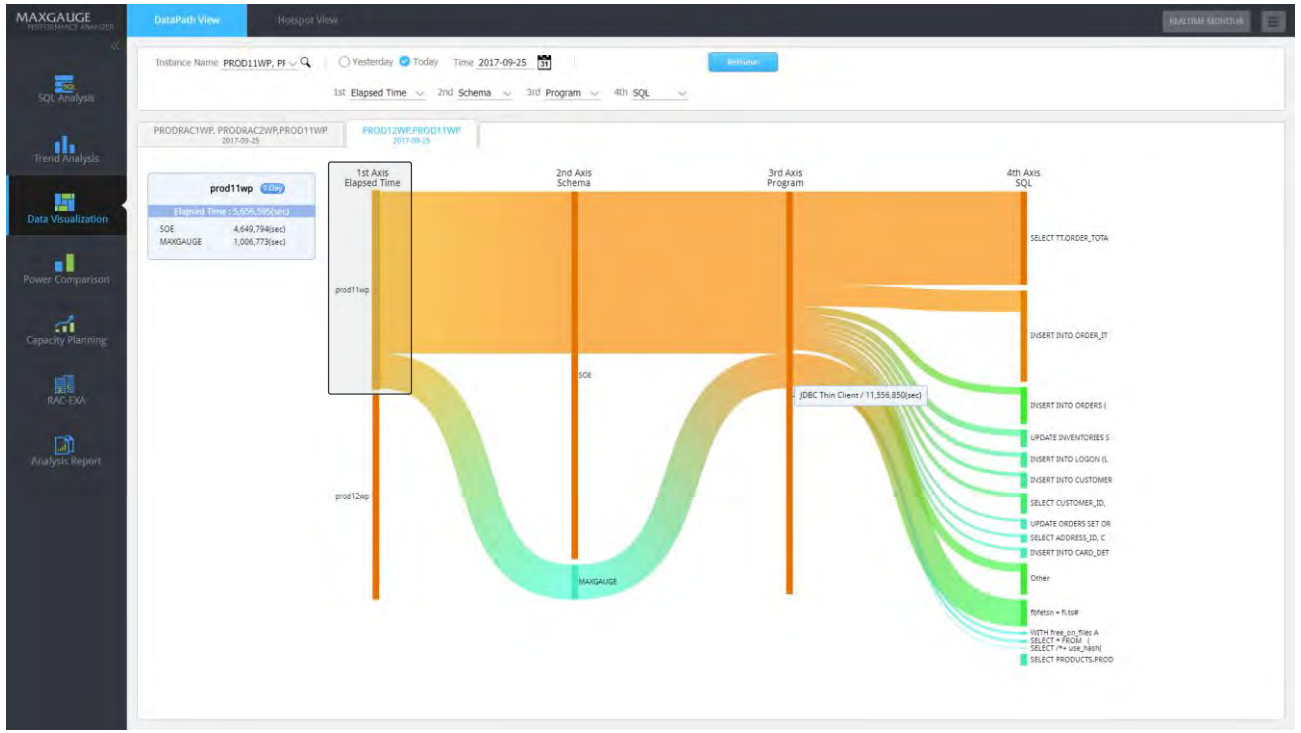
4 2차 축에 해당하는 Schema, Program, Module 중에서 선택 할 수 있습니다.

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 결과 화면 (1차 축 내의 특정 instance 선택 시의 화면)

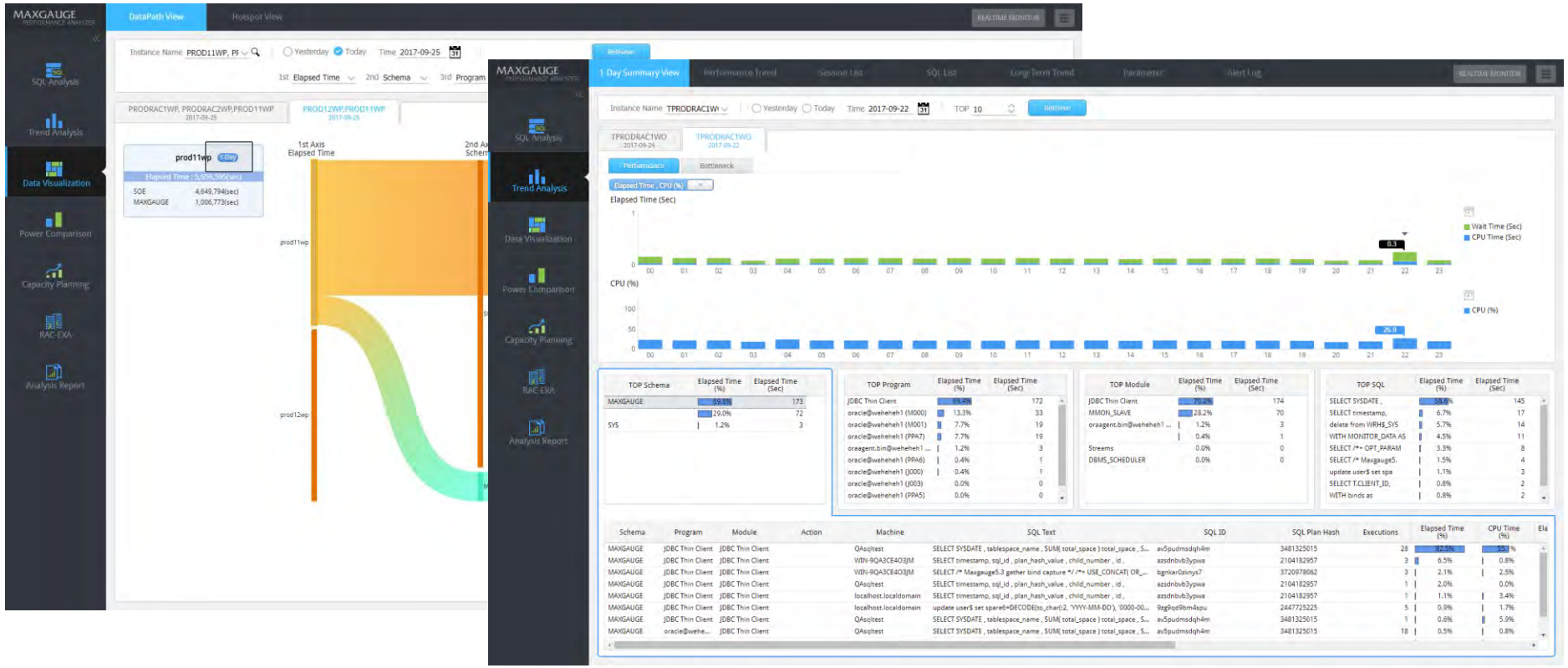


✓ 1차 축인 Elapsed Time에서 원하는 instance 를 선택합니다.

✓ 선택 된 Instance 를 기준으로 Instance -> Schema -> Program -> SQL로의 Top-Down 방식의 분석이 가능합니다.

DataPath View의 사용 방법

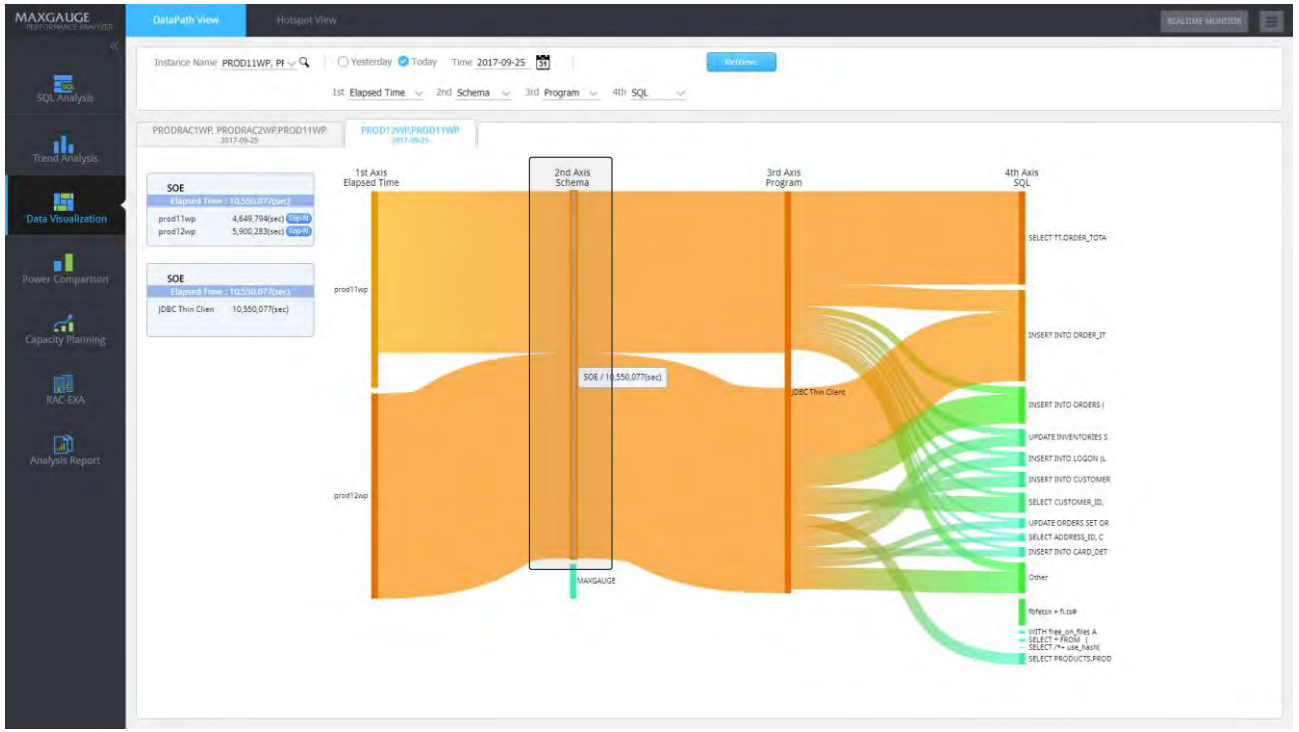
» DataPath view 검색 결과 화면 (1차 축 내의 특정 instance 선택 시의 화면)



✓ 좌측 상단의 1 –Day 버튼을 클릭하면 해당 Instance에 대한 1-Day Summary View 로 연계 됩니다.

DataPath View의 사용 방법

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

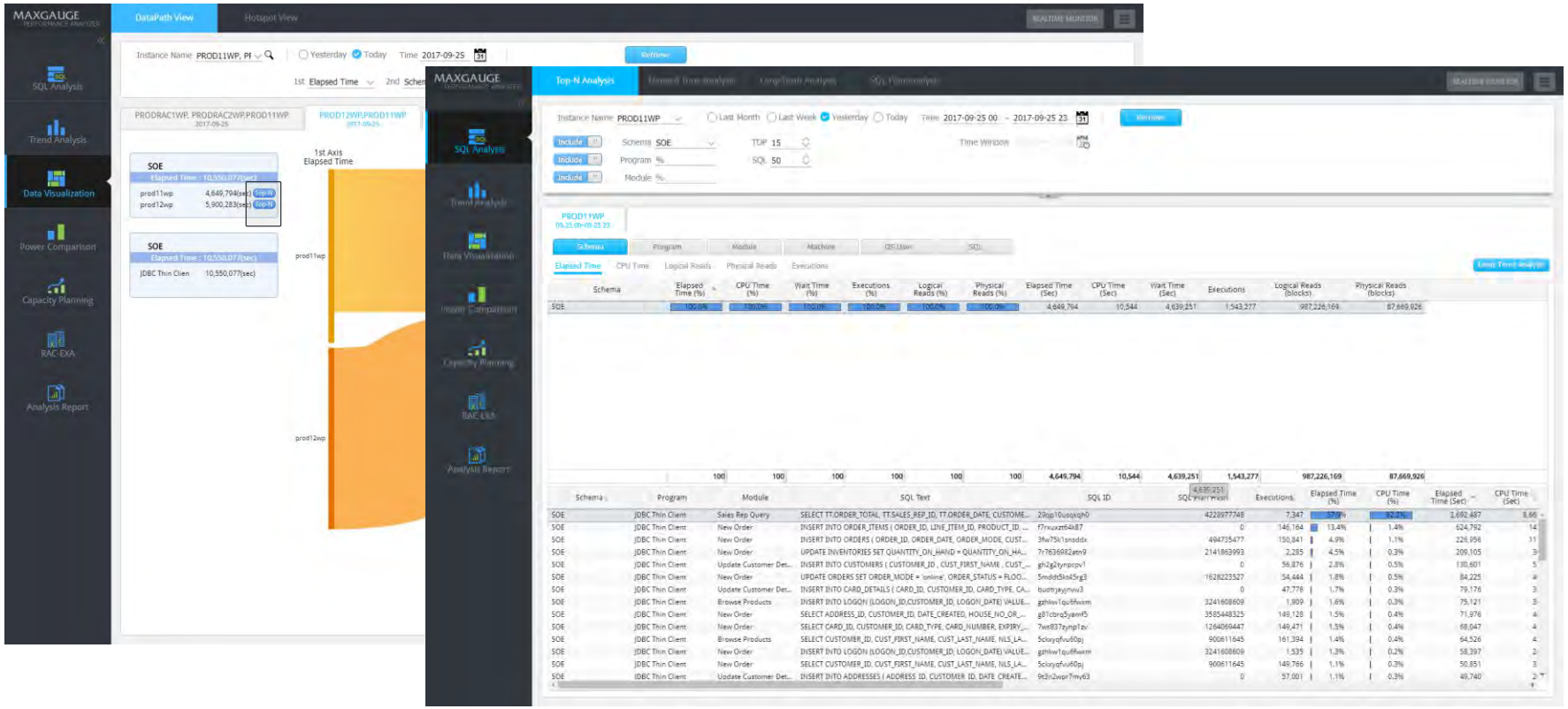


✓ 2차 축인 Schema에서 원하는 Schema 를 선택합니다.

✓ 선택 된 Schema 를 기준으로 Instance <- Schema -> Program -> SQL로의 양방향의 분석이 가능합니다.

DataPath View의 사용 방법

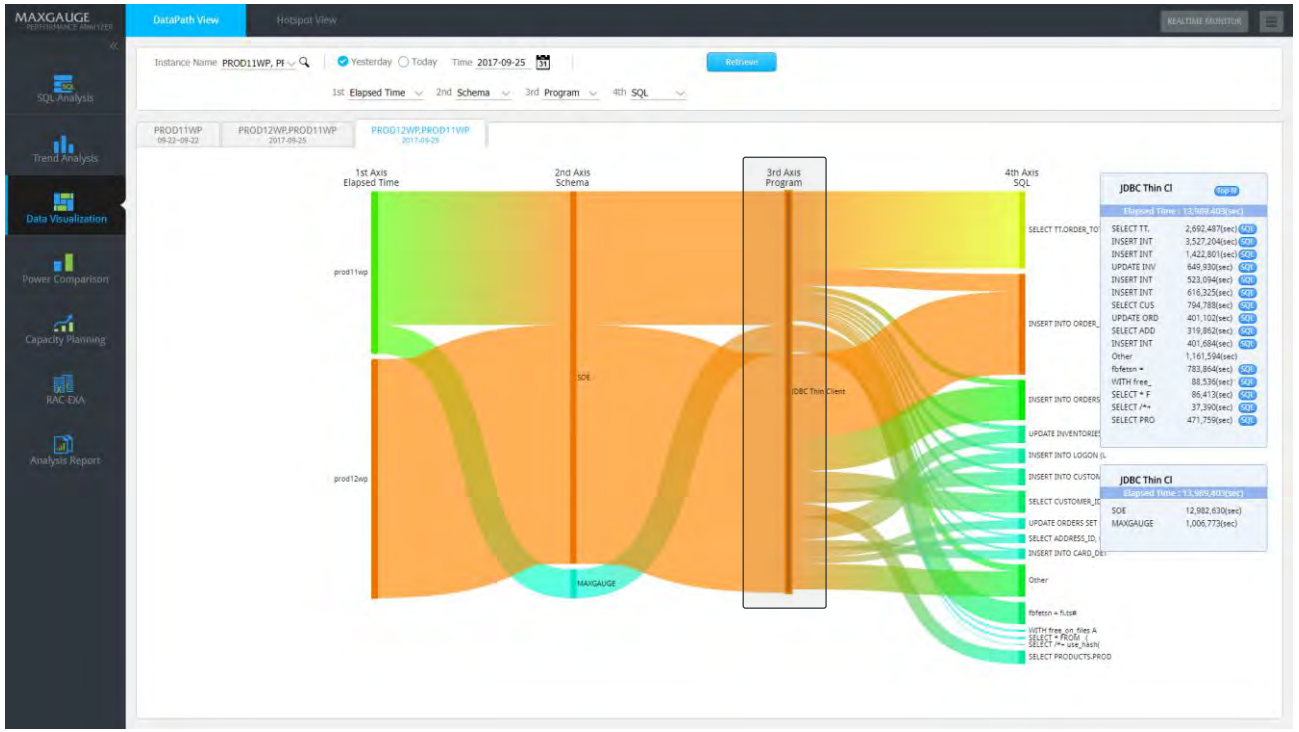
» DataPath view 검색 결과 화면 (2차 축 내의 특정 Schema선택 시의 화면)



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

DataPath View의 사용 방법

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

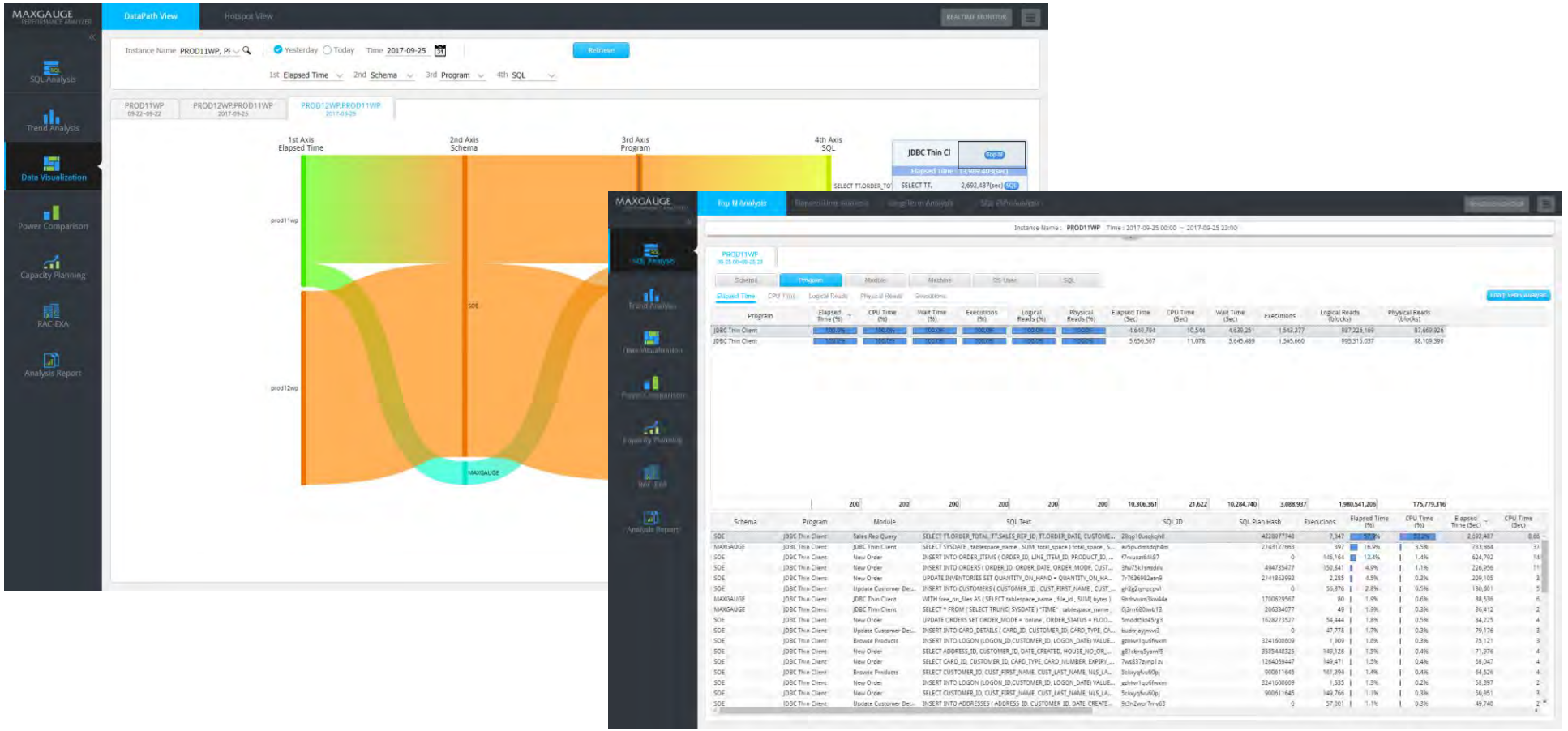


✓ 3차 축인 Program에서 원하는 Program을 선택합니다.

✓ 선택 된 Schema 를 기준으로 Instance <- Schema <- Program -> SQL로의 양방향의 분석이 가능합니다.

DataPath View의 사용 방법

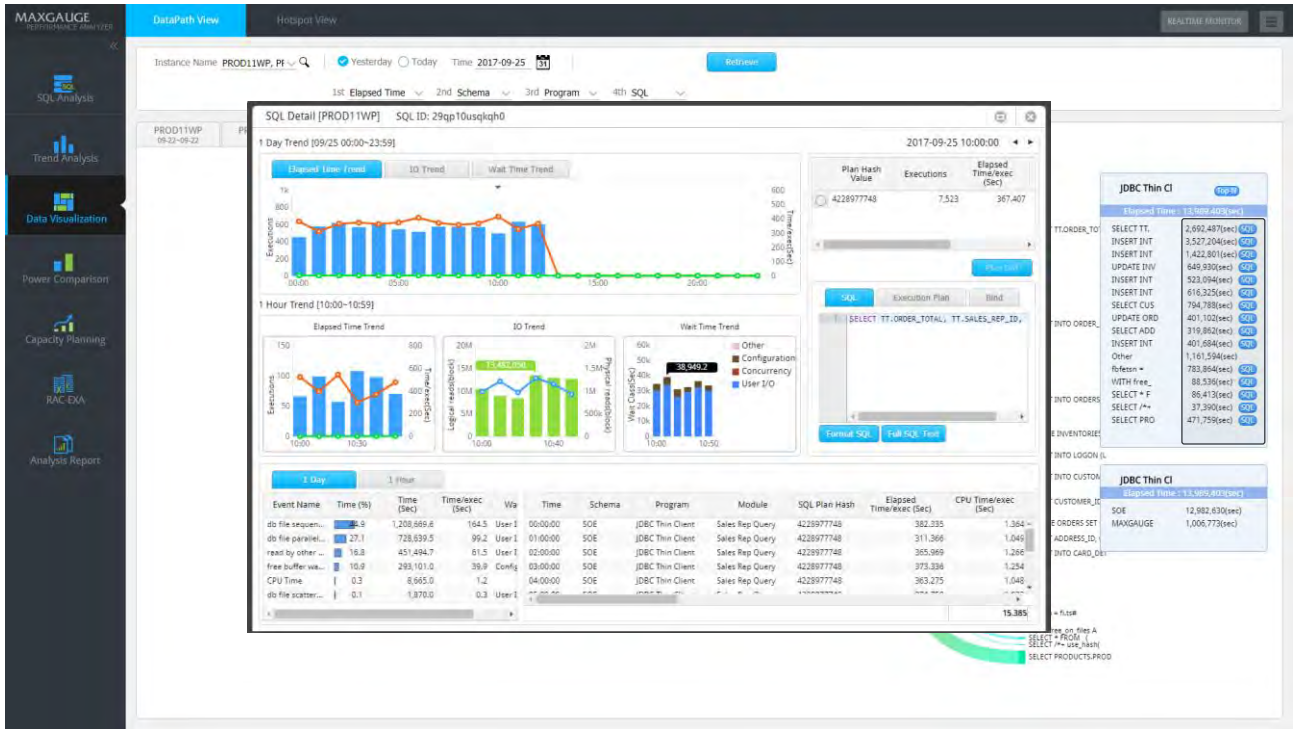
» DataPath view 검색 결과 화면 (2차 축 내의 특정 Schema선택 시의 화면)



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

DataPath View의 사용 방법

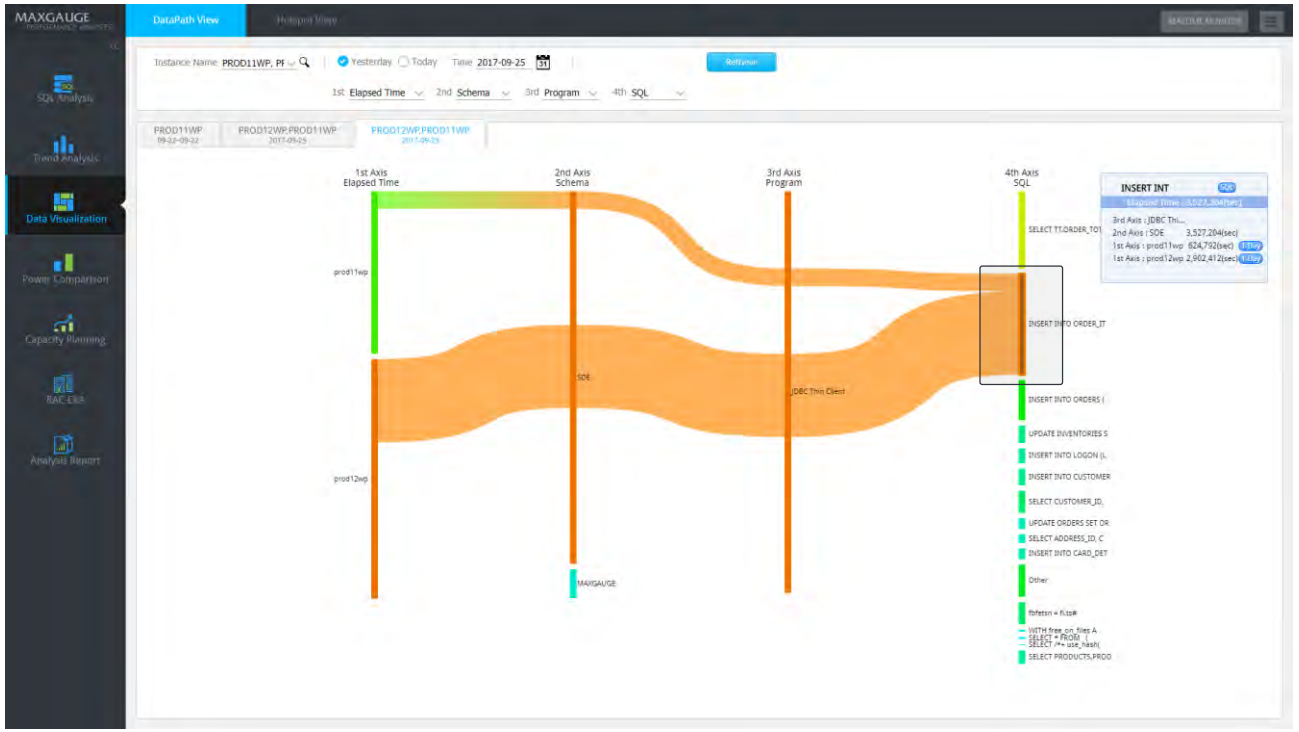
» DataPath view 검색 결과 화면 (3차 축 내의 특정 Program선택 시의 화면)



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

DataPath View의 사용 방법

» DataPath view 검색 결과 화면 (4차 축 내의 특정 SQL선택 시의 화면)

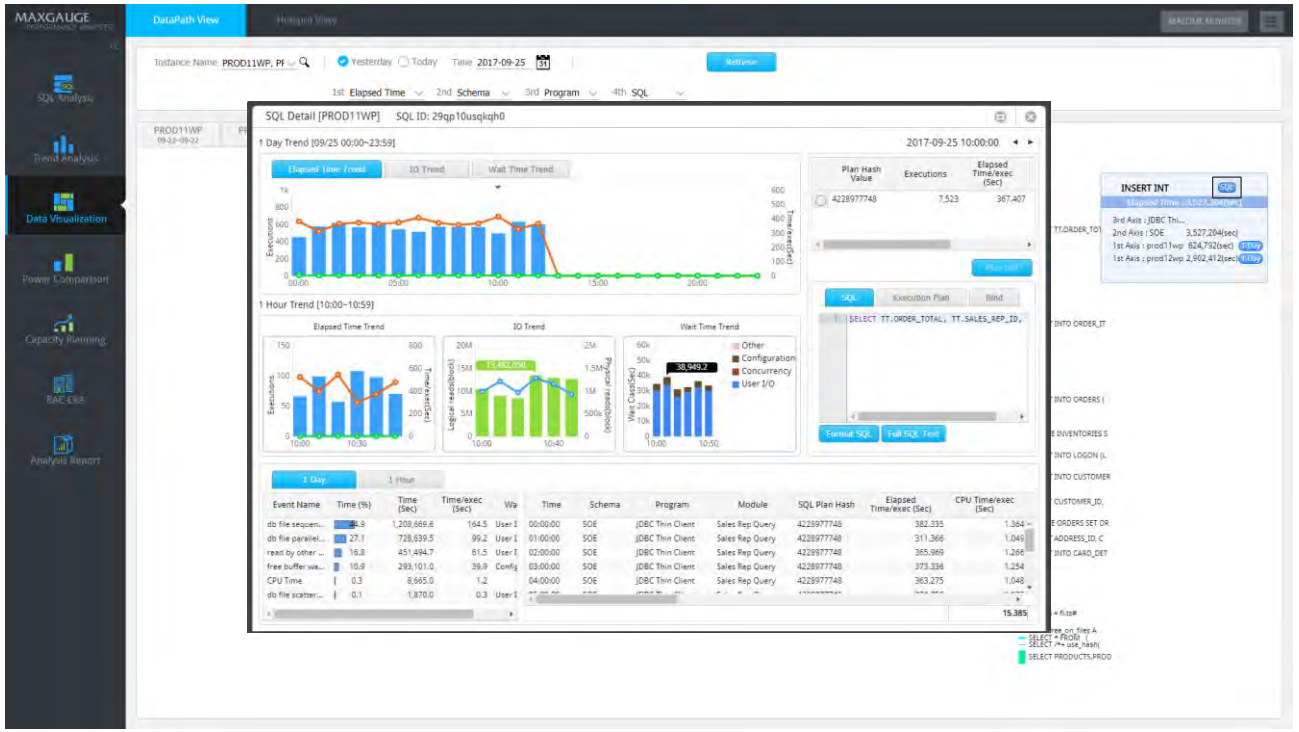


✓ 4차 축인 SQL에서 원하는 SQL을 선택합니다.

✓ 선택 된 Schema 를 기준으로 Instance <- Schema <- Program <- SQL로의 bottom-Up 분석이 가능합니다.

DataPath View의 사용 방법

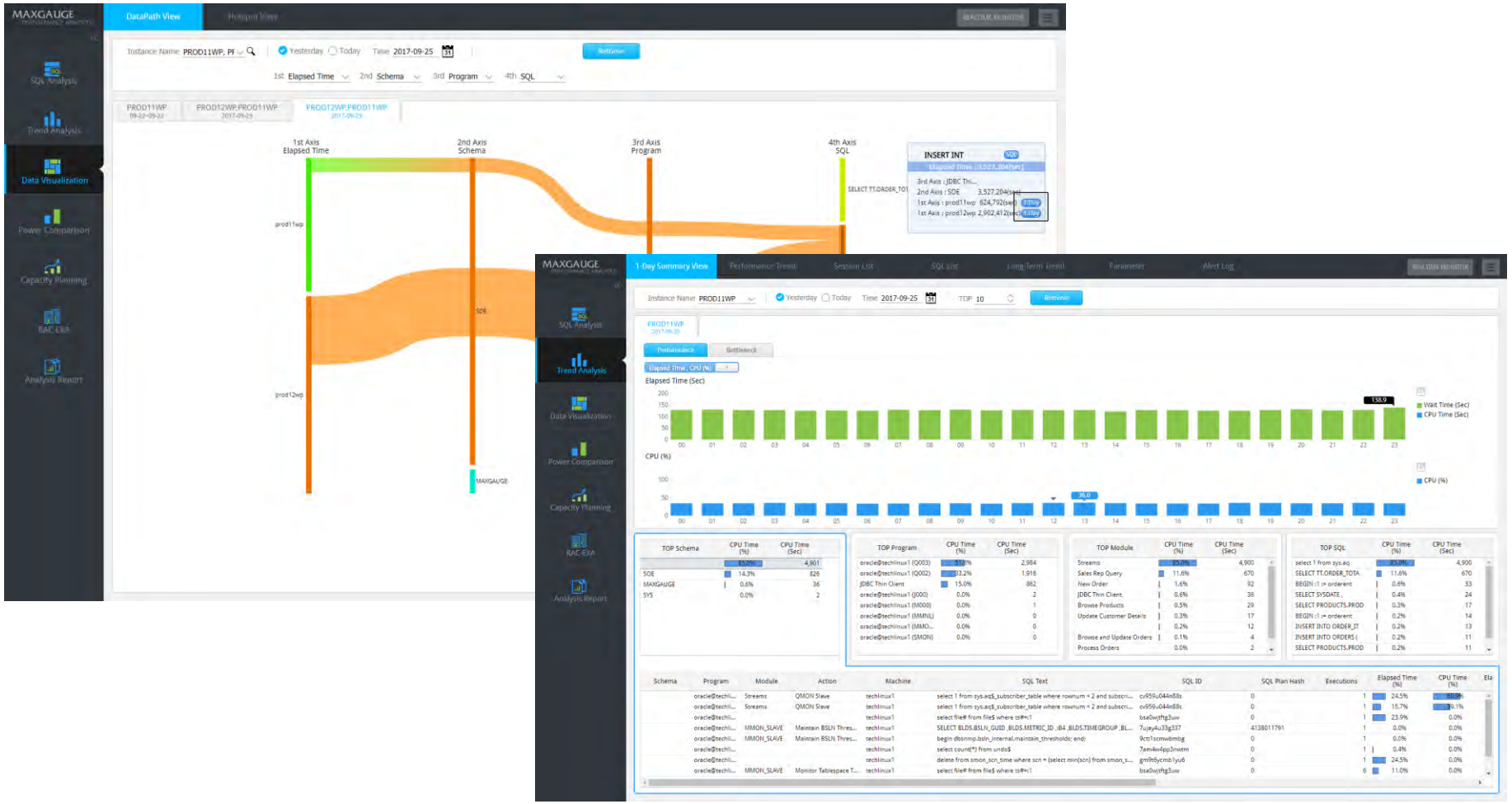
» DataPath view 검색 결과 화면 (4차 축 내의 특정 SQL선택 시의 화면)



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

DataPath View의 사용 방법

» DataPath view 검색 결과 화면 (4차 축 내의 특정 SQL선택 시의 화면)



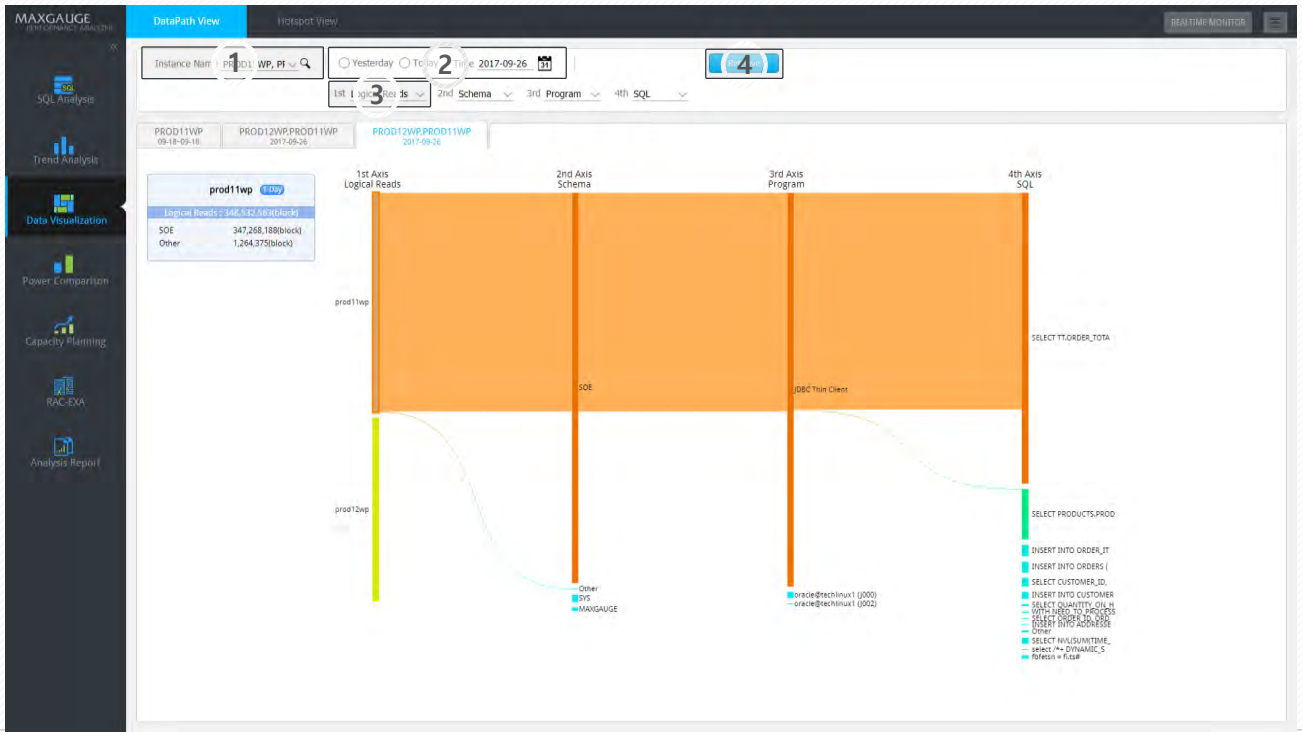
✓ 우측 1-Day 버튼을 클릭하면 해당 Instance에 대한 1-Day Summary View 로 연계 됩니다.

실전 분석 사례 Case 1.

어떤 노드가 업무비중도가 높은가요?

"어떤 노드가 업무비중도가 높은가요?"

DataPath View검색 화면



✓ **시나리오**

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

✓ **목표**

업무 비중도가 높은 노드를 손쉽게 확인 하기

1 비교 대상 Instance (PROD11WP, PROD12WP)를 선택합니다.

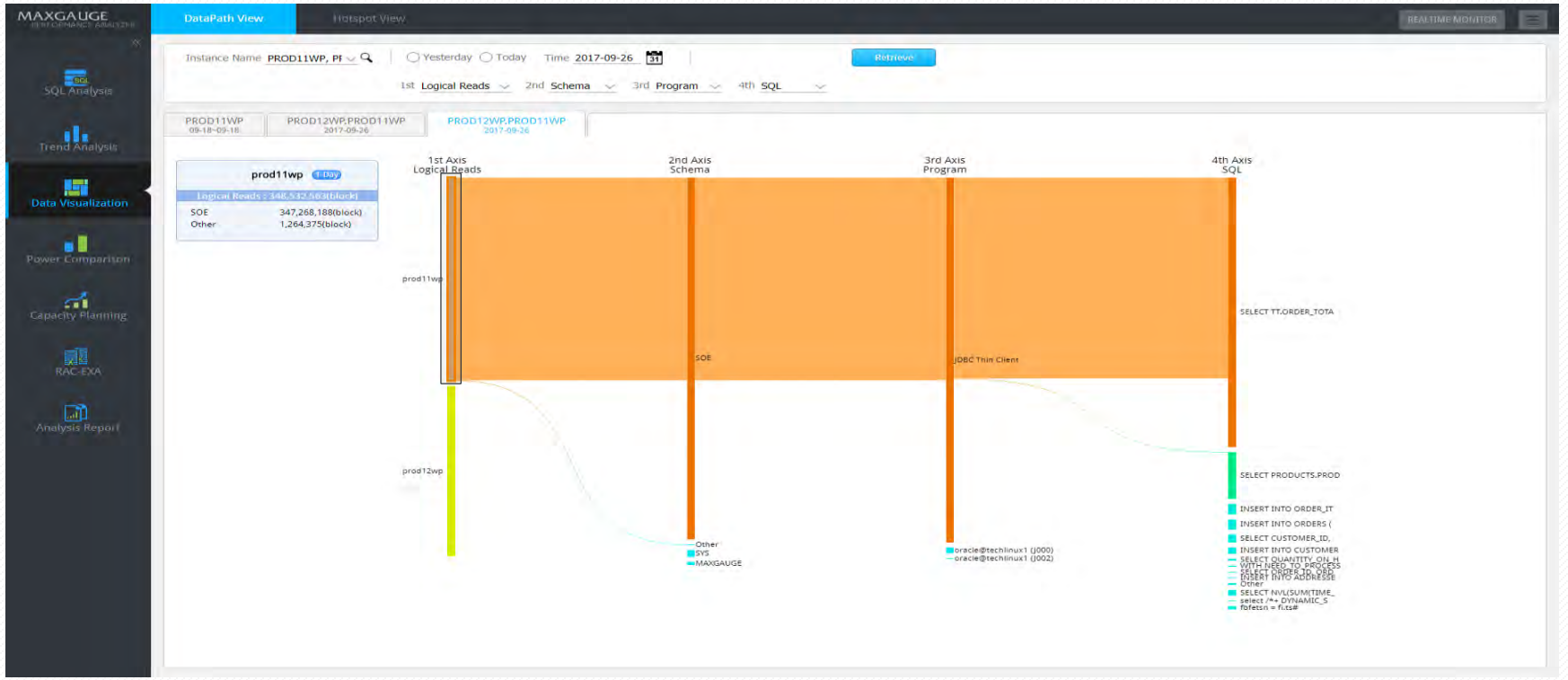
2 분석 날짜 (**2017.09.26**) 는 하루만 선택 가능합니다.

3 업무 비중 도를 비교 하기 위해 1차 축은 'Logical Reads'를 선택 합니다.

4 위의 설정한 조건으로 DataPath View를 **실행합니다.**

"어떤 노드가 업무비중도가 높은가요?"

» Top-SQL Comparison 1:M 조회 화면



✓ 1차 축은 Logical Reads 를 길이로 표현 한 것입니다.

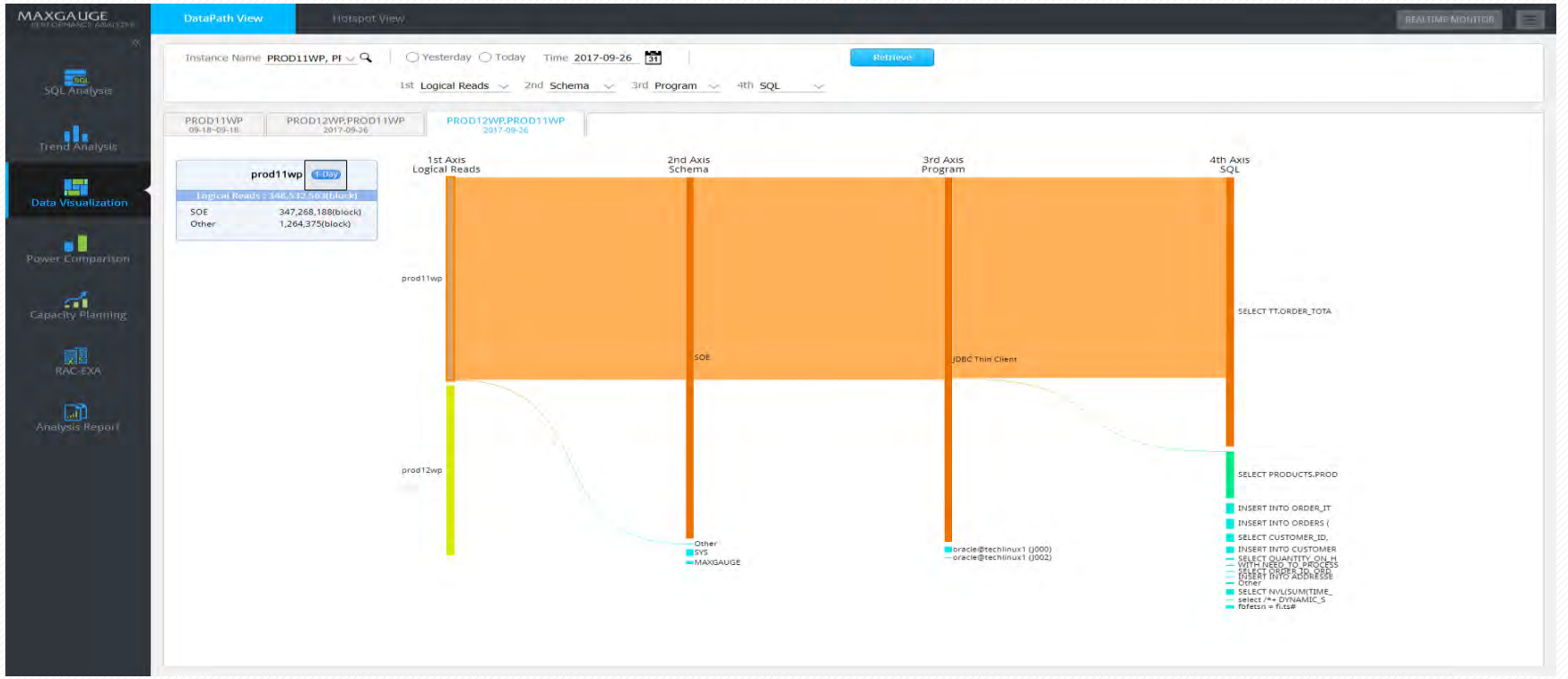
✓ 이 축이 길다는 것은 그 만큼 업무 비중도가 높다는 것을 뜻합니다.

✓ 위 의 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

Top-SQL Comparison 1:M 조회 화면



✓ 1차 축으로 부터 우측을 보면 순서대로 Schema, Program, SQL등을 확인 할 수 있습니다.

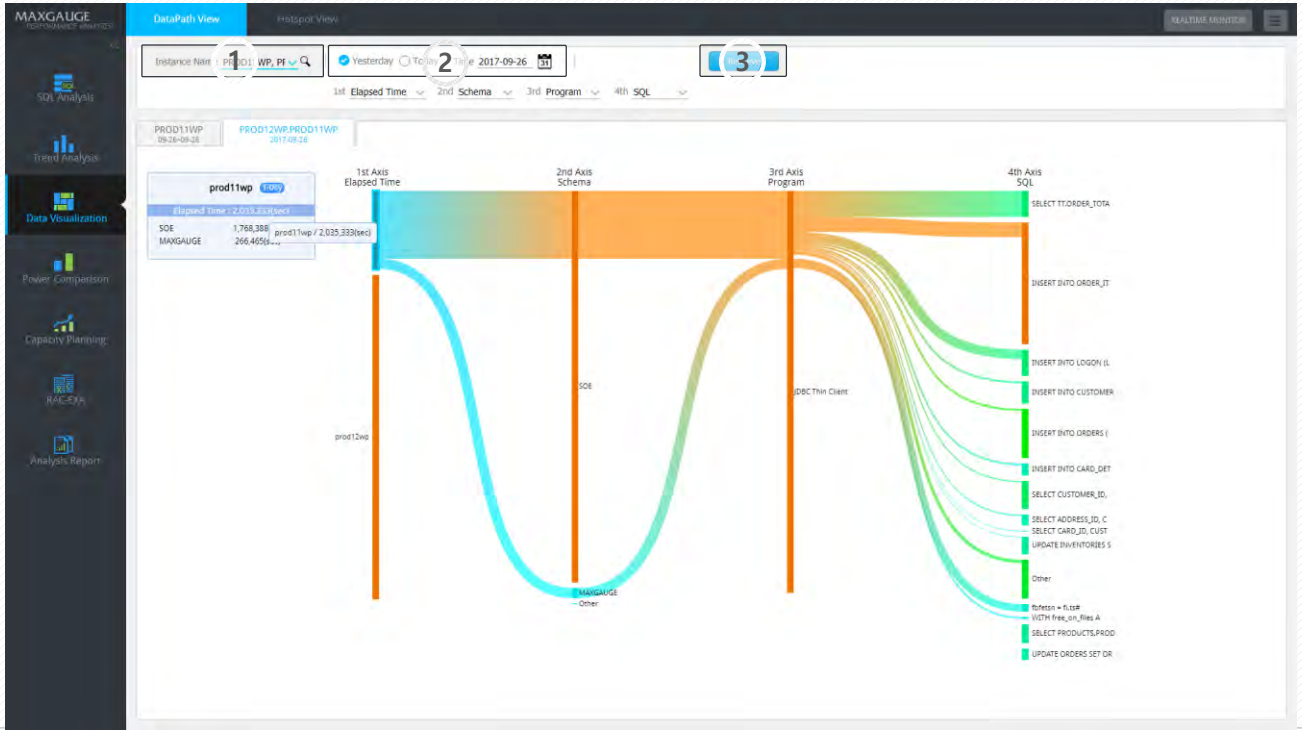
✓ 이 날짜의 상세한 내용을 확인 하려면 우측 상단의 1-Day를 확인 합니다.

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 View검색 화면



✓ 시나리오

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

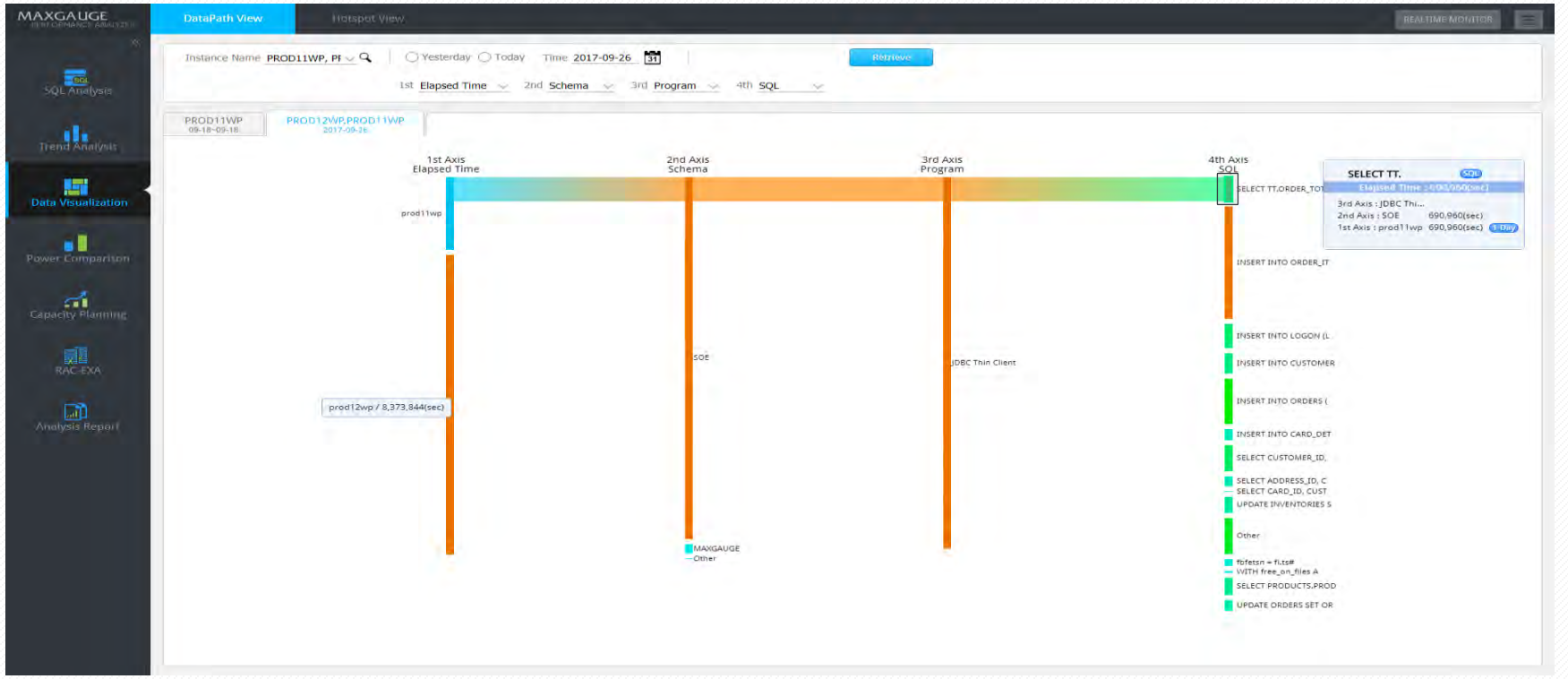
✓ 목표

다른 노드에서 실행 되면서 gc 이벤트를 발생 시키는 SQL을 확인. /
노드 별, SQL 의 수행 비중을 쉽게 확인.

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.**

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

» Top-SQL Comparison 1:M 조회 화면



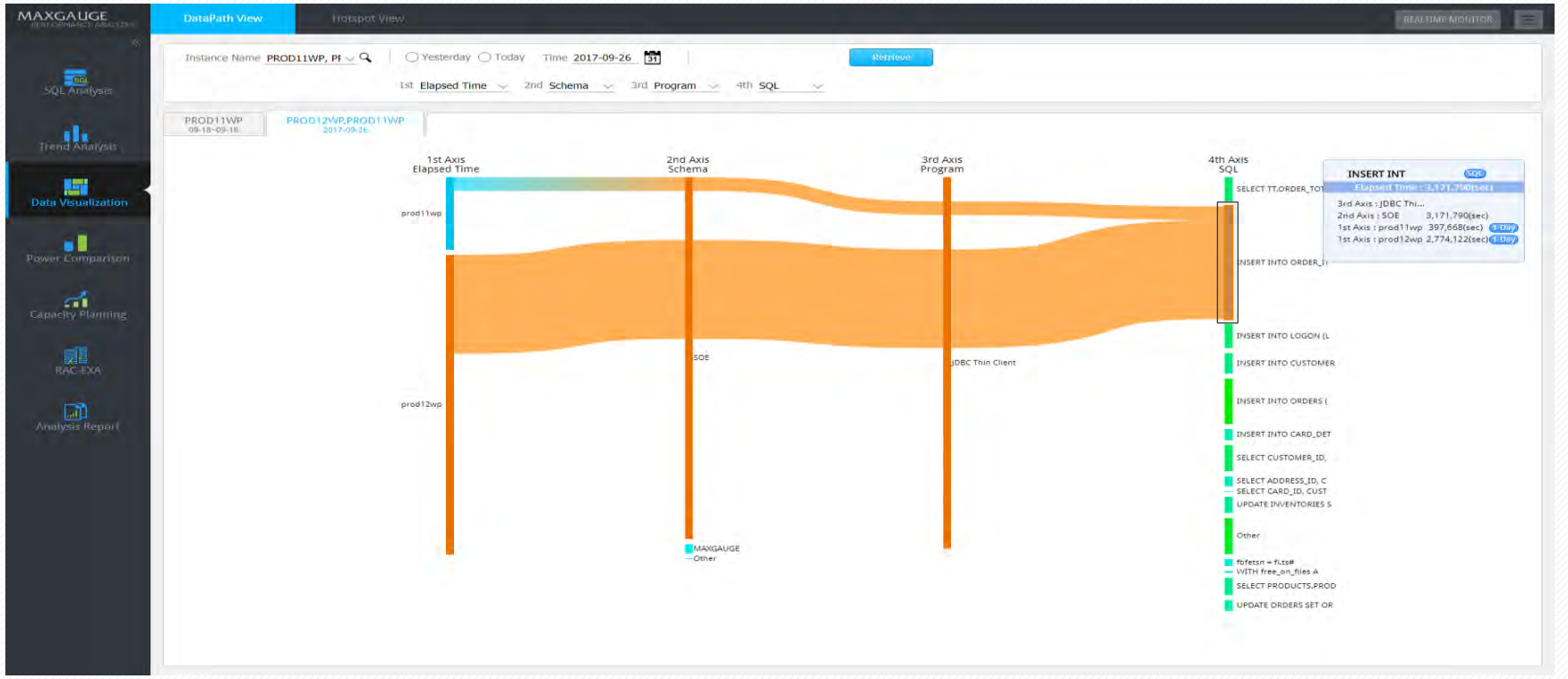
✓ 4차 축의 SQL 들 중에서 선택했을 때 위 그림처럼 축 전체가 하나의 노드만 바라봅니다.

✓ 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?"

» Top-SQL Comparison 1:M 조회 화면



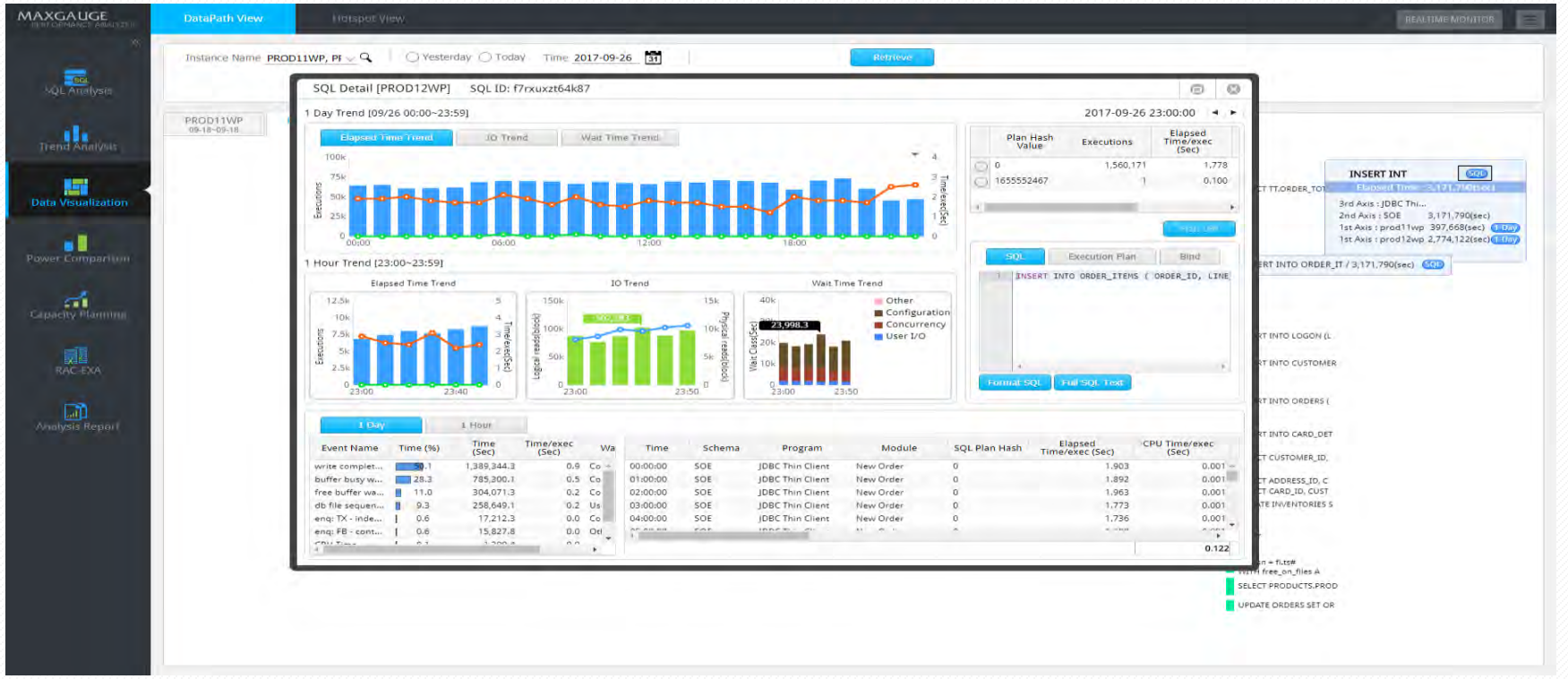
✓ 4차 축의 다음 SQL을 선택 했을 때 갈라져 보인다면, 이는 여러 노드에서 실행 되는 SQL을 뜻합니다.

✓ 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?"

» Top-SQL Comparison 1:M 조회 화면



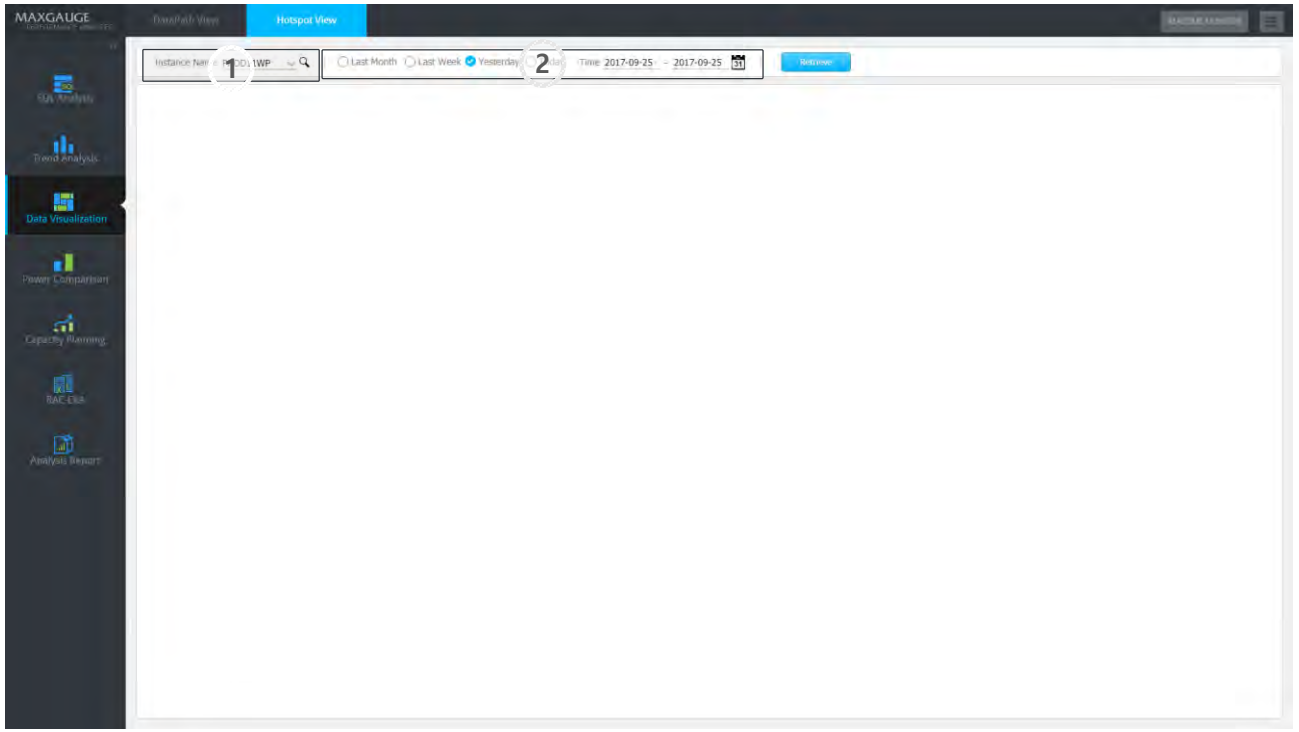
✓ 해당 SQL의 상세한 내용을 확인 하려면 우측 상단의 SQL 버튼을 선택 합니다.

✓ 해당 SQL의 SQL detail과 연계 됩니다.

Hotspot View의 활용

Hotspot View의 사용 방법

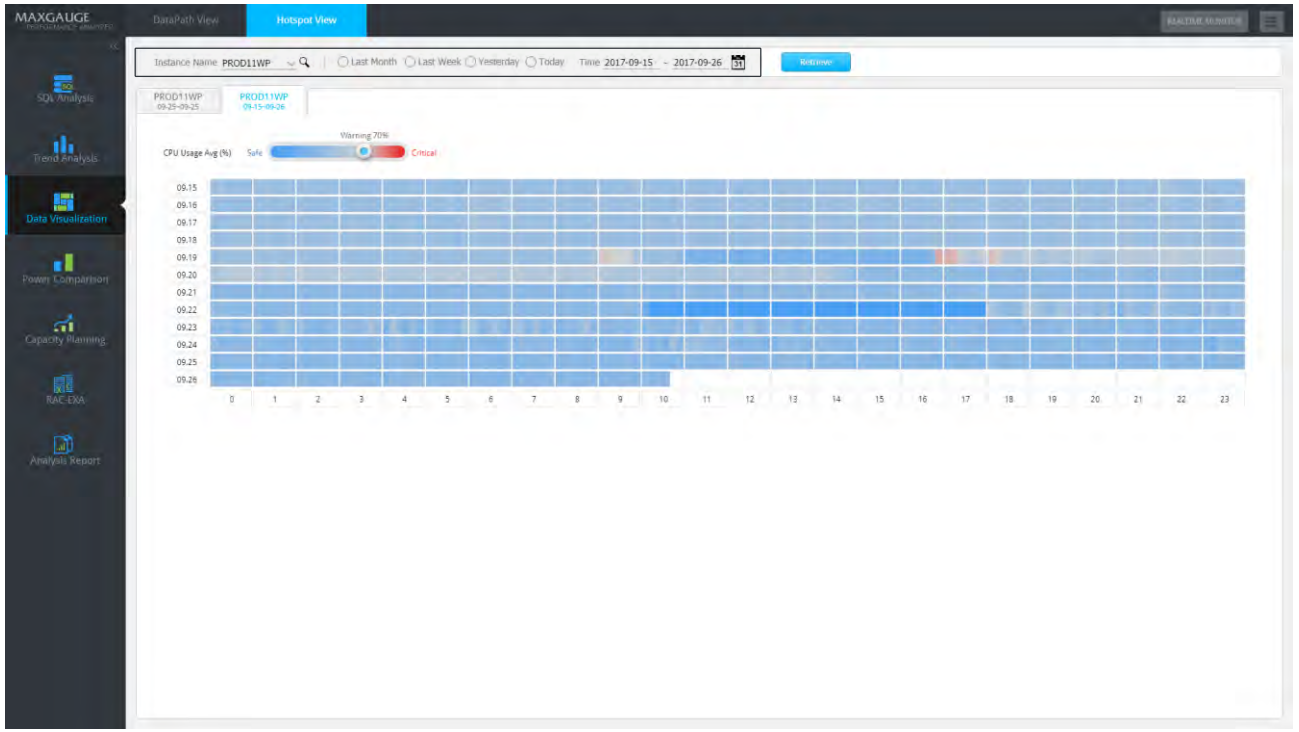
Hotspotview 검색화면



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 결과 화면

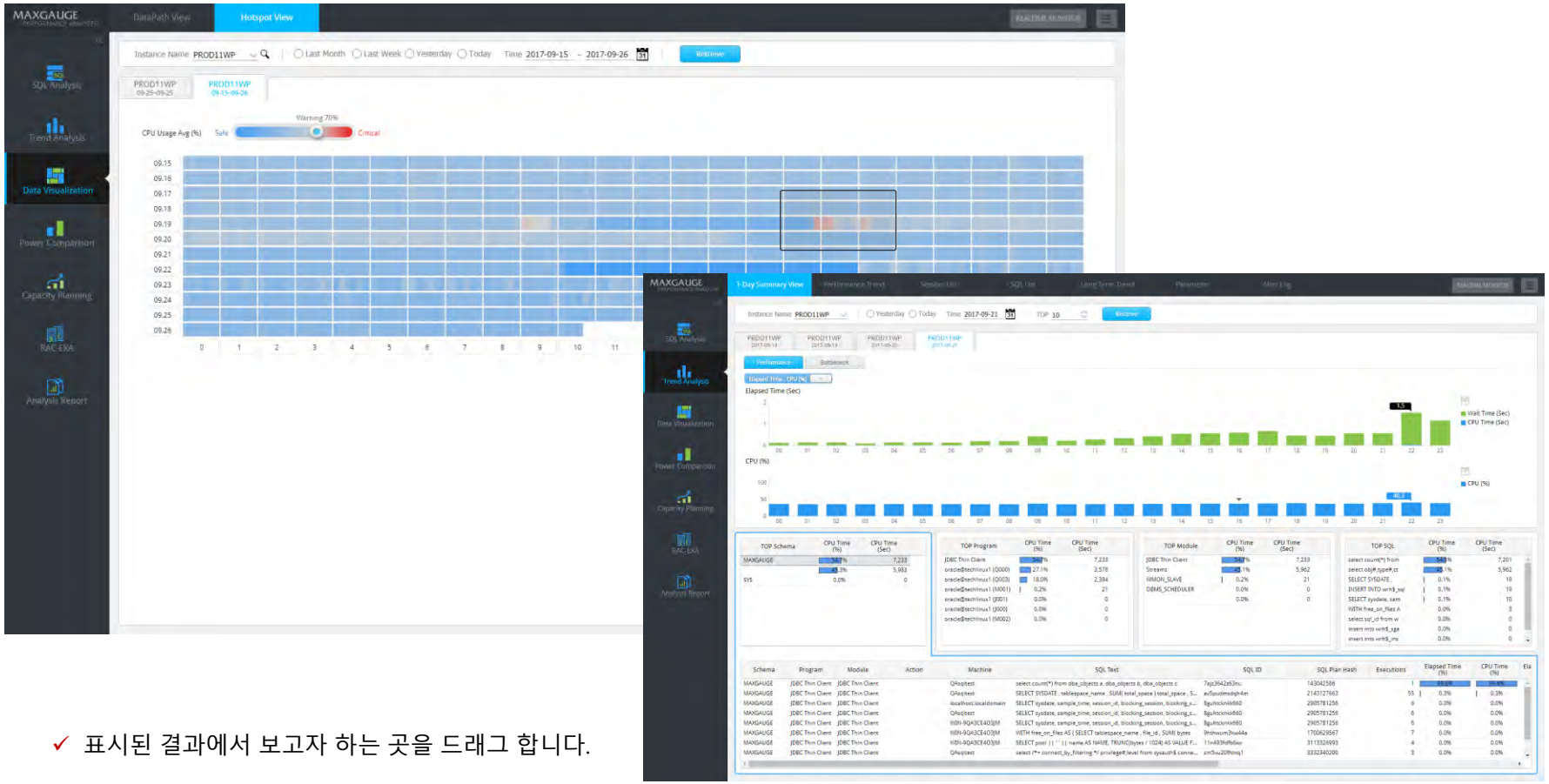


✓ 검색 조건에서 원하는 instance 와 기간을 선택합니다.

✓ 선택 된 Instance 를 기준으로 일자와 10분 단위 시간 별 CPU 사용량 분석이 가능합니다.

Hotspot View의 사용 방법

» Hotspot view 검색 결과 화면

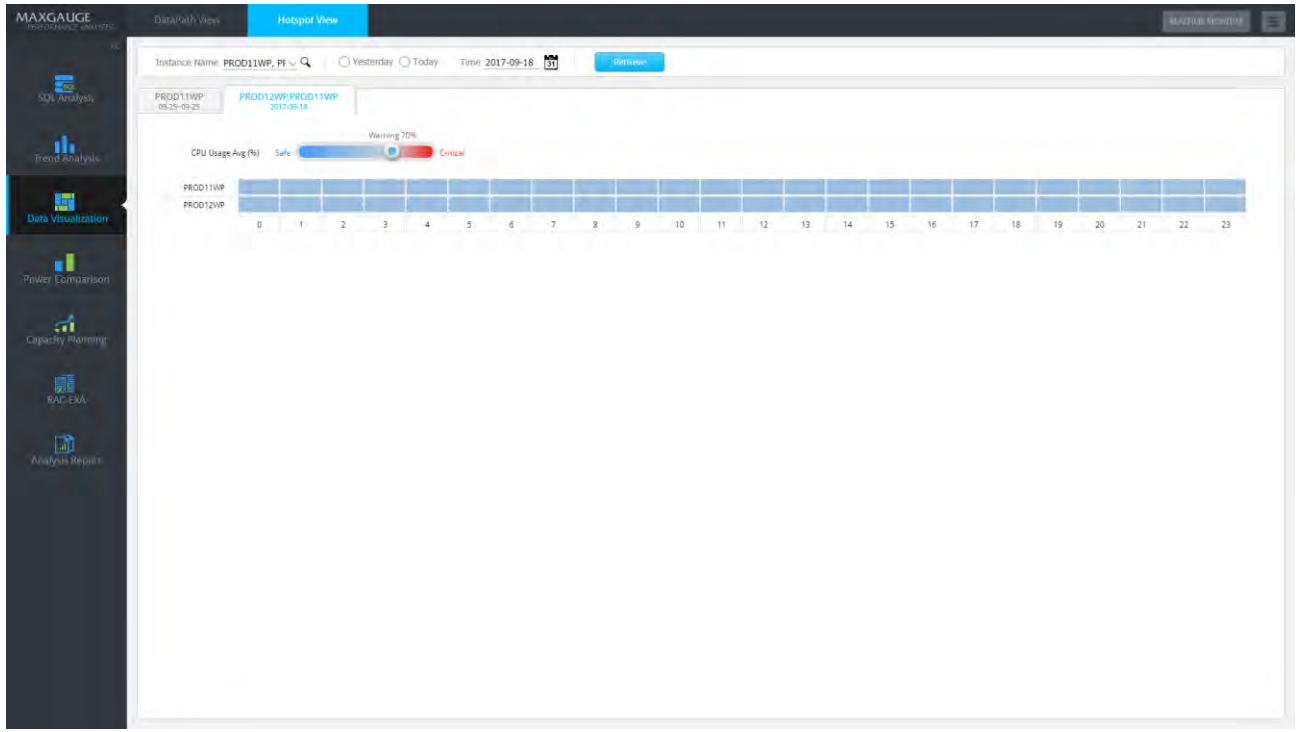


✓ 표시된 결과에서 보고자 하는 곳을 드래그 합니다.

✓ 선택 된 부분의 날짜와 시간대의 1-Day Summary View 로 연계 됩니다.

Hotspot View의 사용 방법

» Hotspotview 검색 결과 화면

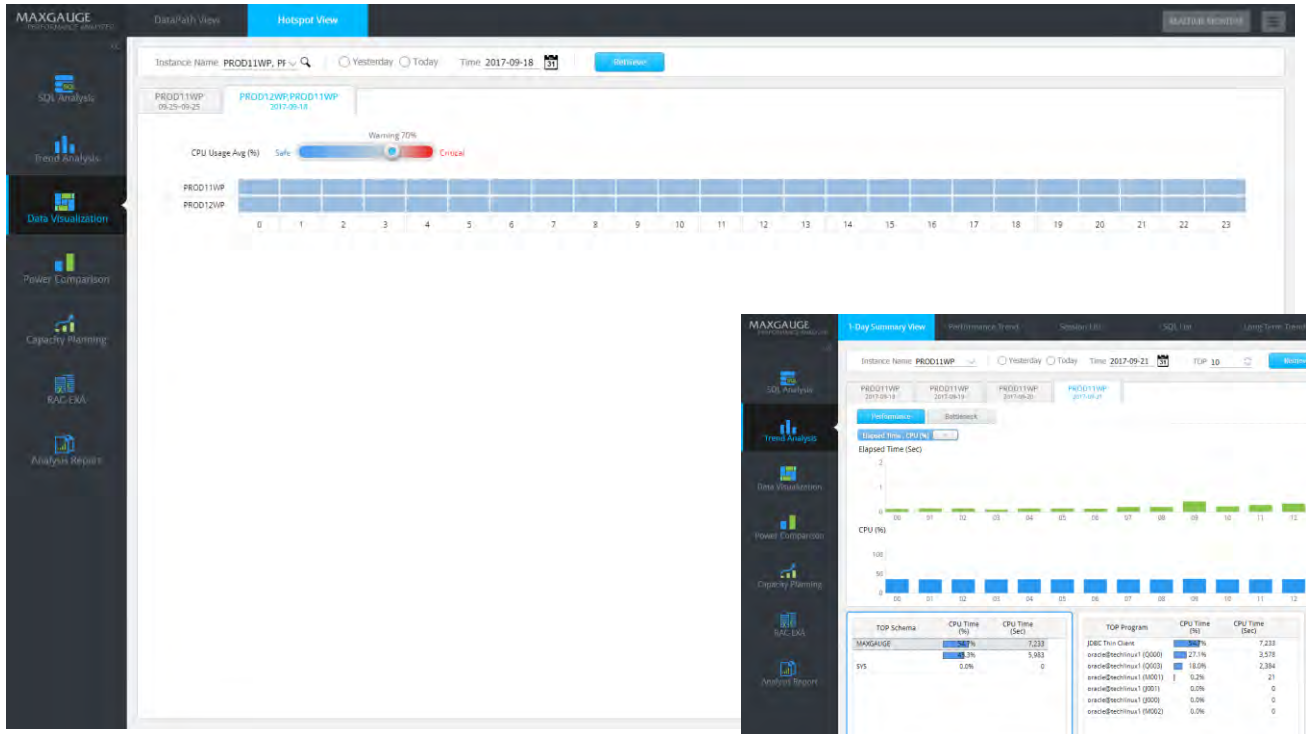


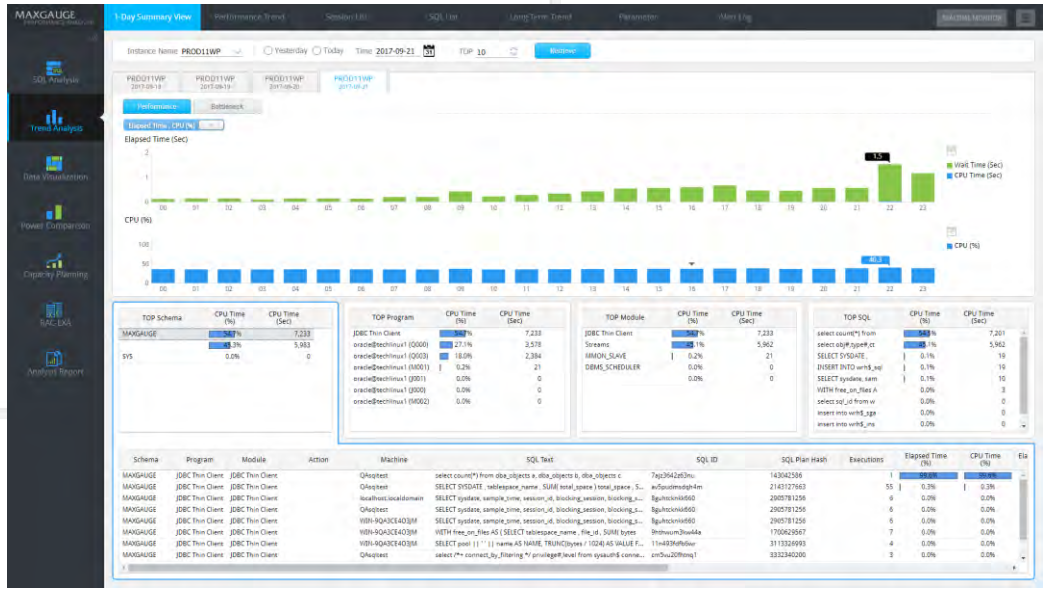
✓ 검색 조건에서 원하는 instance 들과 날짜 선택합니다. Multi Instance의 경우 하루치의 내용만 비교 가능 합니다.

✓ 선택 된 Instance 를 기준으로 일자와 10분 단위 시간 별 CPU 사용량 분석이 가능합니다.

Hotspot View의 사용 방법

» Hotspot view 검색 결과 화면





✓ 표시된 결과에서 보고자 하는 곳을 드래그 합니다.

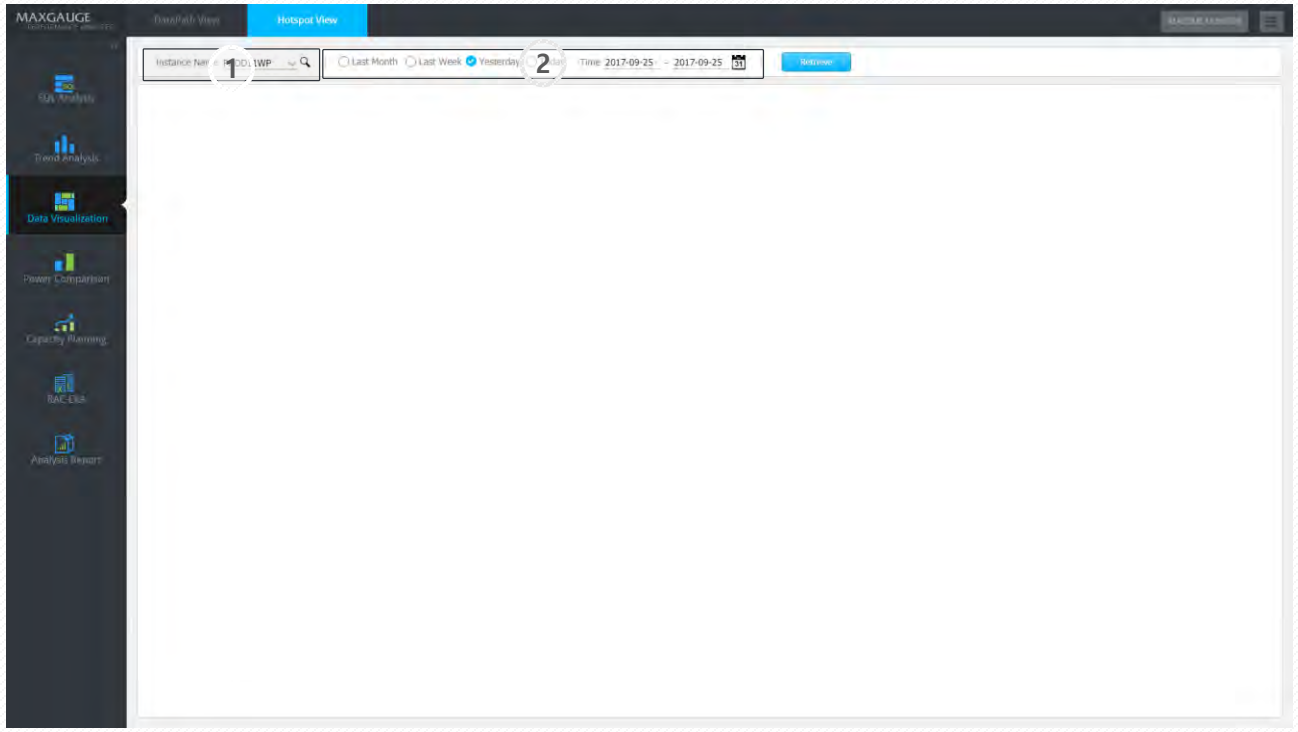
✓ 선택 된 부분의 날짜와 시간대의 1-Day Summary View 로 연계 됩니다.

실전 분석 사례 Case 1.

장기출장 기간 동안 언제 부하가 발생 했고, 원인은 무엇인지 확인 하고 싶어요.

"언제 부하가 발생 했죠?"

Hotspotview 검색화면



✓ **시나리오**

INSTANCE	: PROD11WP
대상 일자	: 2017년 9월 18일 ~ 26일
현재상황	: 담당자가 약 열흘 정도 자리를 비운 뒤 복귀 했을 때, 그 사이 리소스 관련 부하 발생.

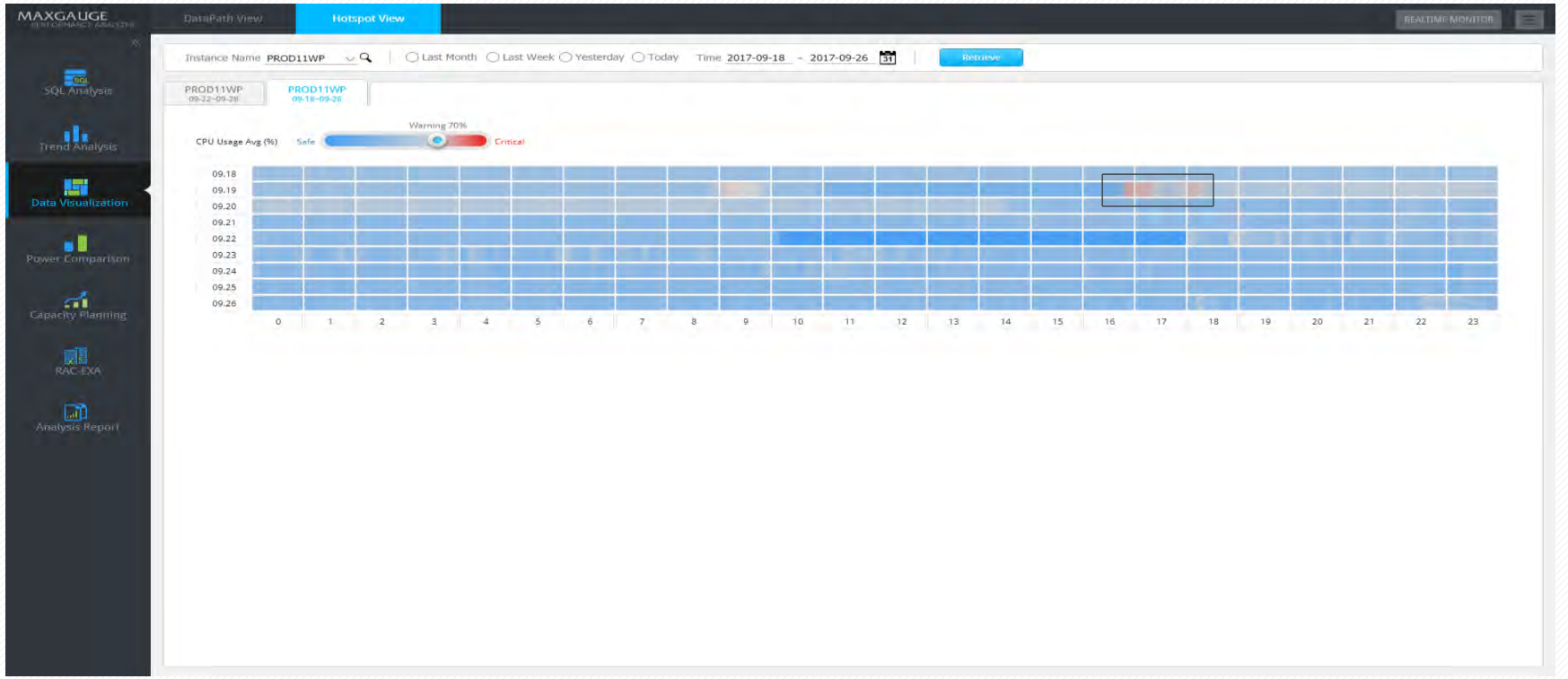
✓ **목표**

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

1. 관제 대상 Instance (PROD11WP)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 결과 화면



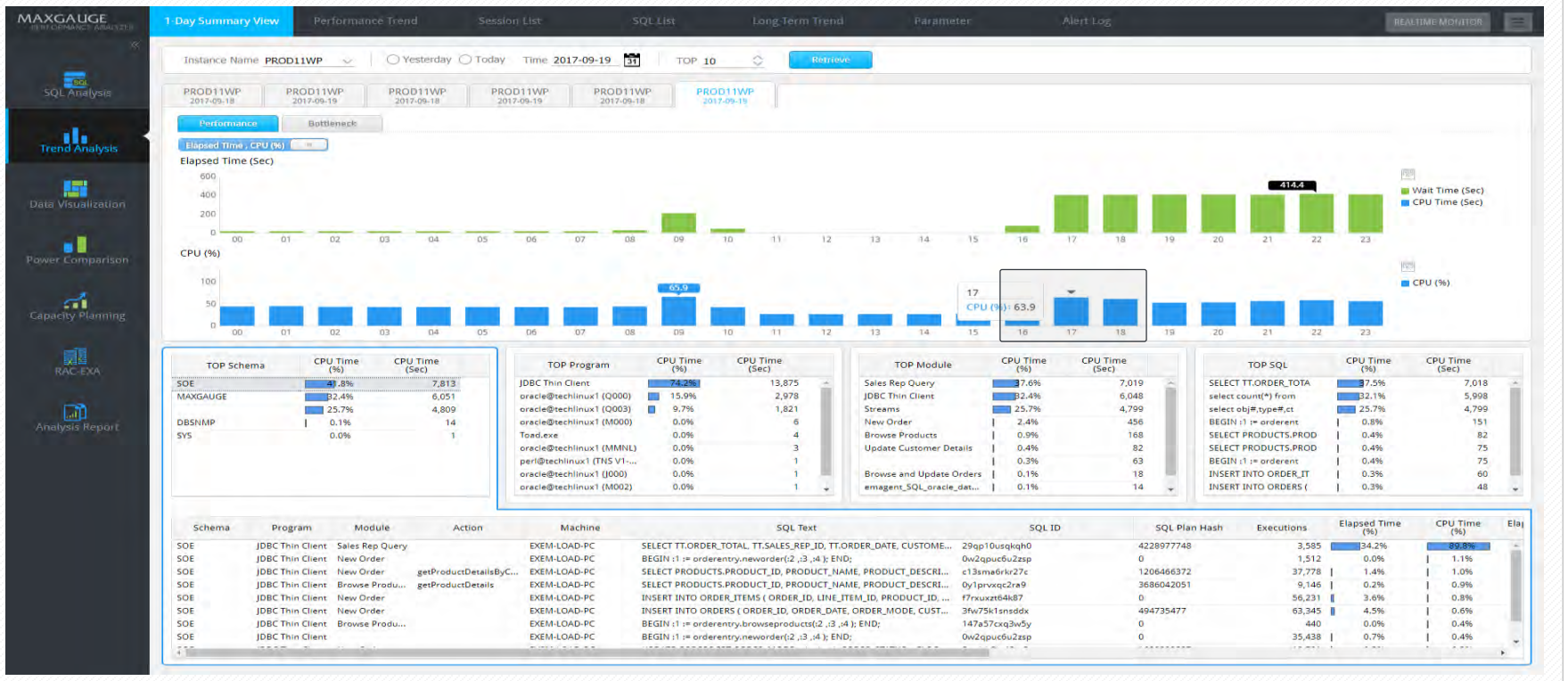
✓ 선택한 기간에 해당하는 리소스 부하 정보를 일자는 행으로, 시간은 열로 표시 합니다.

✓ 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 View 로 연계 됩니다.

"언제 부하가 발생 했죠?"

» Hotspot view 검색 결과 화면

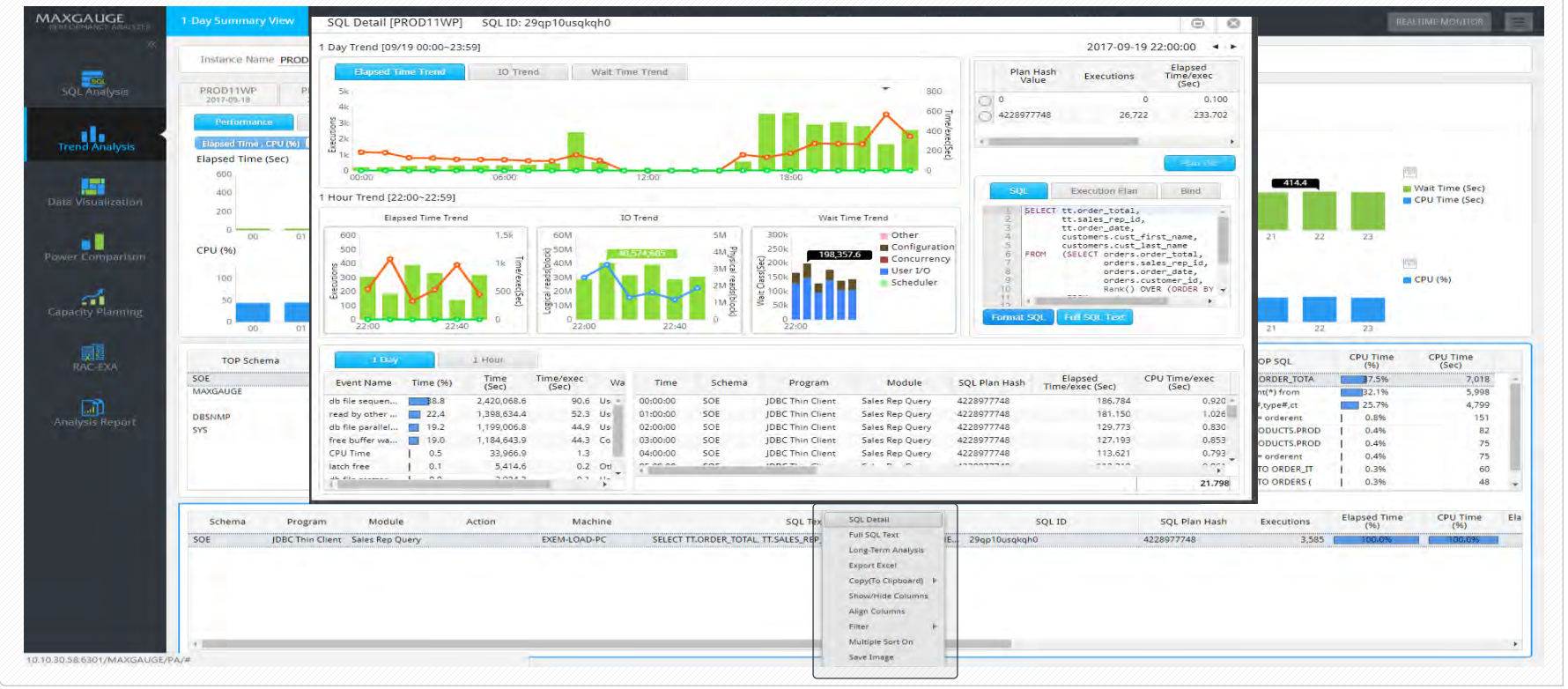


✓ 연계된 1-Day Summary View 화면에서 확인한 결과 63.9%의 사용률을 보였던 것으로 분석 됩니다.

✓ 화면 하단에서 해당 시간대에서 실행된 정보들을 확인 할 수 있습니다.

"언제 부하가 발생 했죠?"

» Hotspot view 검색 결과 화면



✓ 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]

PRODUCT
DOCUMENTATION

Practical Guide

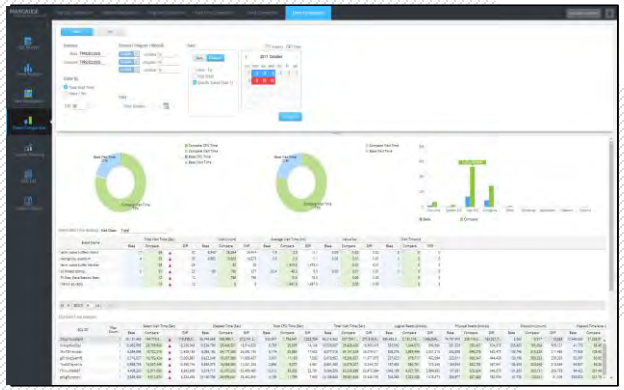
Event Comparison

Contents

Event Comparison?

Event Comparison의 분석 방법론

OWI



Oracle Wait Interface

Oracle은 항상 단서를 남긴다
Oracle Process가 겪는 대기현상

OWI 방법론의 MaxGauge

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

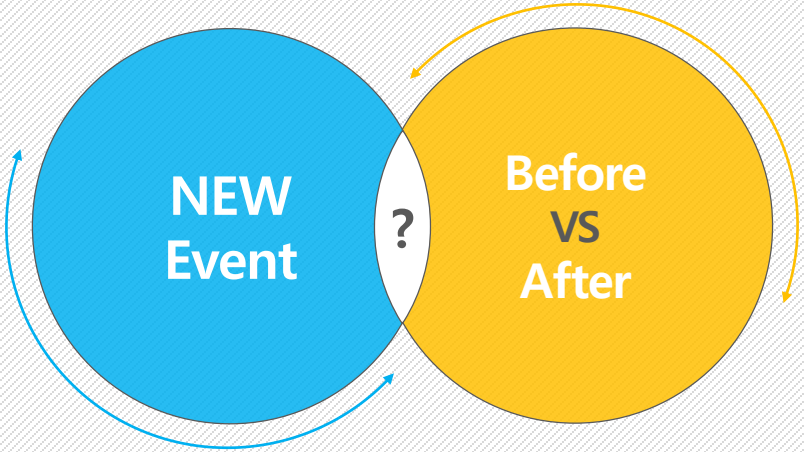
OWI 방법론의 MaxGauge 활용

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 «



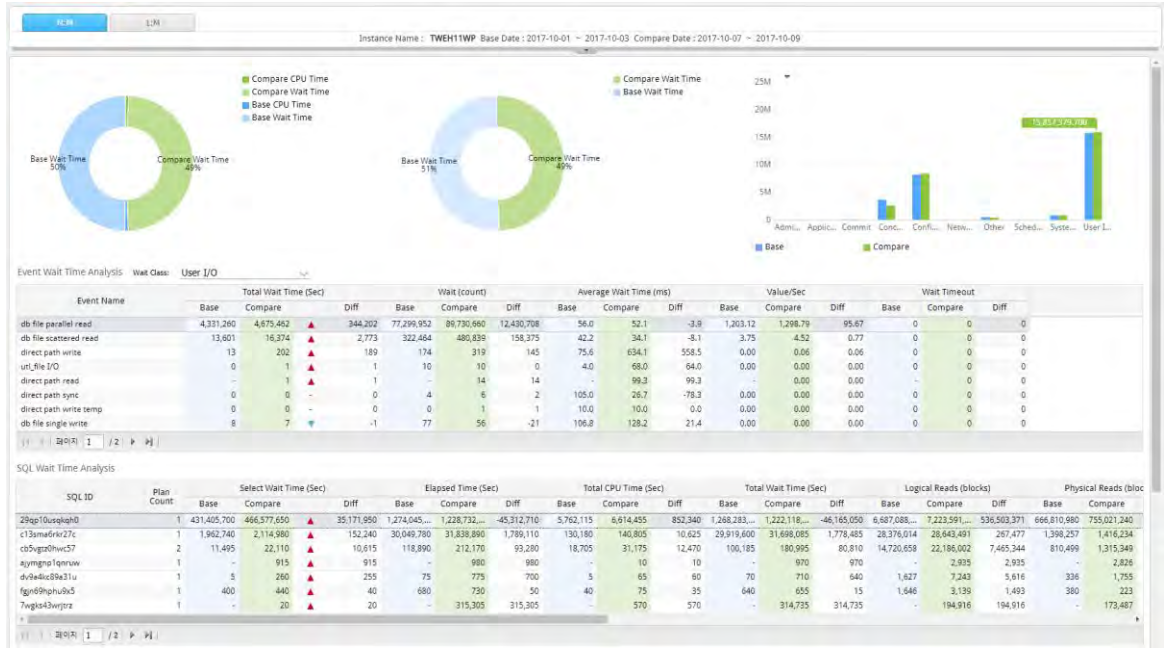
» 기준 날짜 대비 비교 날짜에 발생한 대기 현상을 **Wait Class단위로 분석**할 때.

» 대기 현상을 유발하는 **SQL을 추적**할 때.

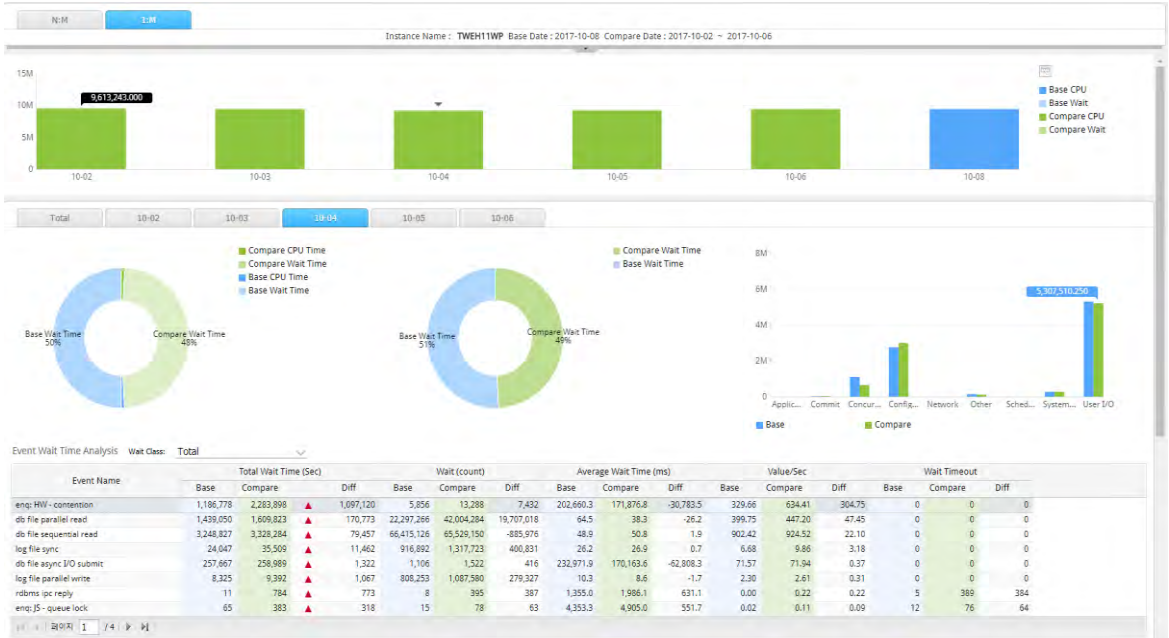
Event Comparison

- 기준 날짜 대비 비교 날짜들의 **Wait Event 발생 현황과 특정 Event로 대기했던 SQL의 비교 기능**이다.
- **Total Wait Time, Wait Class 기준**으로 Base Time과 Compare Time **두 구간에 대한 1:M, N:M** 비교 분석을 지원한다.
- 특정 Event를 선택하면, **선택된 Event로 대기했던 SQL List를 출력**하고, 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.

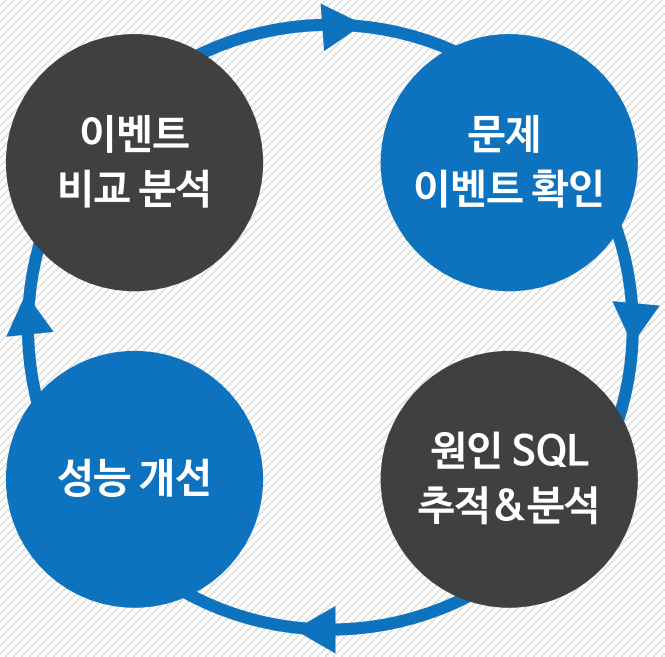
N : M Comparison



1 : M Comparison



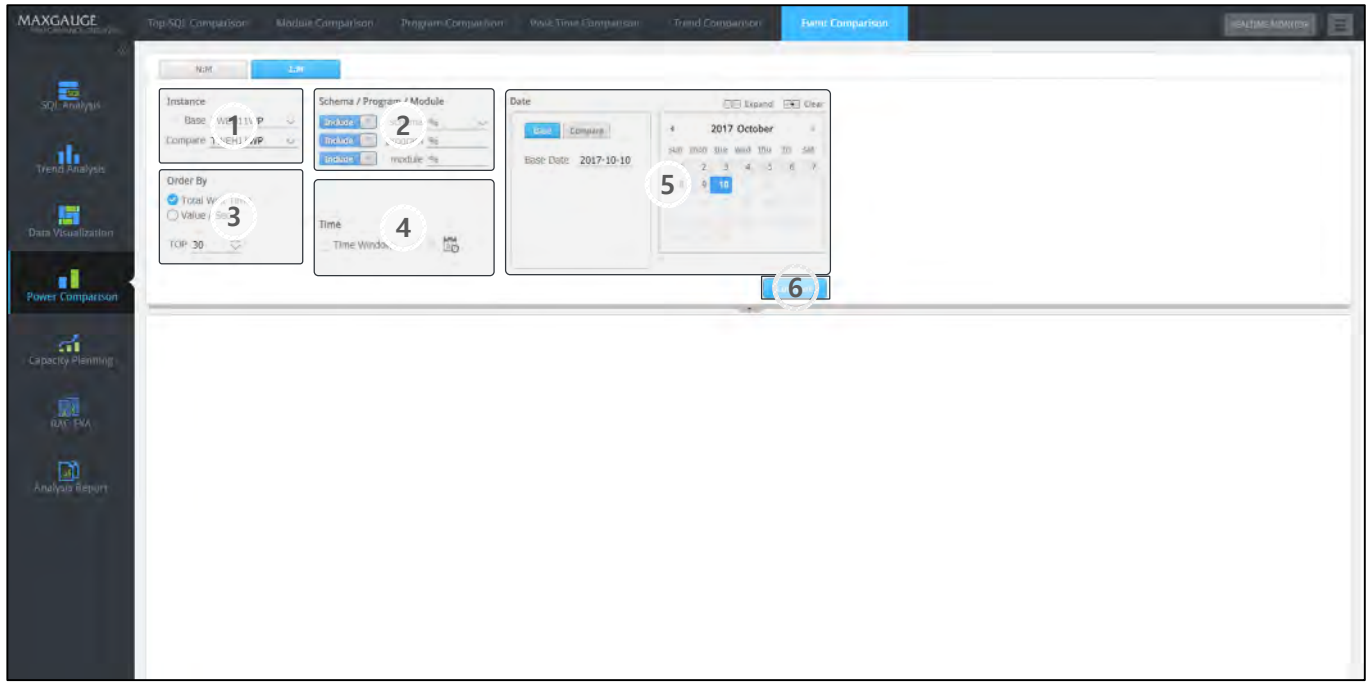
OWI 기반 지속적인 성능 관리를 할 수 있습니다!



Event Comparison의 활용

Event Comparison의 사용 방법 (1:M)

» Event Comparison 1 : M 검색 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.

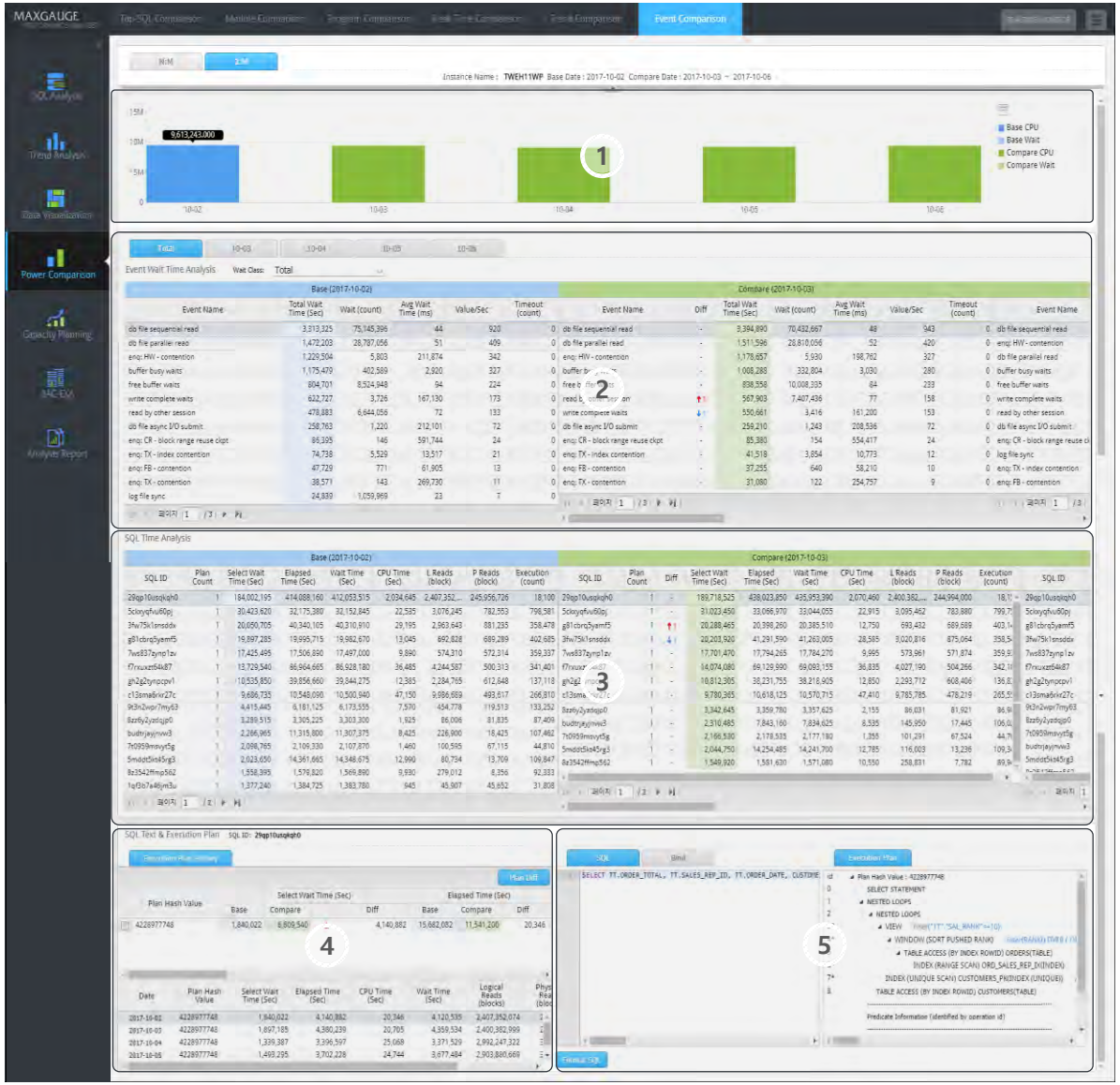
4 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.

5 기준 날짜(**Base Date**)는 하루만 선택되며, 비교 날짜(**Compare Date**)는 Working day, Week, 특정 날짜로 선택 가능합니다.

6 위의 설정한 조건으로 **Event Comparison를 실행**합니다.

Event Comparison의 사용 방법 (1:M)

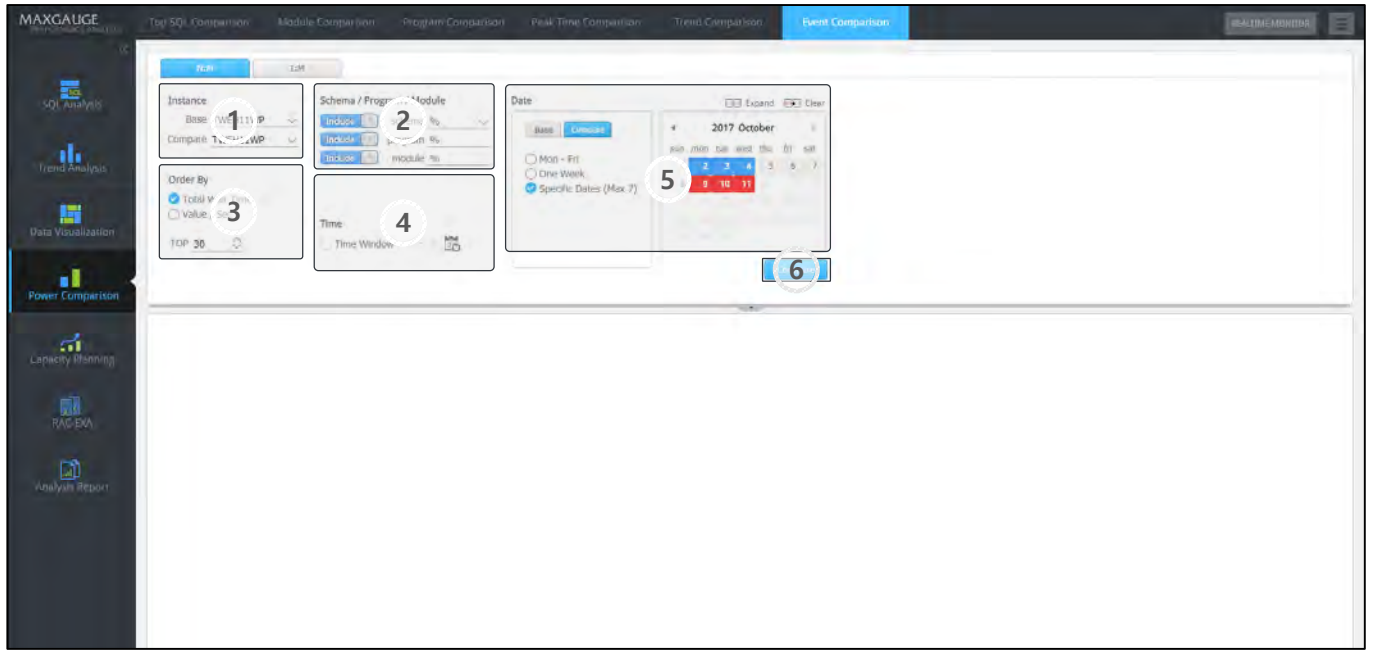
» Event Comparison 1 : M 조회 화면



1. 기준 날짜와 비교 날짜의 **CPU Time**과 **Wait Time**을 일자별 그래프로 제공합니다.
2. 선택한 일자 별(base Date, Compare Date) Total Wait Time 기준으로 **Top Event** 정보를 제공합니다.
3. Event Wait Time Analysis에서 선택된 **Event를 발생 시킨 SQL** 정보를 Total Wait Time 기준으로 제공합니다.
4. SQL Time Analysis에서 선택된 **SQL의 성능** 정보를 Plan Hash Value 단위로 제공하며, 일자 별 수행 이력을 제공합니다.
5. 선택한 SQL에 대한 **SQL Text** 및 **Bind, Execution Plan** 정보를 제공합니다.

Event Comparison의 사용 방법 (N:M)

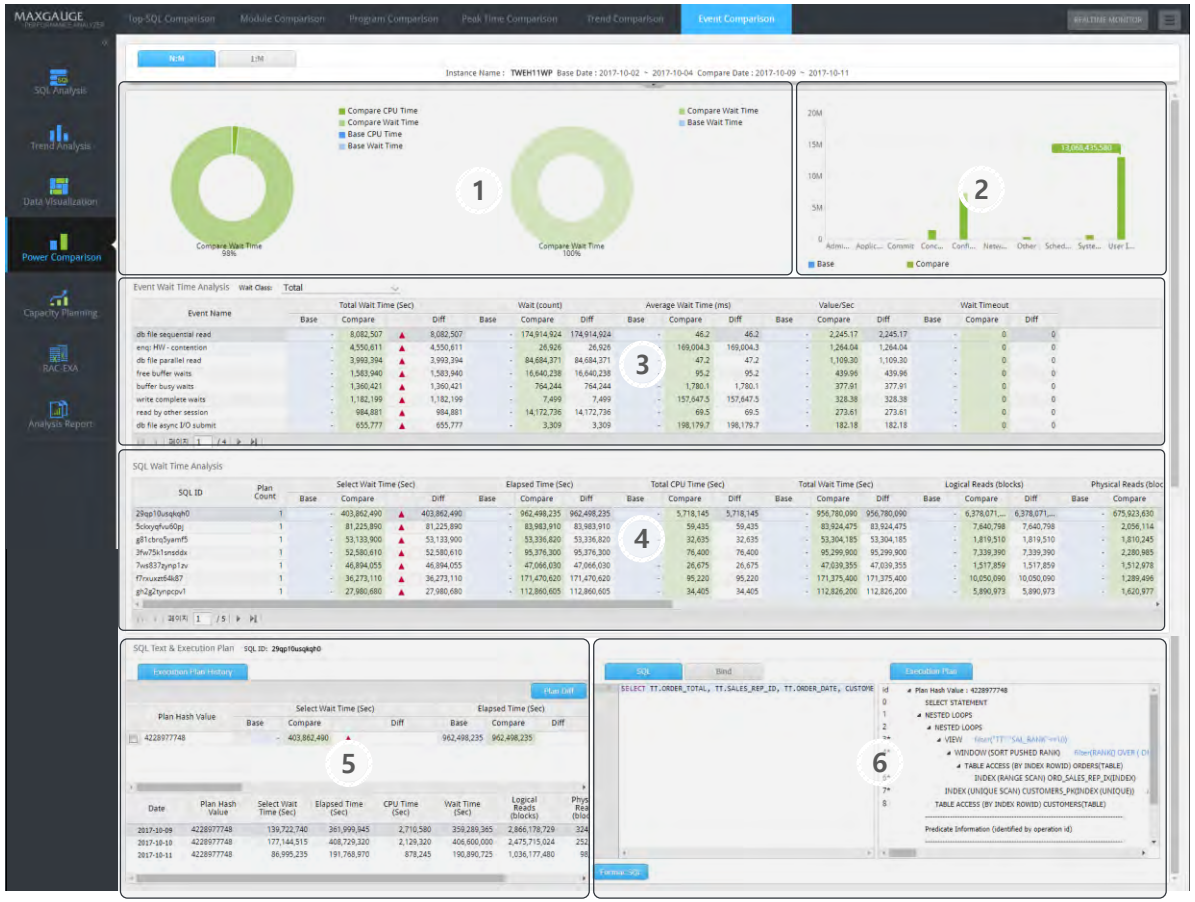
» Event Comparison N : M 검색 화면



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 조회 화면



1. 기준 날짜와 비교 날짜의 **CPU Time**과 **Wait Time**의 비율을 원형 그래프로 나타냅니다.

2. 기준 날짜와 비교 날짜의 **Wait Class 단위 Wait Time**을 그래프로 제공합니다.

3. 기준 날짜와 비교 날짜의 Total Wait Time 기준으로 **Top Event** 정보를 제공합니다.

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

5. SQL Time Analysis에서 선택된 **SQL의 성능** 정보를 Plan Hash Value 단위로 제공하며, 일자 별 수행 이력을 제공합니다.

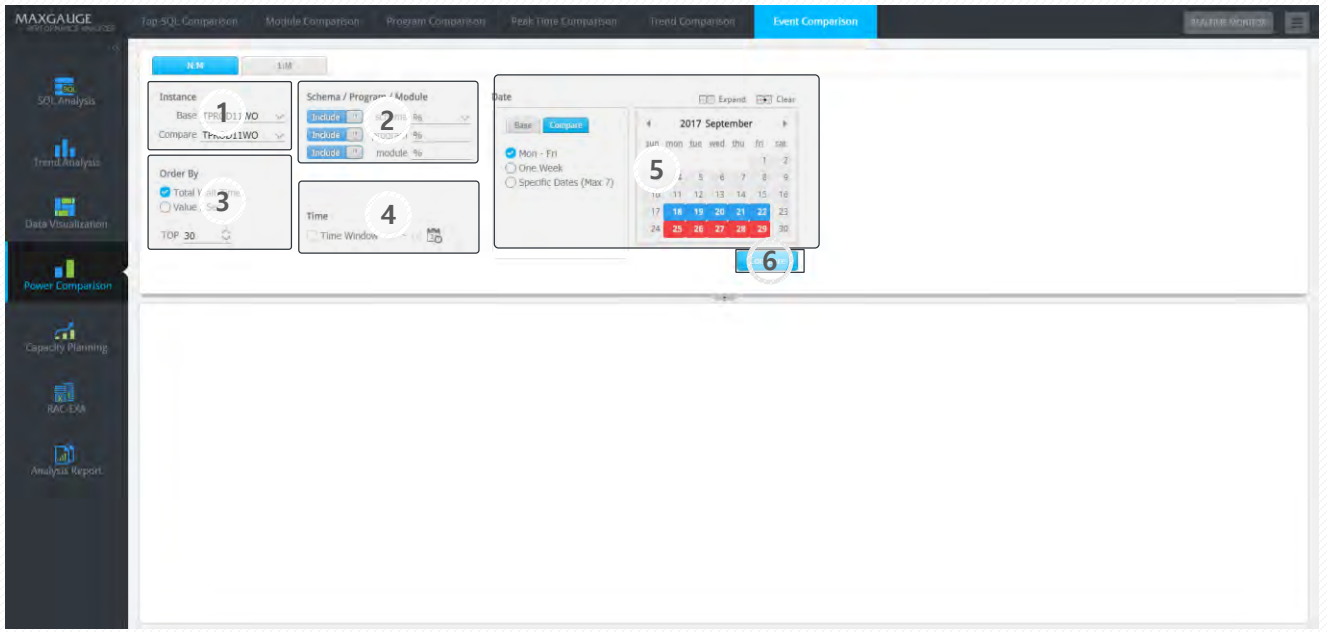
6. 선택한 SQL에 대한 **SQL Text** 및 **Bind, Execution Plan** 정보를 제공합니다.

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)

Event Comparison 검색 화면



✓ 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.

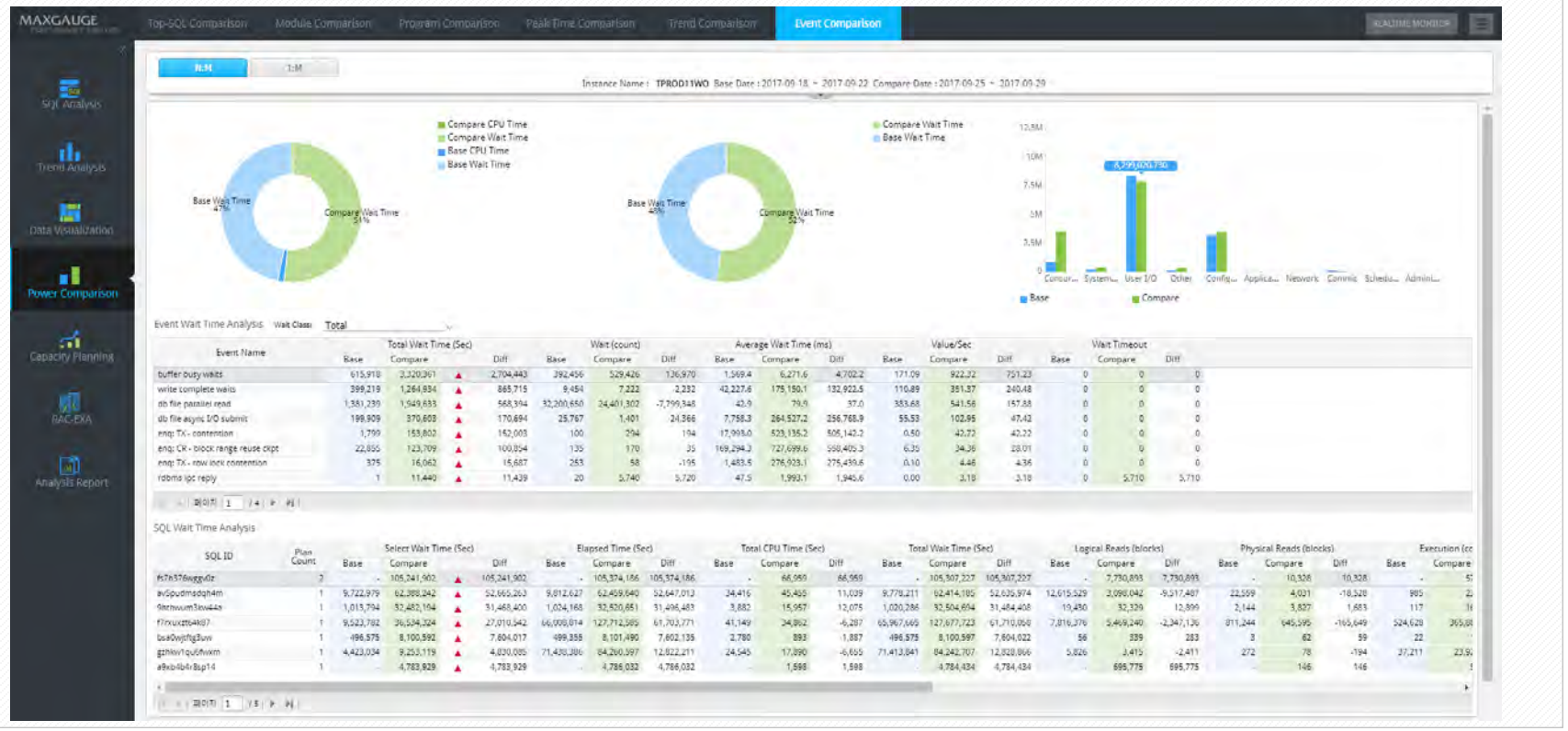
1. 비교 대상 Instance (**TPROD11WO**) 를 선택합니다.
2. Schema (**MAXGAUGE**), Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
3. 정렬 기준을 선택하고, 출력되는 Top Event의 개수(**TOP 30**)를 설정합니다. 정렬 기준은 Total Wait Time과 Value / Sec이 있습니다.
4. 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (**09:00 ~ 18:00**) 을 선택적으로 선택 할 수 있습니다.
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.**

"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 Class단위 분석
   - 발생 대기 이벤트 확인
   - 대기 이벤트 정보 확인
   - 대기 이벤트 발생 SQL 추적
   - SQL 상세 정보 확인

» Event Comparison N:M 조회 화면



✓ 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 이벤트 이며, 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.

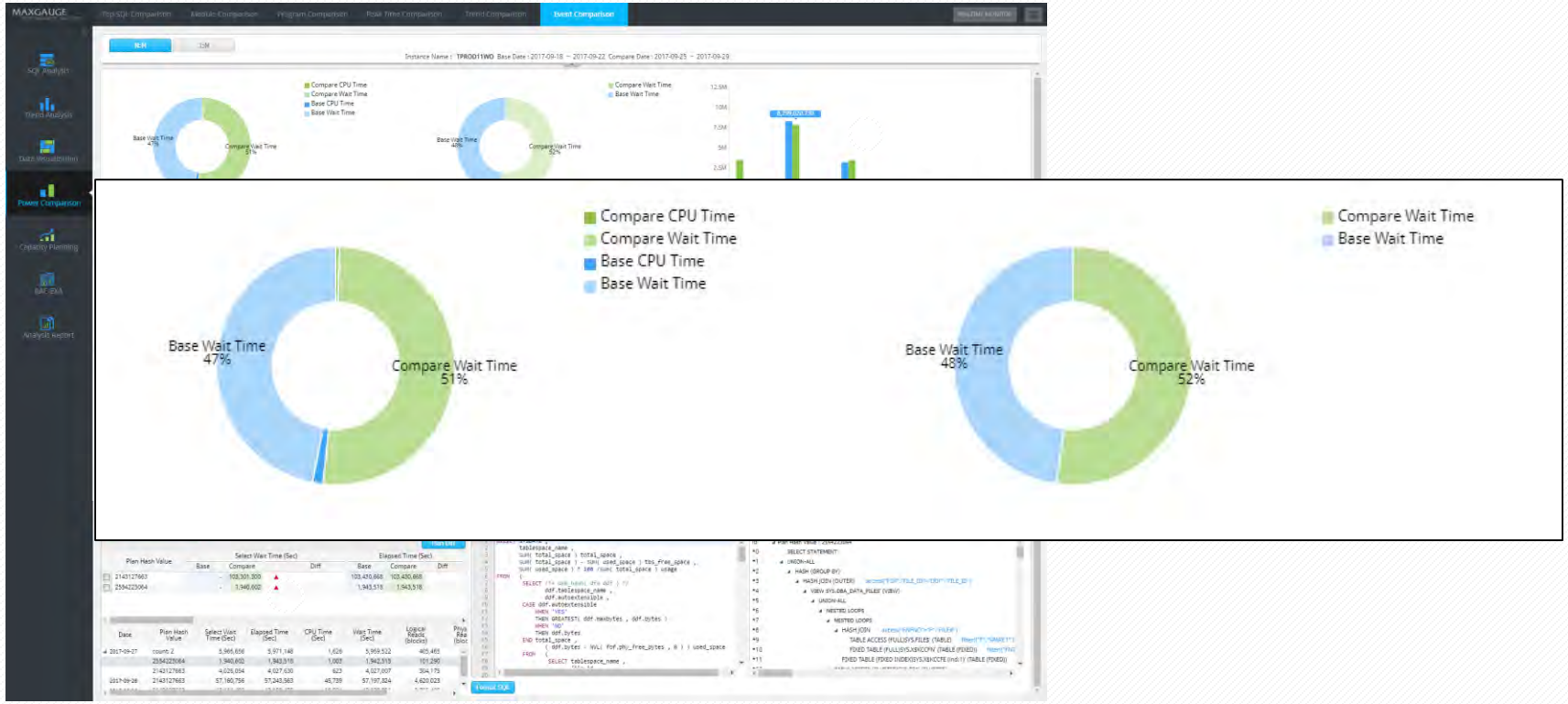
✓ SQL Wait Time Analysis에서 "▲"로 표시된 SQL은 Wait Time이 증가한 SQL입니다.

"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 Class단위 분석
   - 발생 대기 이벤트 확인
   - 대기 이벤트 정보 확인
   - 대기 이벤트 발생 SQL 추적
   - SQL 상세 정보 확인

» Event Comparison N:M 조회화면



✓ 상단 그래프를 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 그래프로 표현합니다.

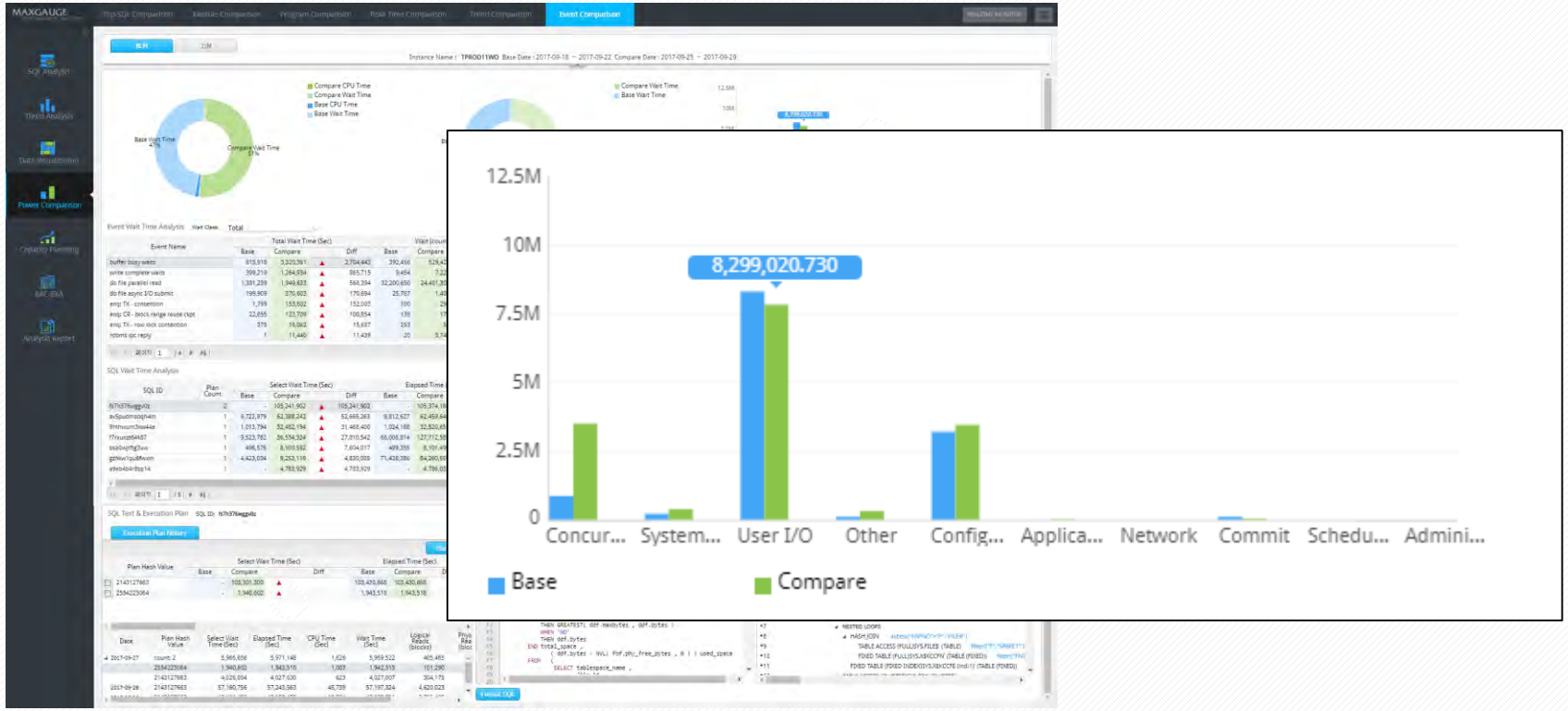
✓ Compare Wait Time이 차지하는 비율이 52%로, Wait Time이 증가했음을 확인할 수 있습니다.

"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 Class단위 분석**
   - 발생 대기 이벤트 확인
   - 대기 이벤트 정보 확인
   - 대기 이벤트 발생 SQL 추적
   - SQL 상세 정보 확인

» Event Comparison N:M 조회화면



✓ Wait Class단위 Base Wait Time과 Compare Wait Time 값을 비교 분석합니다.

✓ 좌측의 파란색은 base Wait Time을, 우측의 초록색은 Compare Wait Time을 나타냅니다.

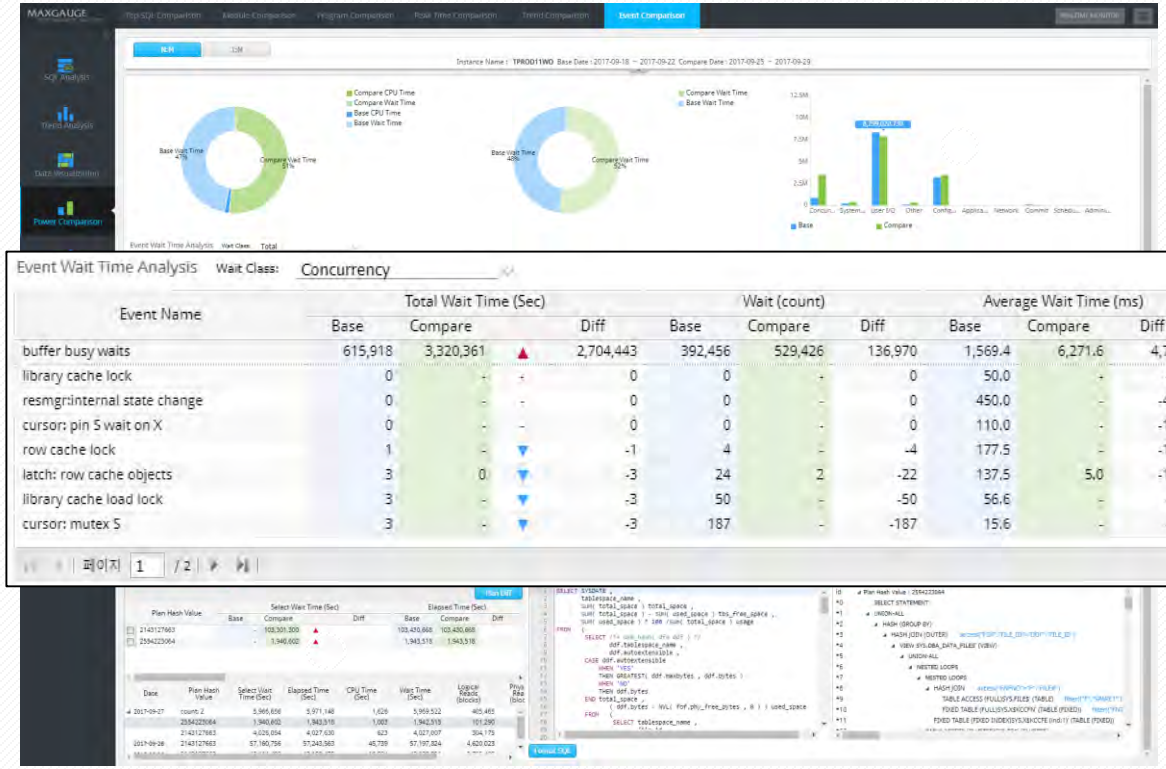
✓ Concurrency Class에서 기준날짜 대비 비교 날짜의 Wait Time이 크게 증가한 것을 확인할 수 있습니다.

"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 Class단위 분석
   - **발생 대기 이벤트 확인**
   - 대기 이벤트 정보 확인
   - 대기 이벤트 발생 SQL 추적
   - SQL 상세 정보 확인

» Event Comparison N:M 조회화면



Event Wait Time Analysis Wait Class: Concurrency

Event Name	Total Wait Time (Sec)				Wait (count)			Average Wait Time (ms)			Value/Sec			Wait Timeout		
	Base	Compare		Diff	Base	Compare	Diff	Base	Compare	Diff	Base	Compare	Diff	Base	Compare	Diff
buffer busy waits	615,918	3,320,361	▲	2,704,443	392,456	529,426	136,970	1,569.4	6,271.6	4,702.2	171.09	922.32	751.23	0	0	0
library cache lock	0	-	-	0	0	-	0	50.0	-	-50.0	0.00	-	0.00	0	-	0
resmgr:internal state change	0	-	-	0	0	-	0	450.0	-	-450.0	0.00	-	0.00	0	-	0
cursor: pin S wait on X	0	-	-	0	0	-	0	110.0	-	-110.0	0.00	-	0.00	0	-	0
row cache lock	1	-	▼	-1	4	-	-4	177.5	-	-177.5	0.00	-	0.00	0	-	0
latch: row cache objects	3	0	▼	-3	24	2	-22	137.5	5.0	-132.5	0.00	0.00	0.00	0	0	0
library cache load lock	3	-	▼	-3	50	-	-50	56.6	-	-56.6	0.00	-	0.00	0	-	0
cursor: mutex S	3	-	▼	-3	187	-	-187	15.6	-	-15.6	0.00	-	0.00	0	-	0

페이지 1 / 2

✓ 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.

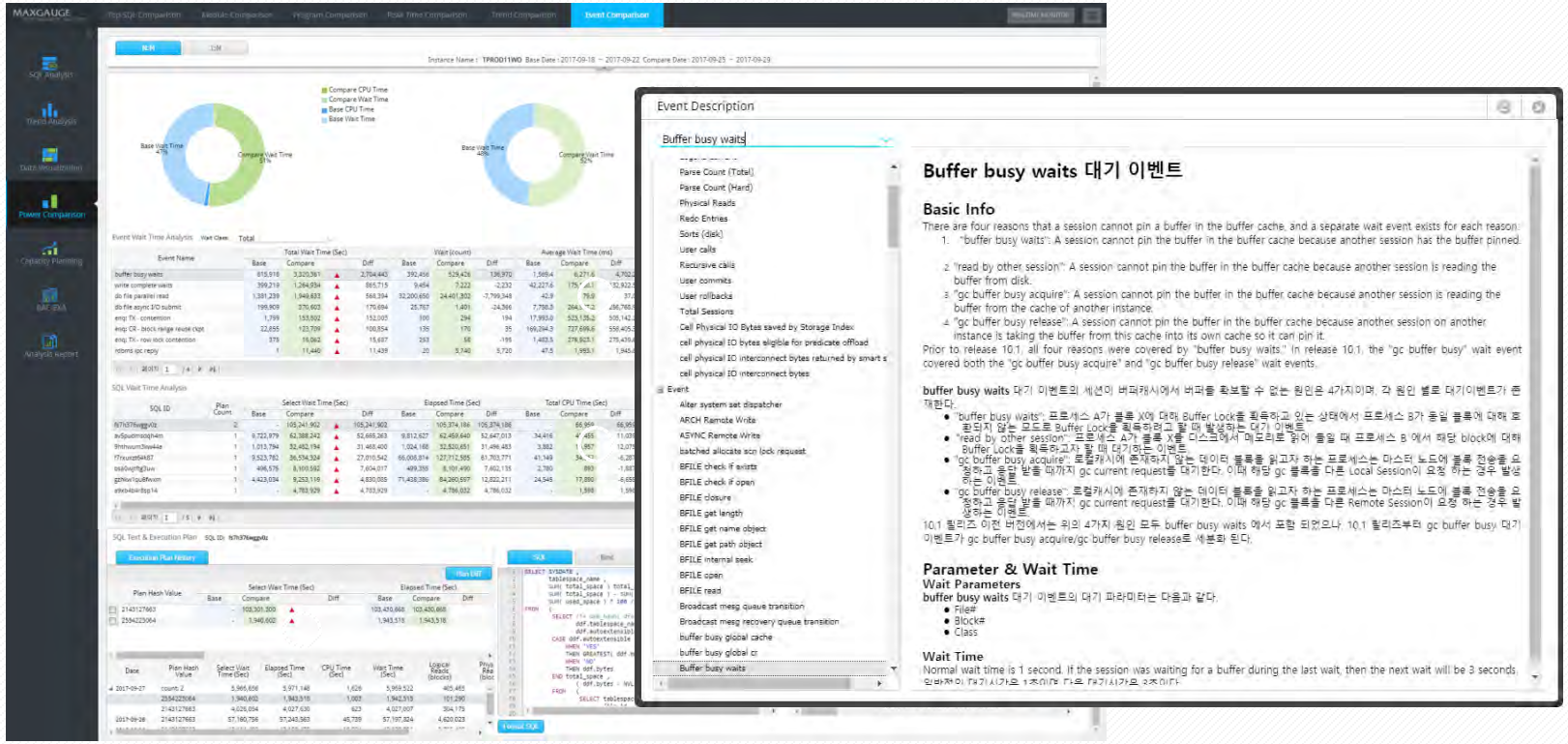
✓ Buffer busy waits 이벤트가 base 구간에 비해서 2,704,443초 증가했음을 확인할 수 있습니다.

"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 Class단위 분석
   - 발생 대기 이벤트 확인
   - **대기 이벤트 정보 확인**
   - 대기 이벤트 발생 SQL 추적
   - SQL 상세 정보 확인

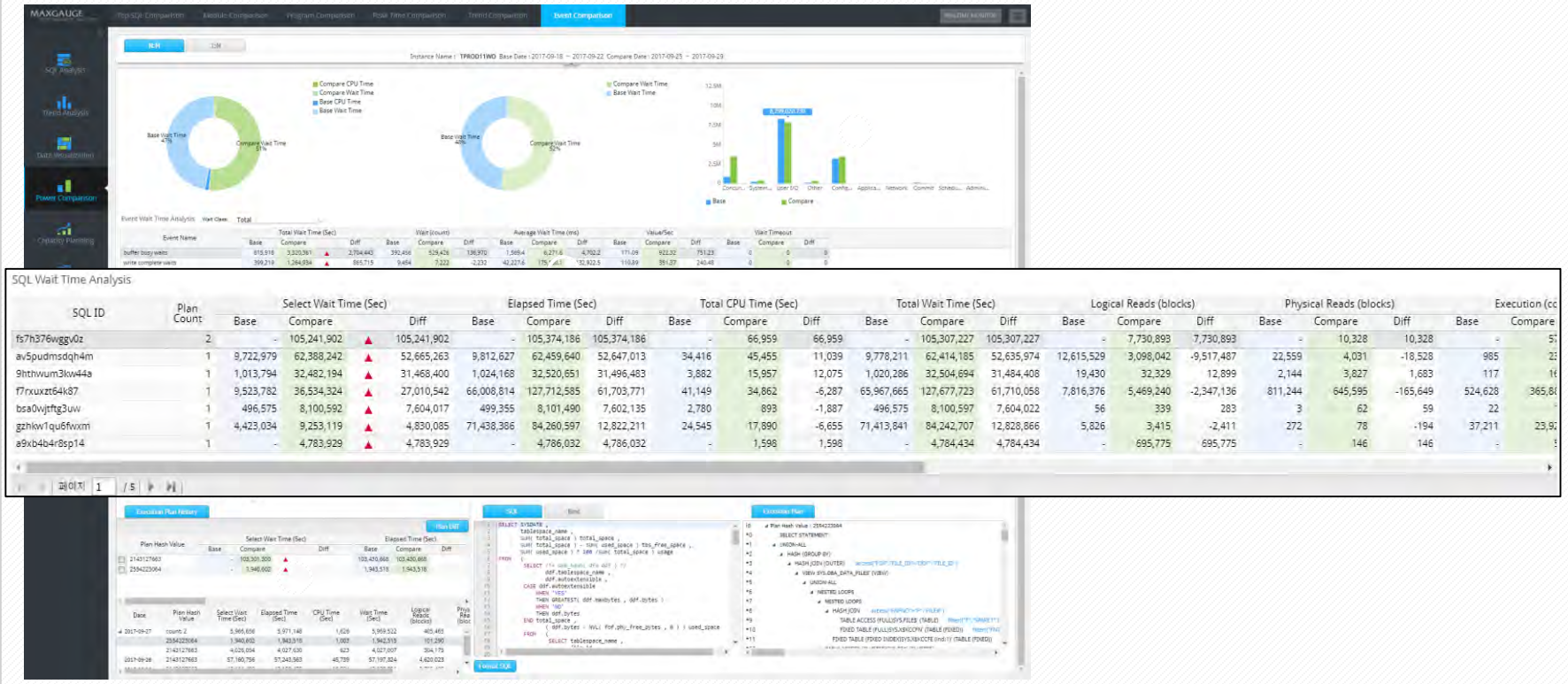
» Event Comparison N:M 조회화면



✓ 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)

» Event Comparison N:M 조회화면



SQL ID	Plan Count	Select Wait Time (Sec) Base	Select Wait Time (Sec) Compare	Select Wait Time (Sec) Diff	Elapsed Time (Sec) Base	Elapsed Time (Sec) Compare	Elapsed Time (Sec) Diff	Total CPU Time (Sec) Base	Total CPU Time (Sec) Compare	Total CPU Time (Sec) Diff	Total Wait Time (Sec) Base	Total Wait Time (Sec) Compare	Total Wait Time (Sec) Diff	Logical Reads (blocks) Base	Logical Reads (blocks) Compare	Logical Reads (blocks) Diff	Physical Reads (blocks) Base	Physical Reads (blocks) Compare	Physical Reads (blocks) Diff	Execution (cc Base	Execution (cc Compare
fs7h376wggv0z	2	-	105,241,902	▲ 105,241,902	-	105,374,186	105,374,186	-	66,959	66,959	-	105,307,227	105,307,227	-	7,730,893	7,730,893	-	10,328	10,328	-	5
av5pudmsdqh4m	1	9,722,979	62,388,242	▲ 52,665,263	9,812,627	62,459,640	52,647,013	34,416	45,455	11,039	9,778,211	62,414,185	52,635,974	12,615,529	3,098,042	-9,517,487	22,559	4,031	-18,528	985	2
9hthwum3kw44a	1	1,013,794	32,482,194	▲ 31,468,400	1,024,168	32,520,651	31,496,483	3,882	15,957	12,075	1,020,286	32,504,694	31,484,408	19,430	32,329	12,899	2,144	3,827	1,683	117	1
f7rxuxzt64k87	1	9,523,782	36,534,324	▲ 27,010,542	66,008,814	127,712,585	61,703,771	41,149	34,862	-6,287	65,967,665	127,677,723	61,710,058	7,816,376	5,469,240	-2,347,136	811,244	645,595	-165,649	524,628	365,8
bsa0wjtftg3uw	1	496,575	8,100,592	▲ 7,604,017	499,355	8,101,490	7,602,135	2,780	893	-1,887	496,575	8,100,597	7,604,022	56	339	283	3	62	59	22	-
gzhkw1qu6fwxm	1	4,423,034	9,253,119	▲ 4,830,085	71,438,386	84,260,597	12,822,211	24,545	17,890	-6,655	71,413,841	84,242,707	12,828,866	5,826	3,415	-2,411	272	78	-194	37,211	23,9
a9xb4b4r8sp14	1	-	4,783,929	▲ 4,783,929	-	4,786,032	4,786,032	-	1,598	1,598	-	4,784,434	4,784,434	-	695,775	695,775	-	146	146	-	5

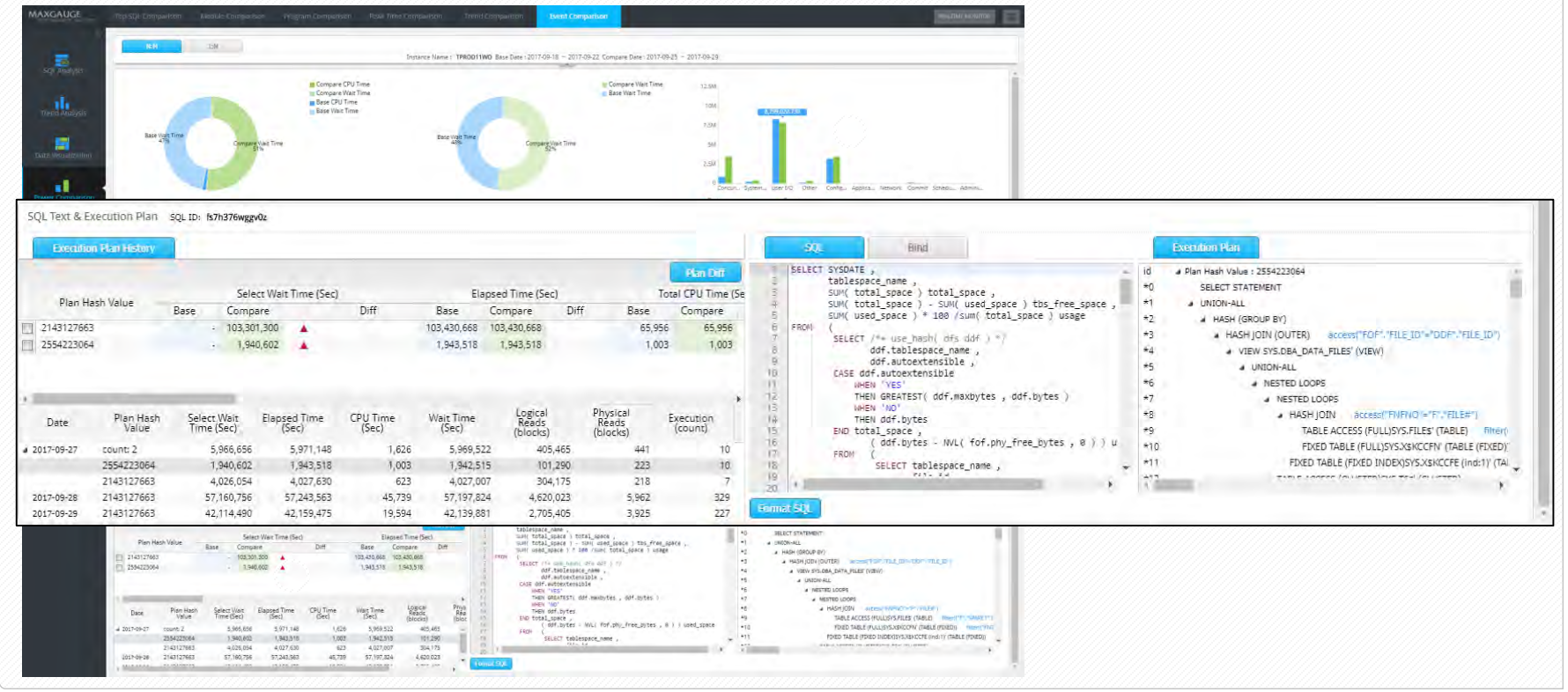
- ✓ SQL Wait Time Analysis에서 buffer busy waits로 대기했던 SQL List를 확인할 수 있습니다.
- ✓ Select Wait Time, Elapsed Time, Total CPU Time, Total Wait Time, Logical Reads, Physical Reads, Execution count등의 정보를 제공합니다.
- ✓ Base구간에 Wait Time이 " – "인 SQL은 Compare Date 기간에 추가된 SQL입니다.
- ✓ "SQL ID : fs7h376wggv0z"이 대부분의 buffer busy waits 대기 현상을 유발하며, 이 SQL로 인해서 기존에 수행되던 SQL들의 buffer busy waits 대기 시간도 증가했음을 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 Class단위 분석
   - 발생 대기 이벤트 확인
   - 대기 이벤트 정보 확인
   - 대기 이벤트 발생 SQL 추적
   - **SQL 상세 정보 확인**

» Event Comparison N:M 조회화면



✓ 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(fs7h376wggv0z)의 성능 점검 및 개선이 필요하다는 것을 확인할 수 있습니다.

실전 분석 사례 Case 2.

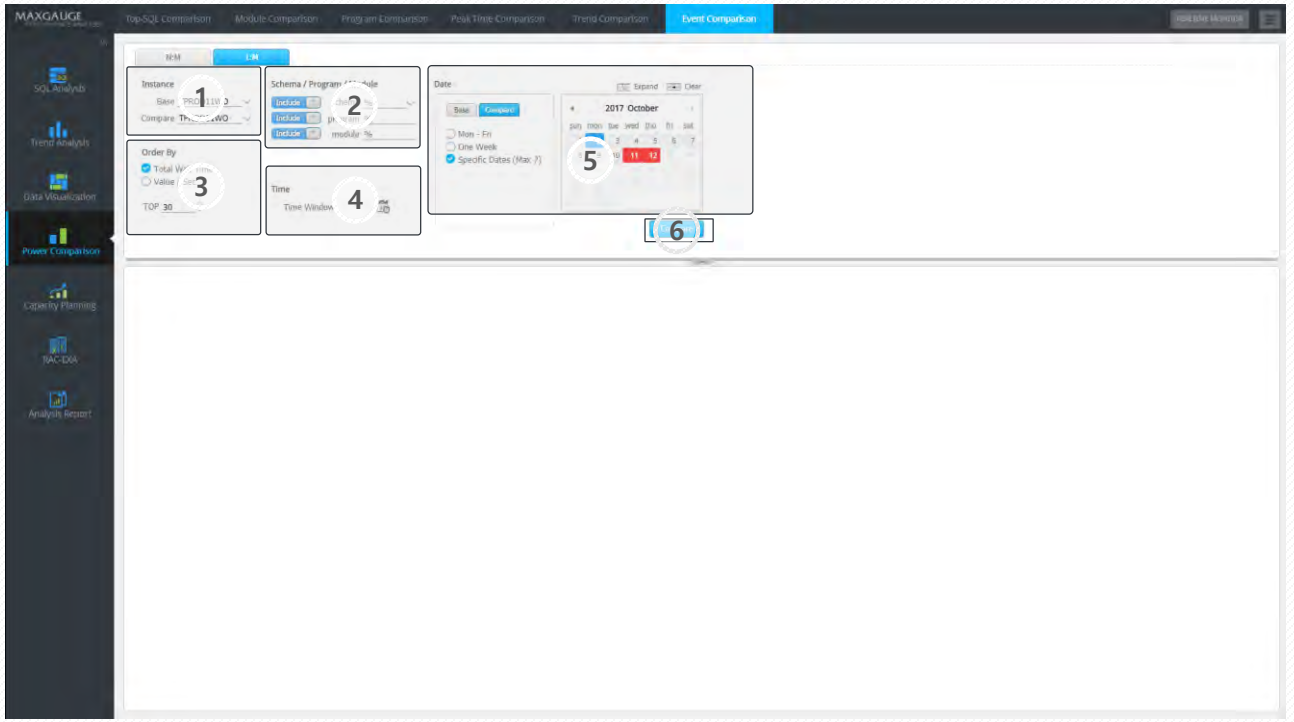
신규 추가 된 업무가 발생시키는 대기 현상이 궁금해요 (1:M 분석)

"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



✓ 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.

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.

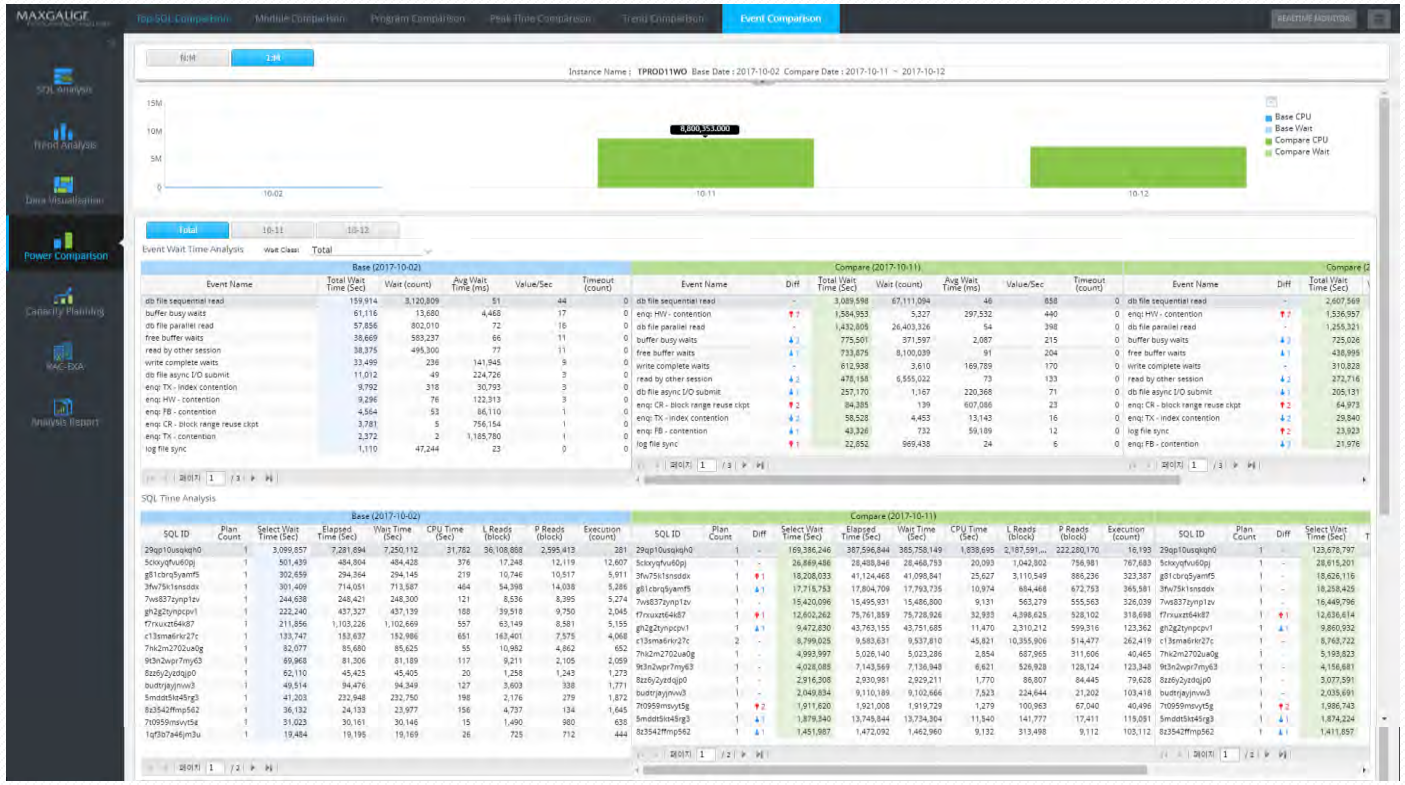
6 위의 설정한 조건으로 **Event Comparison 실행합니다.**

"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



✓ 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 이벤트 입니다.

✓ Event Wait Time Analysis에서 선택한 Event를 발생 시킨 SQL은 SQL Wait Time Analysis에서 확인할 수 있습니다.

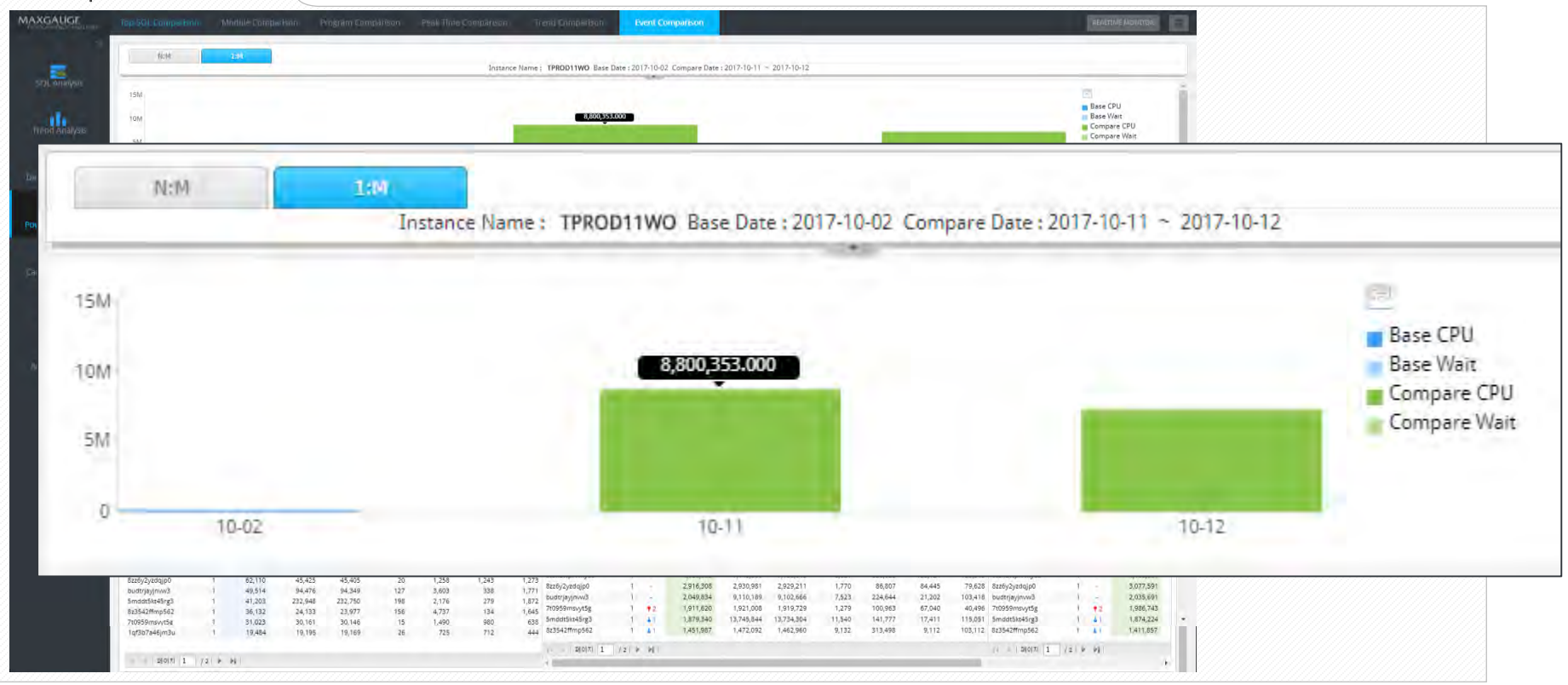
✓ SQL Wait Time Analysis에서 "**NEW**"로 표시된 SQL은 신규 업무 추가로 인해 Wait Time이 증가한 SQL입니다.

"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

» Event Comparison 1:M 조회 화면



✓ 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.

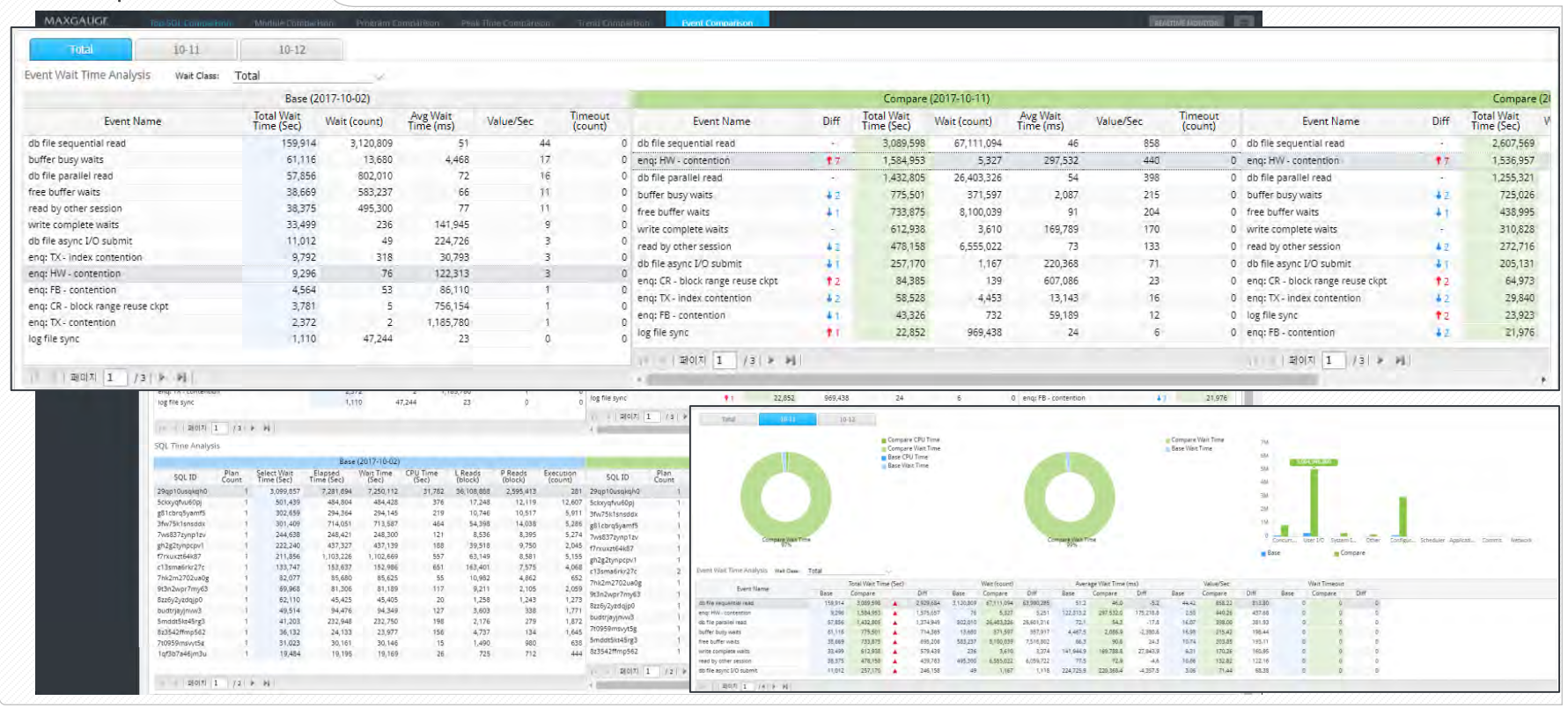
✓ Compare Wait Time이 Base Wait Time에 비해 수치가 높으며, 신규 업무 추가로 Wait Time이 증가했음을 유추할 수 있습니다.

"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 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.

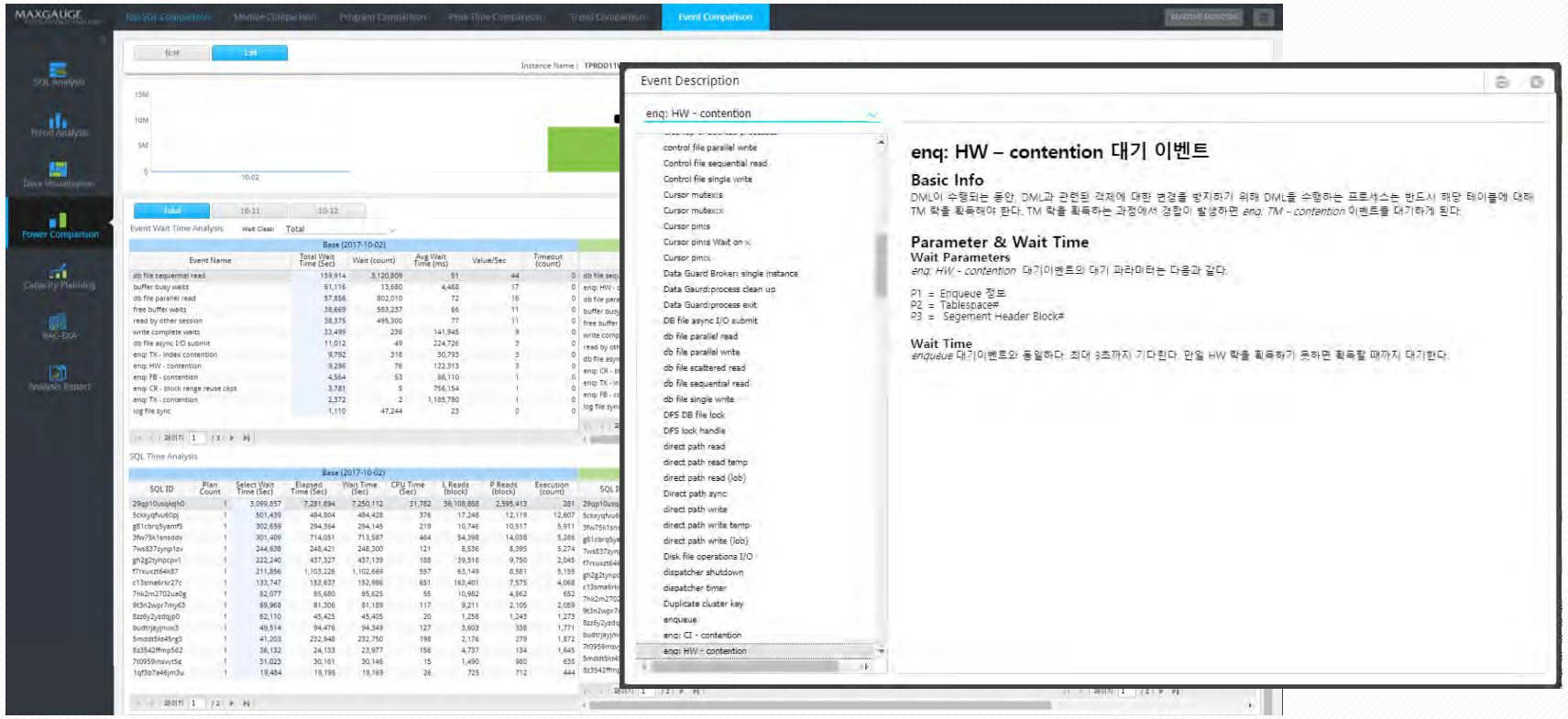
✓ Enq: HW-contention의 이벤트 순위가 ↑7 로 높아진 것으로 확인되며, 신규로 추가된 업무로 인해 Wait Time이 증가한 이벤트 중 하나인 것으로 확인됩니다.

"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



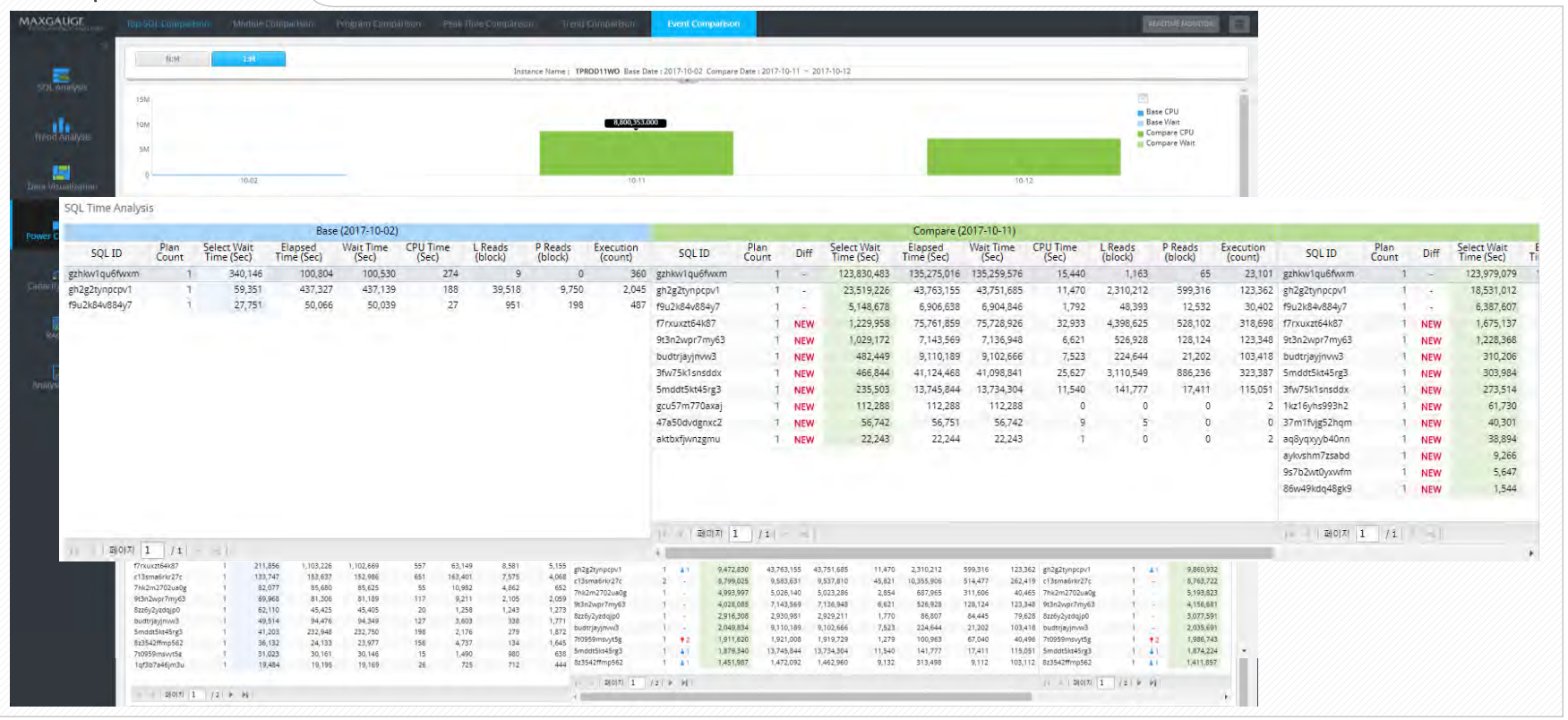
✓ Event Description 기능을 통해 해당 이벤트에 대한 상세 정보를 확인 할 수 있습니다.

"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

» Event Comparison 1:M 조회 화면



✓ 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.

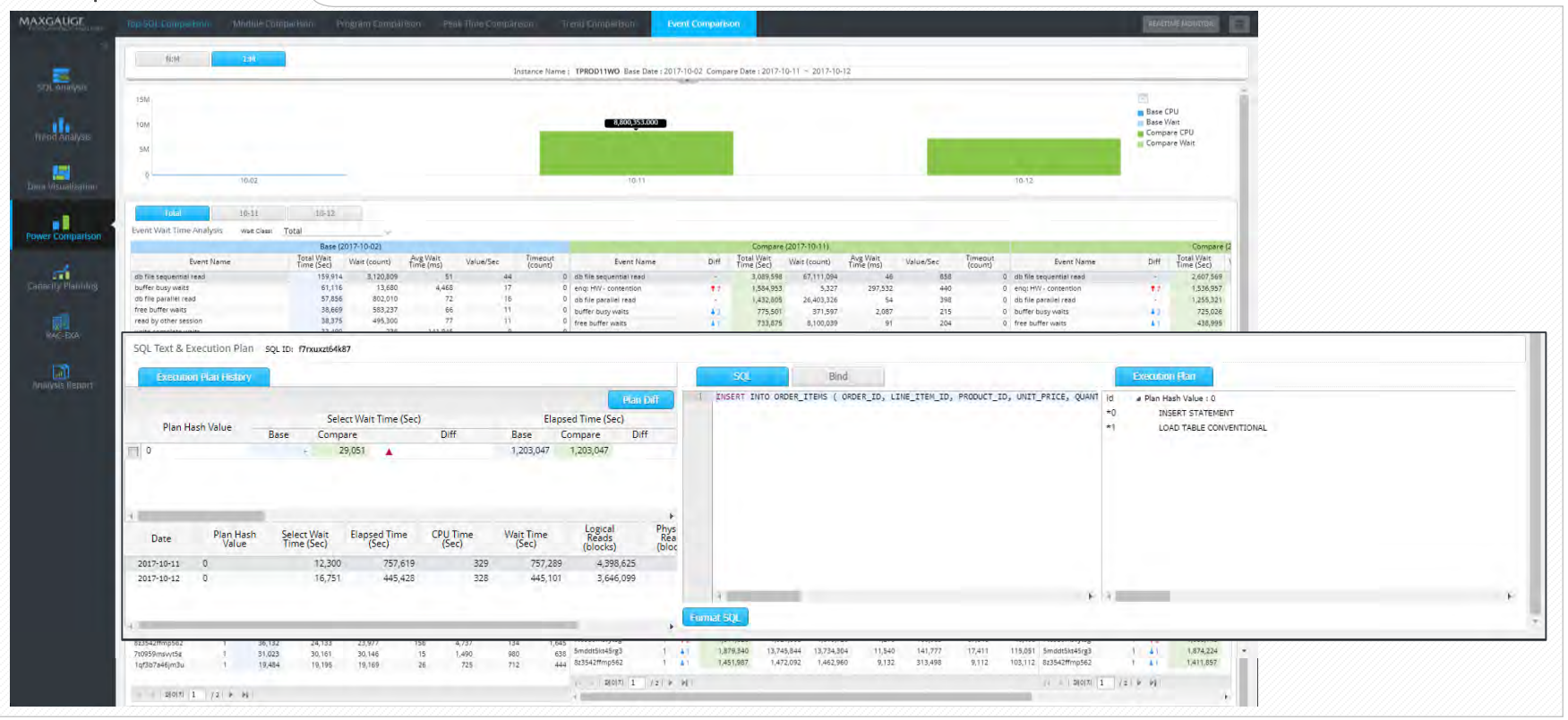
✓ 신규로 추가된 업무의 SQL로 인해 enq: HW–contention 대기가 증가했음을 알 수 있습니다.

"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**

» Event Comparison 1:M 조회 화면



✓ 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.

✓ 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]

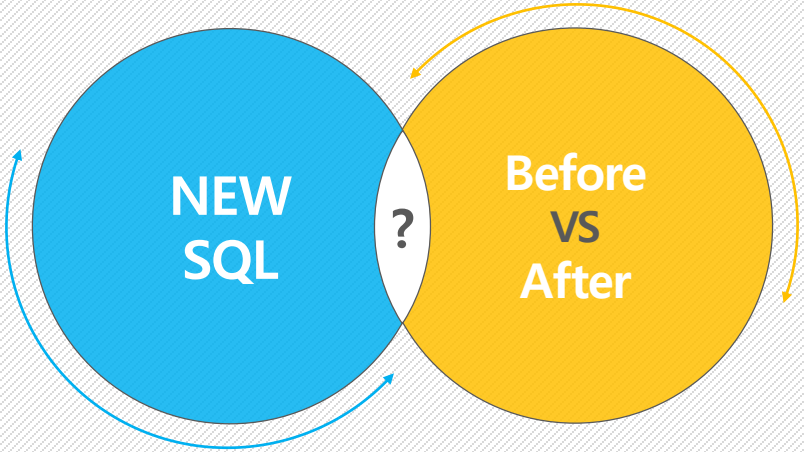
PRODUCT
DOCUMENTATION

Contents

Top-SQL Comparison?

Top SQL Comparison은 언제 쓰나요?

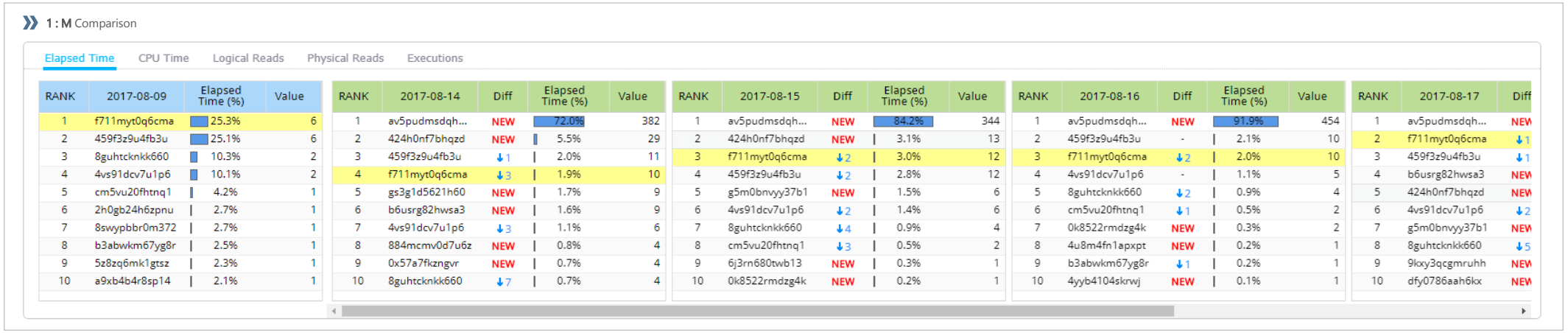
« **신규로 추가된 업무**가
성능에 미치는 영향이 궁금할 때



» **지난 달과 이번 달**의
월 말 결산 작업을 비교하고 싶을 때

Top-SQL Comparison

- 기준 날짜 대비 비교 날짜들의 **Top SQL간 비교 기능**이다.
- 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
1	f711myt0q6cma	25.3%	6
2	459f3z9u4fb3u	25.1%	6
3	8guhtcknkk660	10.3%	2
4	4vs91dcv7u1p6	10.1%	2
5	cm5vu20fhtnq1	4.2%	1
6	2h0gb24h6zpnu	2.7%	1
7	8swypbbr0m372	2.7%	1
8	b3abwkm67yg8r	2.5%	1
9	5z8zq6mk1gtsz	2.3%	1
10	a9xb4b4r8sp14	2.1%	1

RANK	2017-08-14	Diff	Elapsed Time (%)	Value
1	av5pudmsdqh...	NEW	72.0%	382
2	424h0nf7bhqzd	NEW	5.5%	29
3	459f3z9u4fb3u	↓1	2.0%	11
4	f711myt0q6cma	↓3	1.9%	10
5	gs3g1d5621h60	NEW	1.7%	9
6	b6usrg82hwsa3	NEW	1.6%	9
7	4vs91dcv7u1p6	↓3	1.1%	6
8	884mcmv0d7u6z	NEW	0.8%	4
9	0x57a7fkzngvr	NEW	0.7%	4
10	8guhtcknkk660	↓7	0.7%	4

RANK	2017-08-15	Diff	Elapsed Time (%)	Value
1	av5pudmsdqh...	NEW	84.2%	344
2	424h0nf7bhqzd	NEW	3.1%	13
3	f711myt0q6cma	↓2	3.0%	12
4	459f3z9u4fb3u	↓2	2.8%	12
5	g5m0bnvyy37b1	NEW	1.5%	6
6	4vs91dcv7u1p6	↓2	1.4%	6
7	8guhtcknkk660	↓4	0.9%	4
8	cm5vu20fhtnq1	↓3	0.5%	2
9	6j3rn680twb13	NEW	0.3%	1
10	0k8522rmdzg4k	NEW	0.2%	1

RANK	2017-08-16	Diff	Elapsed Time (%)	Value
1	av5pudmsdqh...	NEW	91.9%	454
2	459f3z9u4fb3u	-	2.1%	10
3	f711myt0q6cma	↓2	2.0%	10
4	4vs91dcv7u1p6	-	1.1%	5
5	8guhtcknkk660	↓2	0.9%	4
6	cm5vu20fhtnq1	↓1	0.5%	2
7	0k8522rmdzg4k	NEW	0.3%	2
8	4u8m4fn1apxpt	NEW	0.2%	1
9	b3abwkm67yg8r	↓1	0.2%	1
10	4yyb4104skrwj	NEW	0.1%	1

RANK	2017-08-17	Diff
1	av5pudmsdqh...	NEW
2	f711myt0q6cma	↓1
3	459f3z9u4fb3u	↓1
4	b6usrg82hwsa3	NEW
5	424h0nf7bhqzd	NEW
6	4vs91dcv7u1p6	↓2
7	g5m0bnvyy37b1	NEW
8	8guhtcknkk660	↓5
9	9kxy3qcgmruhh	NEW
10	dfy0786aah6kx	NEW

N : M Comparison

Elapsed Time | CPU Time | Logical Reads | Physical Reads | Executions

RANK	2017-08-14 ~ 2017-08-18	Elapsed Time (%)	Value
1	av5pudmsdqh...	82.3%	1,755
2	f711myt0q6cma	2.4%	52
3	424h0nf7bhqzd	2.2%	48
4	459f3z9u4fb3u	2.2%	46
5	4vs91dcv7u1p6	1.2%	26
6	8guhtcknkk660	0.8%	17
7	b6usrg82hwsa3	0.7%	15
8	cm5vu20fhtnq1	0.5%	11
9	g5m0bnvyy37b1	0.5%	11
10	gs3g1d5621h60	0.4%	9

RANK	2017-08-14 ~ 2017-08-18	Elapsed Time (%)	Value
11	0k8522rmdzg4k	0.3%	6
12	63kf3an2j0pkc	0.3%	6
13	9kxy3qcgmruhh	0.3%	5
14	g9m7794s0k622	0.2%	5
15	884mcmv0d7u6z	0.2%	4
16	0x57a7fkzngvr	0.2%	4
17	f260gh1ks21rz	0.2%	3
18	b3abwkm67yg8r	0.1%	3
19	ffrxyztt4415k	0.1%	3
20	d0xmnp08rhfbg	0.1%	3

RANK	2017-08-07 ~ 2017-08-11	Diff	Elapsed Time (%)	Value
1	av5pudmsdqh...	-	35.7%	76
2	459f3z9u4fb3u	↑2	15.9%	34
3	f711myt0q6cma	↓1	15.1%	32
4	4vs91dcv7u1p6	↑1	5.9%	13
5	8guhtcknkk660	↑1	5.8%	12
6	cm5vu20fhtnq1	↑2	2.2%	5
7	53c2k4c43zcfx	NEW	2.2%	5
8	b3abwkm67yg8r	↑10	1.8%	4
9	4yyb4104skrwj	NEW	1.5%	3
10	a9xb4b4r8sp14	NEW	1.5%	3

RANK	2017-08-07 ~ 2017-08-11	Diff	Elapsed Time (%)	Value
11	6j3rn680twb13	NEW	1.4%	3
12	0k8522rmdzg4k	↓1	1.0%	2
13	8swypbbr0m372	NEW	0.9%	2
14	2h0gb24h6zpnu	NEW	0.9%	2
15	4u8m4fn1apxpt	NEW	0.8%	2
16	anyj2gn8hcd5y	NEW	0.8%	2
17	0ws7ahf1d78qa	NEW	0.6%	1
18	bz6h5vq8kc02q	NEW	0.6%	1
19	27k1sg83rqtcp	NEW	0.6%	1
20	9s0xa5dgvuq55	NEW	0.5%	1

Top-SQL의 비교를 더욱 쉽게 할 수 있습니다!

SQL 분석 방법

MaxGauge

Performance Analyzer → Top-SQL Comparison → 튜닝 대상 SQL 선별

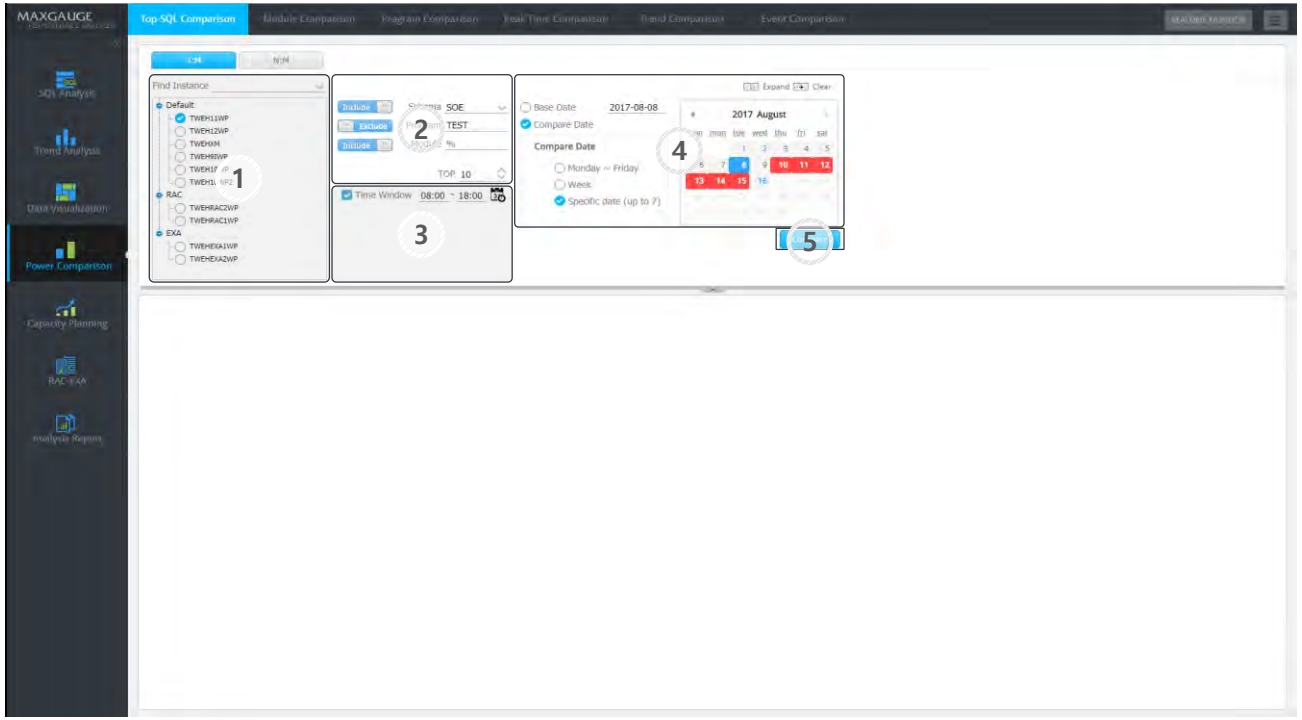
AWR

SQL Tuning

Top-SQL Comparison의 활용

Top-SQL Comparison의 사용 방법 (1:M)

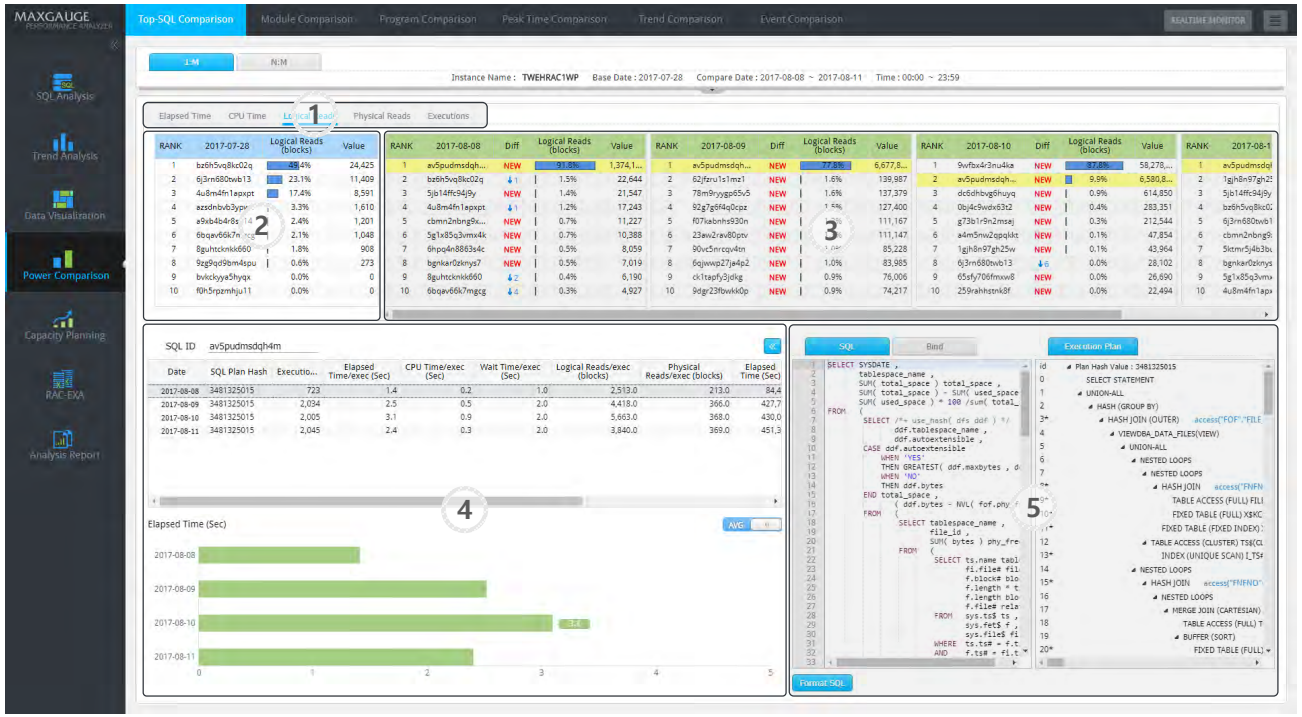
» Top-SQL Comparison 1:M 검색화면



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 실행**합니다.

Top-SQL Comparison의 사용 방법 (1:M)

» Top-SQL Comparison 1 : M 조회 화면



1 TOP SQL 추출 기준은 Elapsed Time, CPU Time, Logical Reads, Physical Reads, Executions 으로 분류 됩니다.

2 신규 업무 추가 전 기준 날짜 (Base Date) 기준 TOP SQL

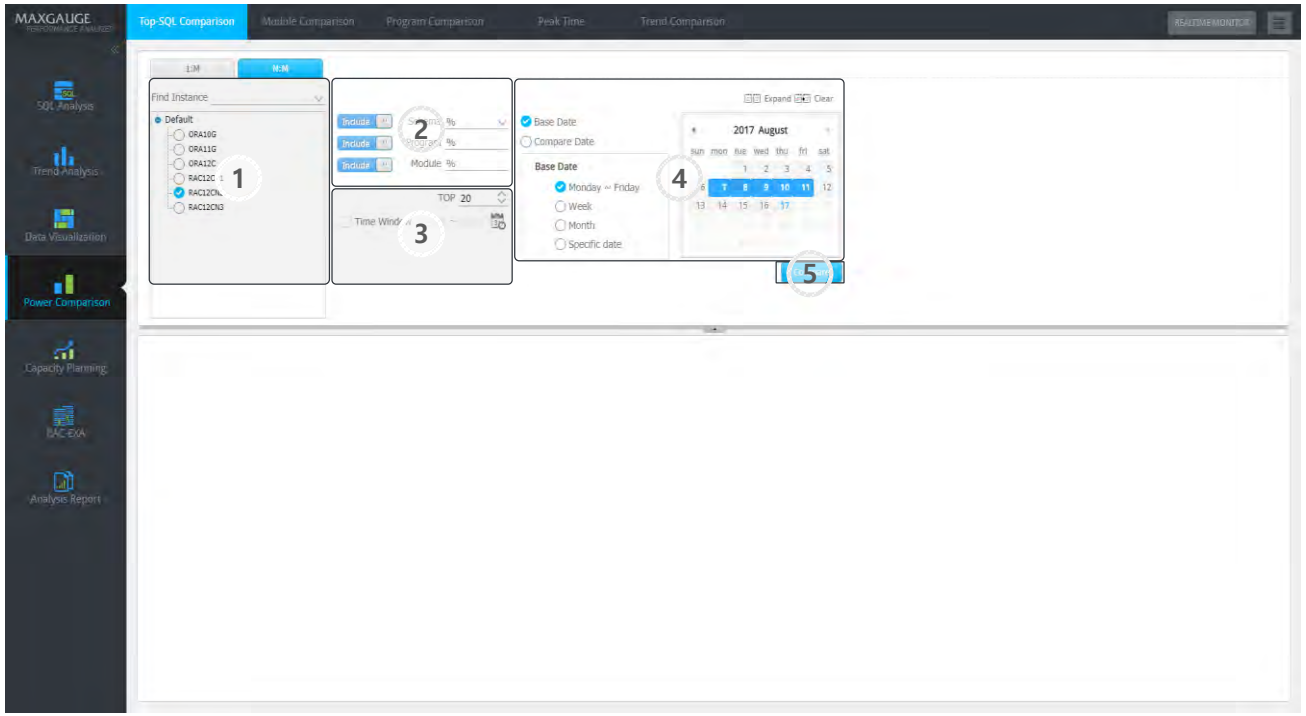
3 신규 업무 추가 후, 각 비교 날짜 (Compare Date) 기준으로 추출된 TOP SQL

4 Compare Date별 수행 정보를 통한 SQL 성능 및 SUM / AVG 정보 확인

5 선택한 SQL에 대한 SQL Text 및 Bind, Execution Plan 정보 제공

Top-SQL Comparison의 사용 방법 (N:M)

» 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.

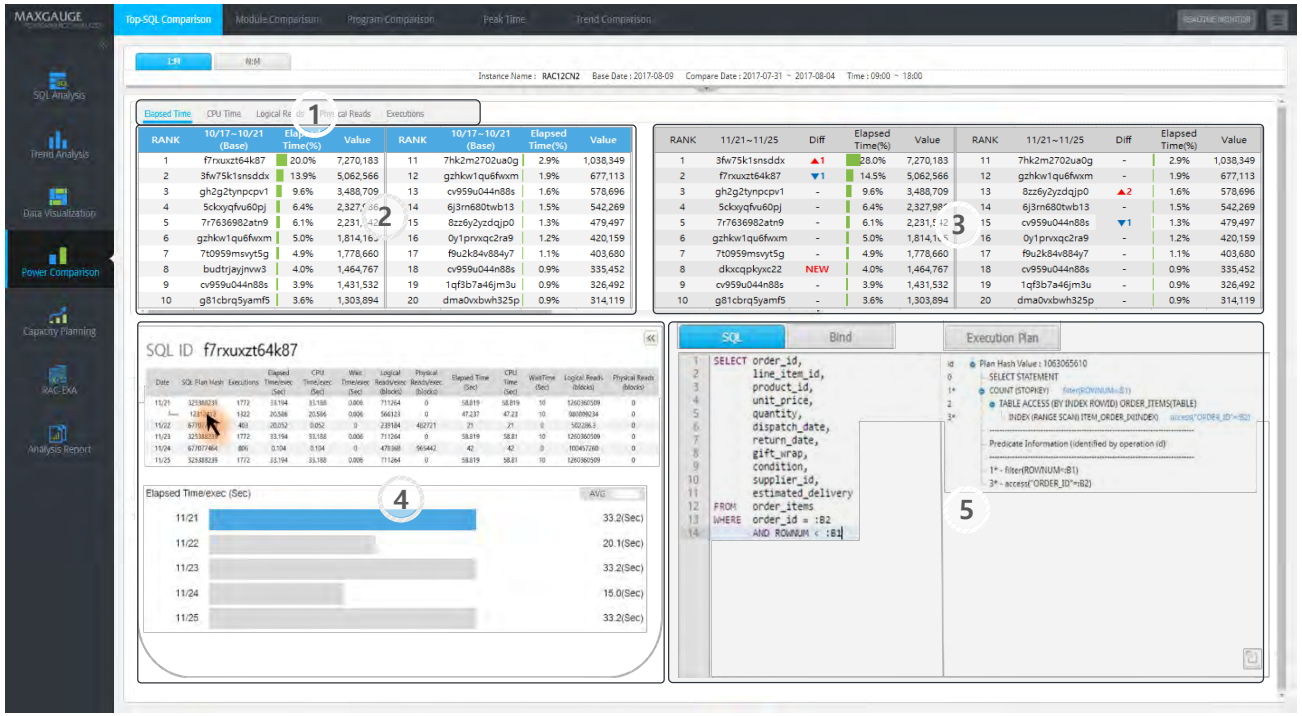
3 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.

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 조회화면



1 TOP SQL 추출 기준은 Elapsed Time, CPU Time, Logical Reads, Physical Reads, Executions 으로 분류 됩니다.

2 월 단위 업무의 기준 날짜 (**Base Date**)의 TOP SQL 정보입니다.

3 월 단위 업무의 비교 날짜 (**Compare Date**) 기준으로 추출된 TOP SQL 정보입니다.

4 비교 날짜(**Compare Date**) 별 수행 정보를 통한 **SQL 성능 및 SUM / AVG 정보**를 확인할 수 있습니다.

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

실전 분석 사례 Case 1.

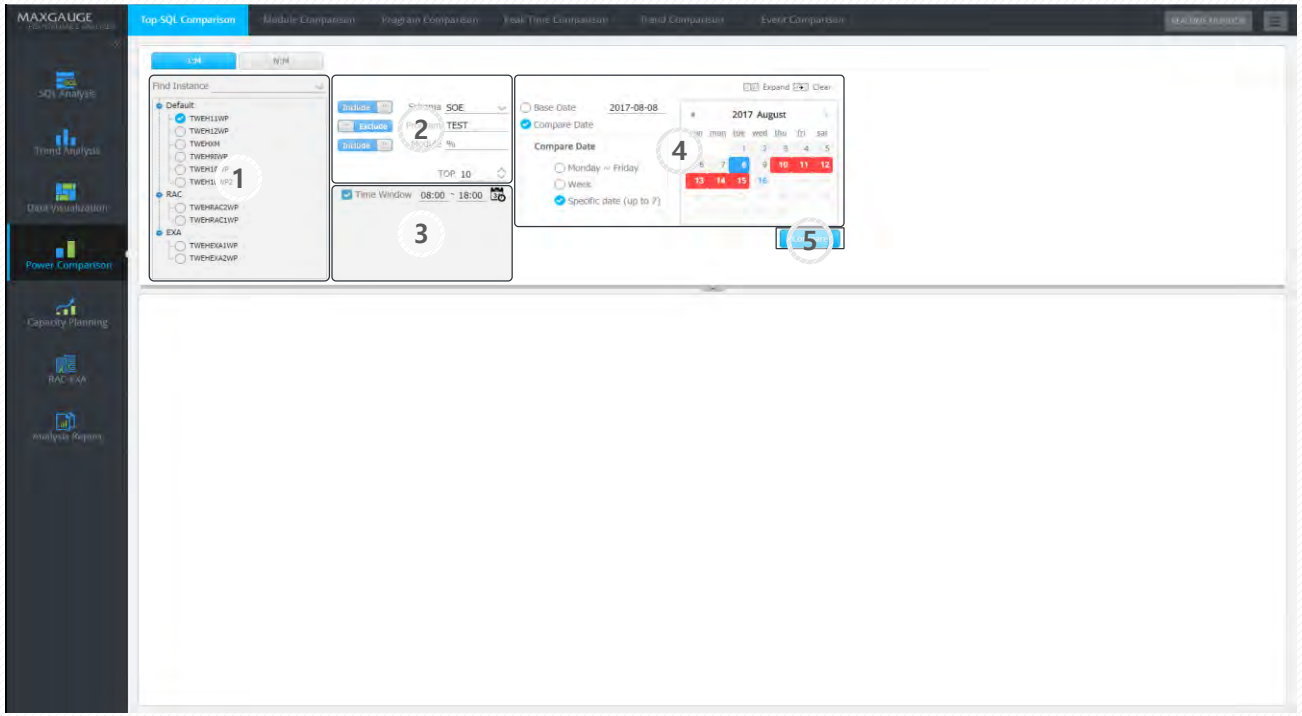
신규로 추가 된 업무 성능이 궁금해요

" 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



✓ **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.

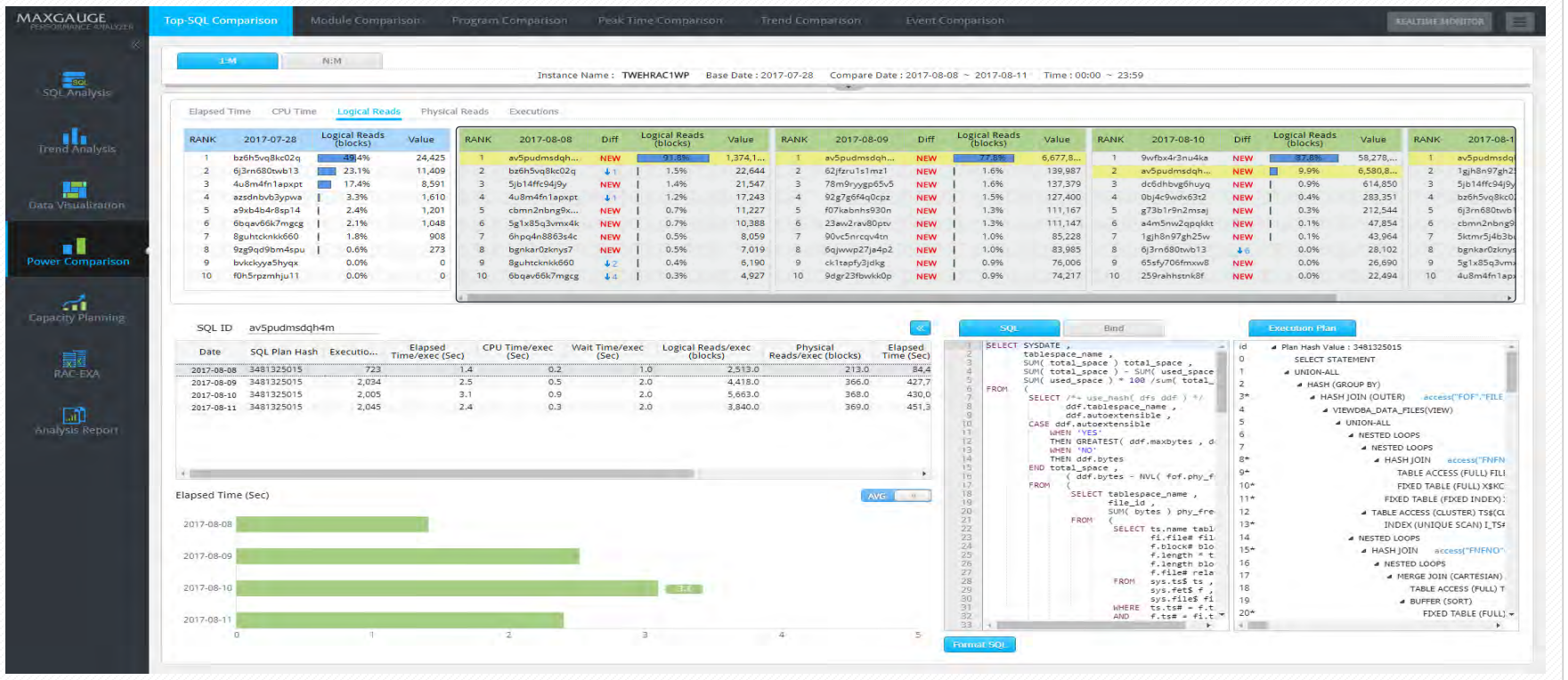
1. 비교 대상 Instance (**TWEH11WP**)를 선택합니다.
2. Schema (**MAXGAUGE**), Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
3. 상세 시간 구분 시 선택하는 탭이며 00:00~23:59 (**09:00 ~ 18:00**) 을 선택적으로 선택 할 수 있습니다.
4. 기준 날짜 (**2017.07.28**) 는 하루만 선택되며, 비교 날짜는 Working day, Week, 특정 날짜 (**2017.08.08 ~ 2017.08.11**)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

" 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



✓ Top SQL중 Logical Reads이 높은 Top SQL들을 확인합니다.

✓ 신규 업무 추가 된 비교 날짜 TOP SQL 중, "NEW" 로 표시된 SQL들이 있는지 확인합니다.

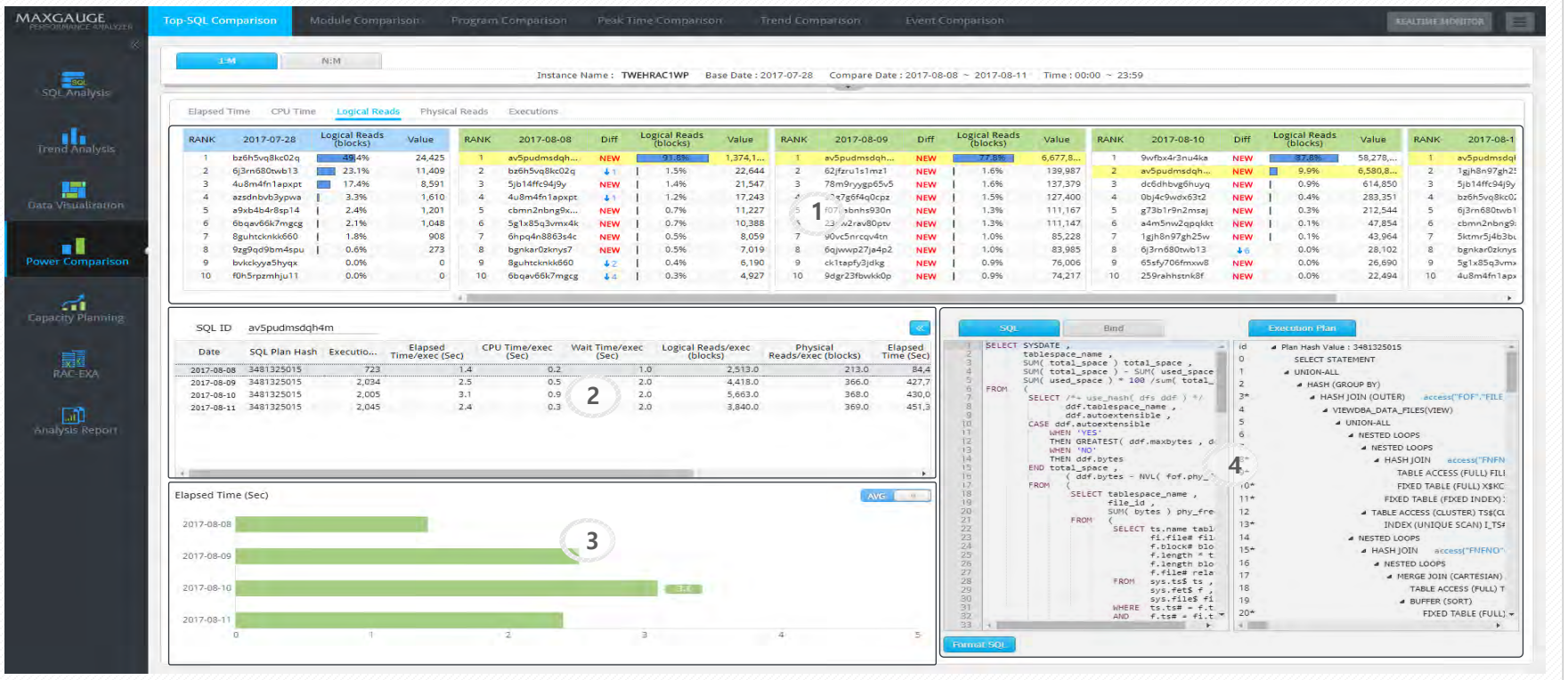
✓ "New" 표시 된 SQL들은 신규 업무로 추가된 SQL 이거나, 기존에 수행되었으나 Plan 변경 등의 이유로 TOP으로 선정된 SQL 일 수 있습니다.

" 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 조회화면



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.

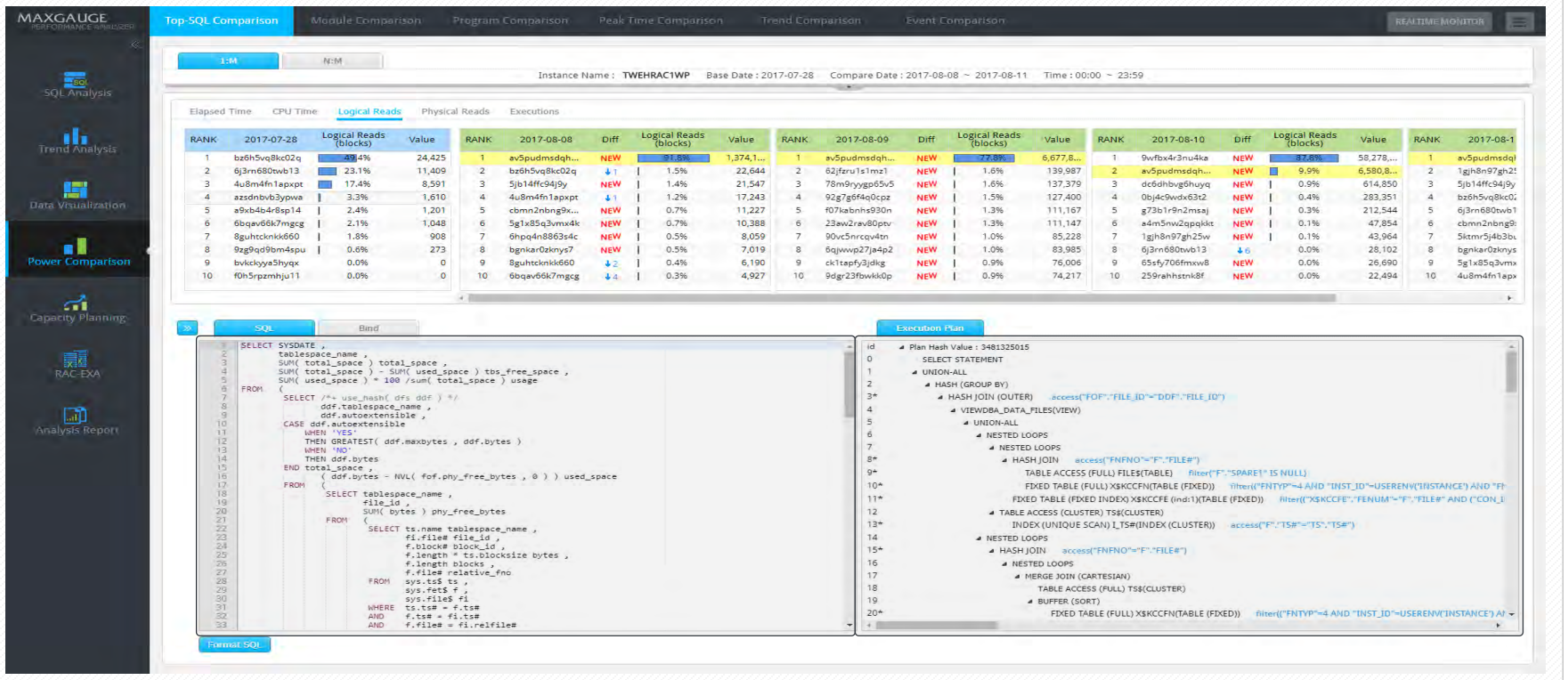
4 선택 된 SQL_ID의 SQL Full Text, Bind 변수, 실행계획을 제공합니다.

" 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



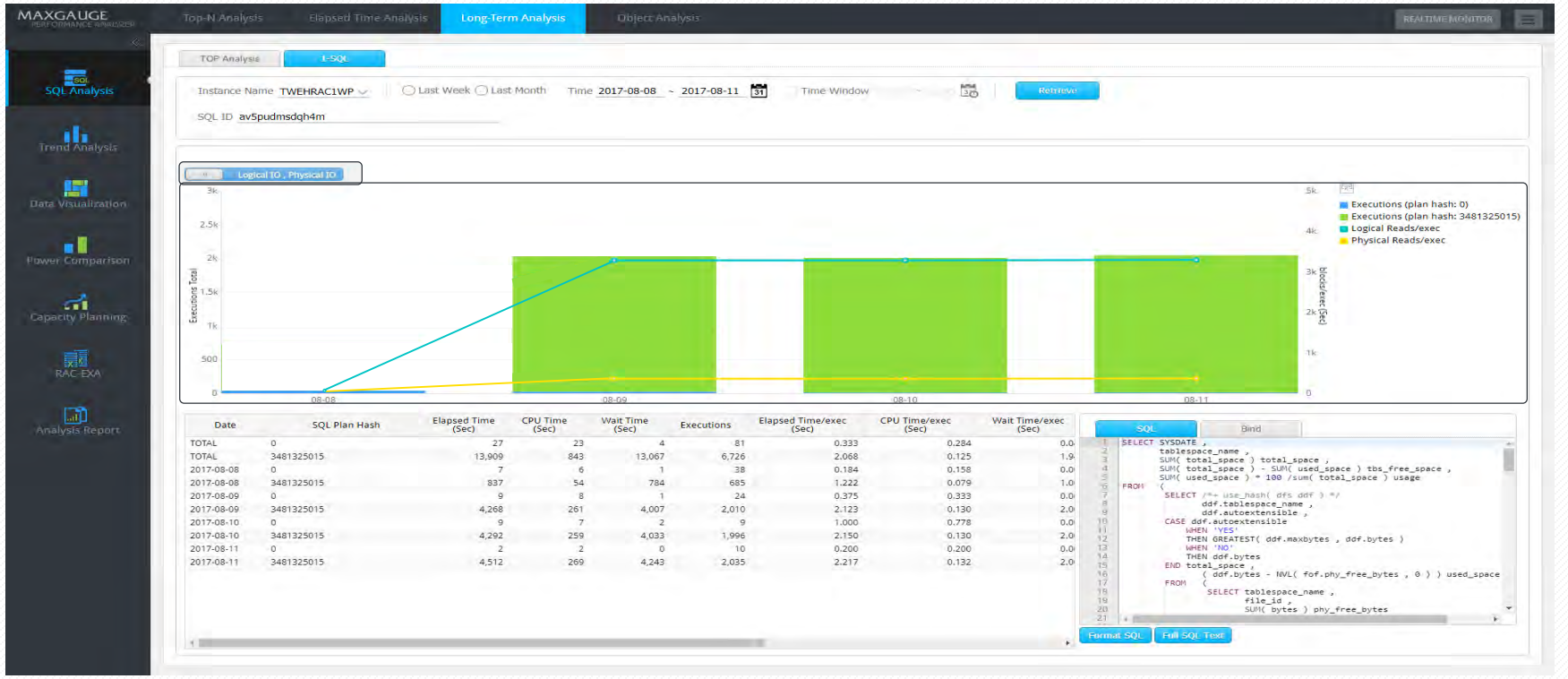
✓ 해당 SQL의 Plan 정보를 확인합니다.

" 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

» Top-SQL Comparison 1:M 조회 화면



✓ "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.

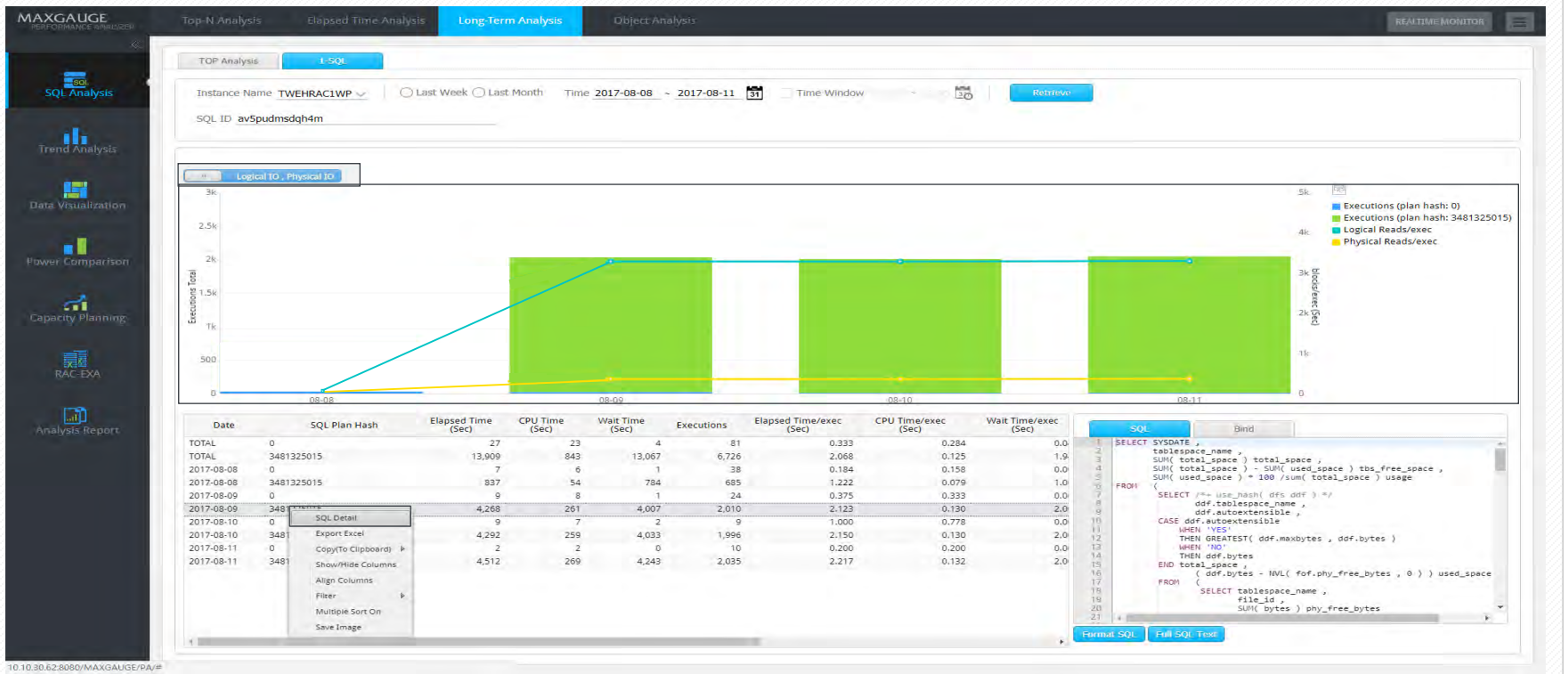
✓ 과거 SQL 수행 이력이 없음을 통해, 신규 업무로 추가된 SQL임을 알 수 있습니다.

" 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



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

" 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

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

✓ 해당 SQL의 시간 별 수행 정보를 통해 성능 지연의 원인을 파악할 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

✓ "db file Sequential Reads"로 대기하는 비중이 큰 것을 알 수 있습니다.

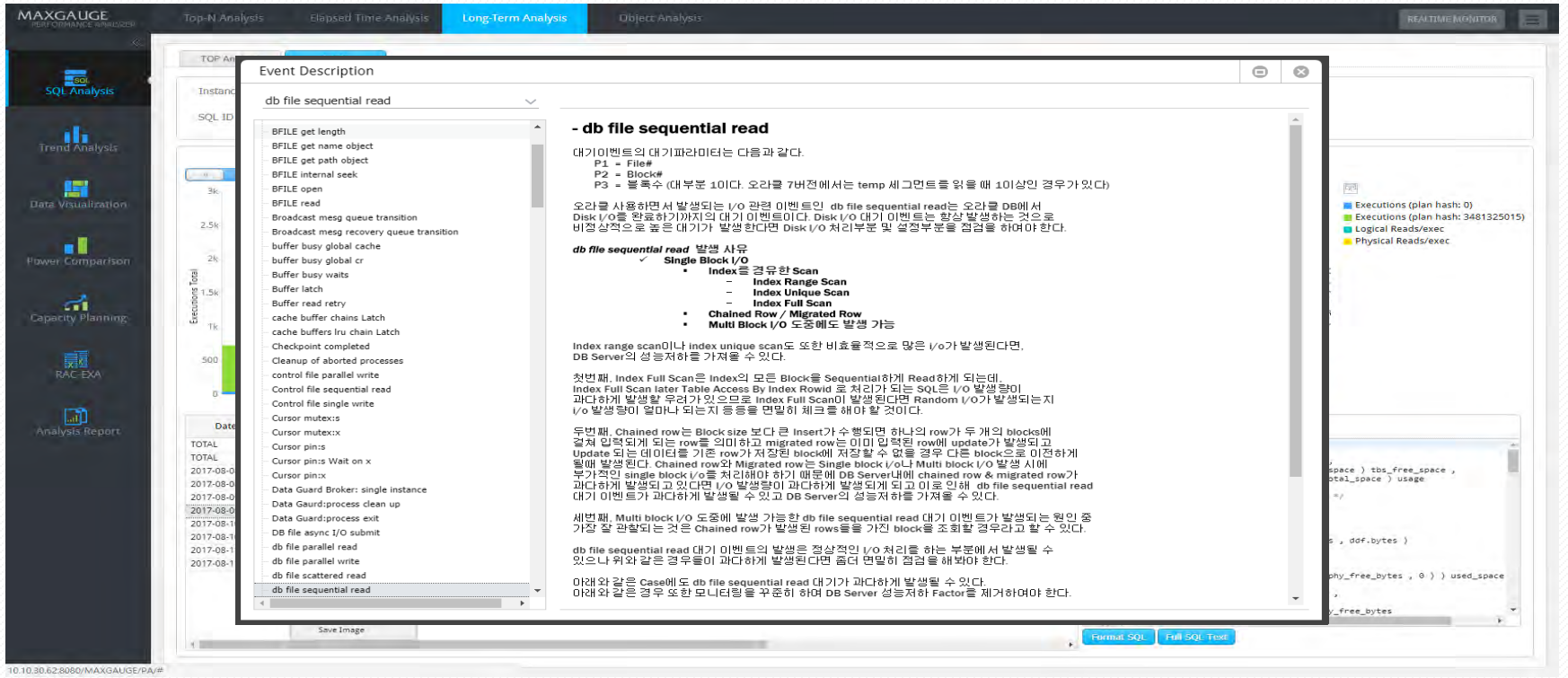
✓ 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 Event Description으로 연계합니다.

" 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



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

실전 분석 사례 Case 2.

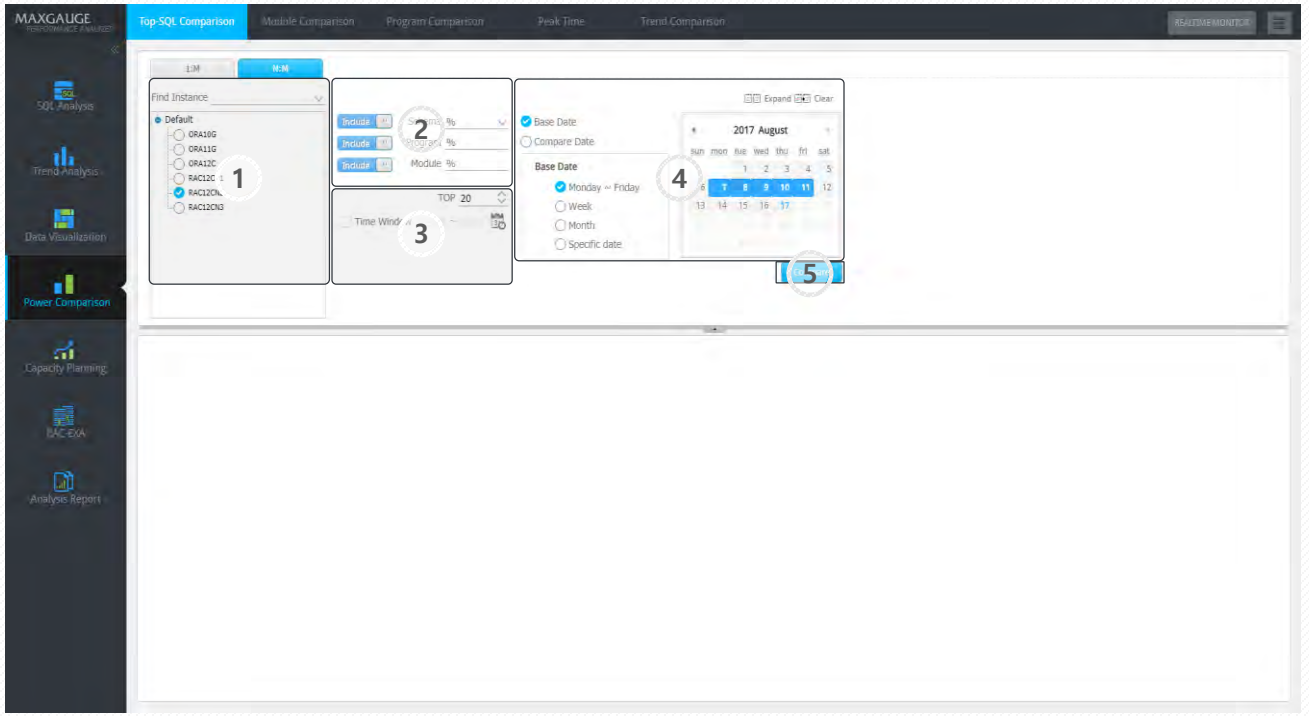
지난 달과 이번 달의 월 단위 작업을 비교하고 싶어요

"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 검색 화면



✓ **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

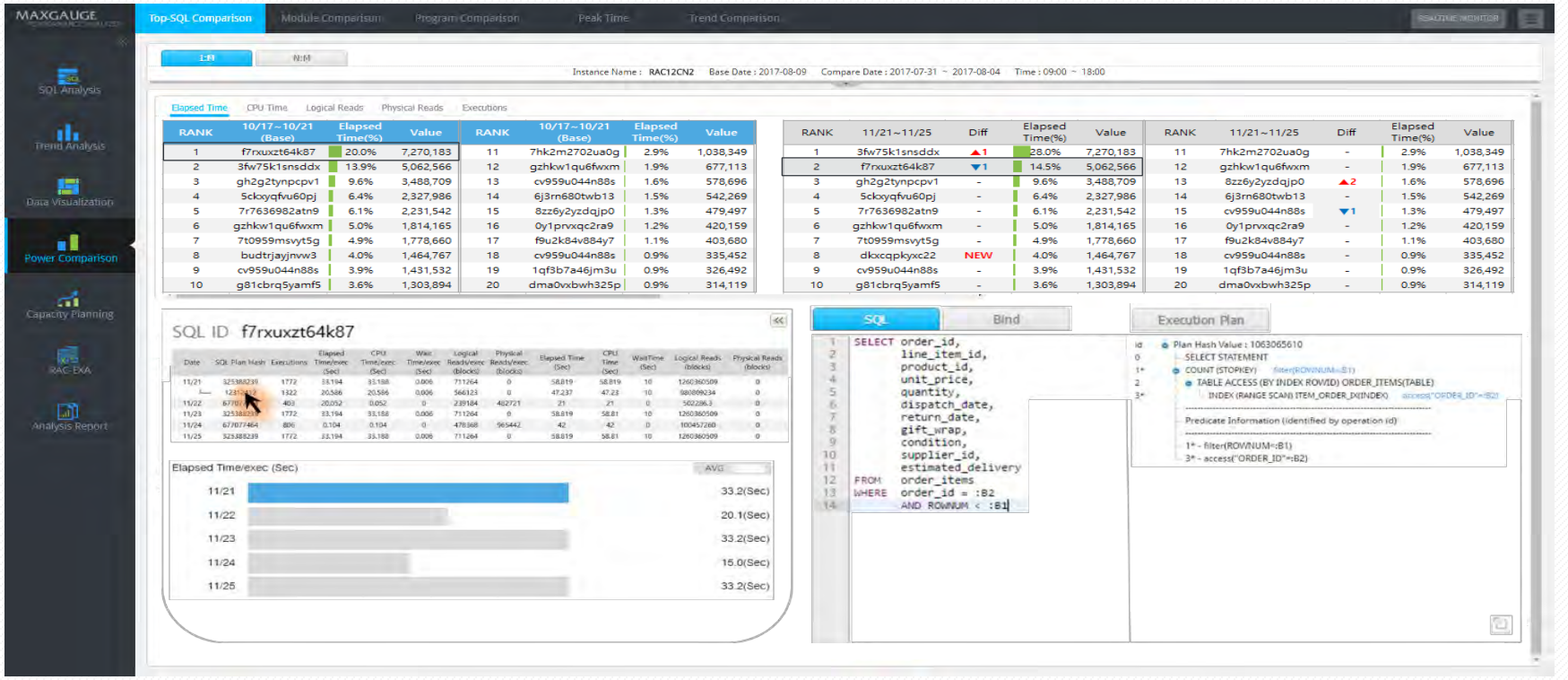
1. 비교 대상 Instance (**RAC12CN2**)를 선택합니다.
2. Schema, Program, Module명을 선택적으로 빼거나 넣을 수 있으며, 출력되는 SQL의 수를 지정할 수 있습니다.
3. 상세 시간 구분 시 선택하는 탭이며 00:00~23:59를 선택적으로 선택 할 수 있습니다.
4. 기준 날짜(**17.08.07 ~ 17.08.11**)와 비교 날짜(**17.07.10~17.07.14**) 모두 Working day, Week, Month, 특정 날짜로 선택 가능합니다.
5. 위의 설정한 조건으로 **Top-SQL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

"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



✓ Top SQL중 가장 Elapsed Time이 높은 Top-1 SQL을 확인합니다.

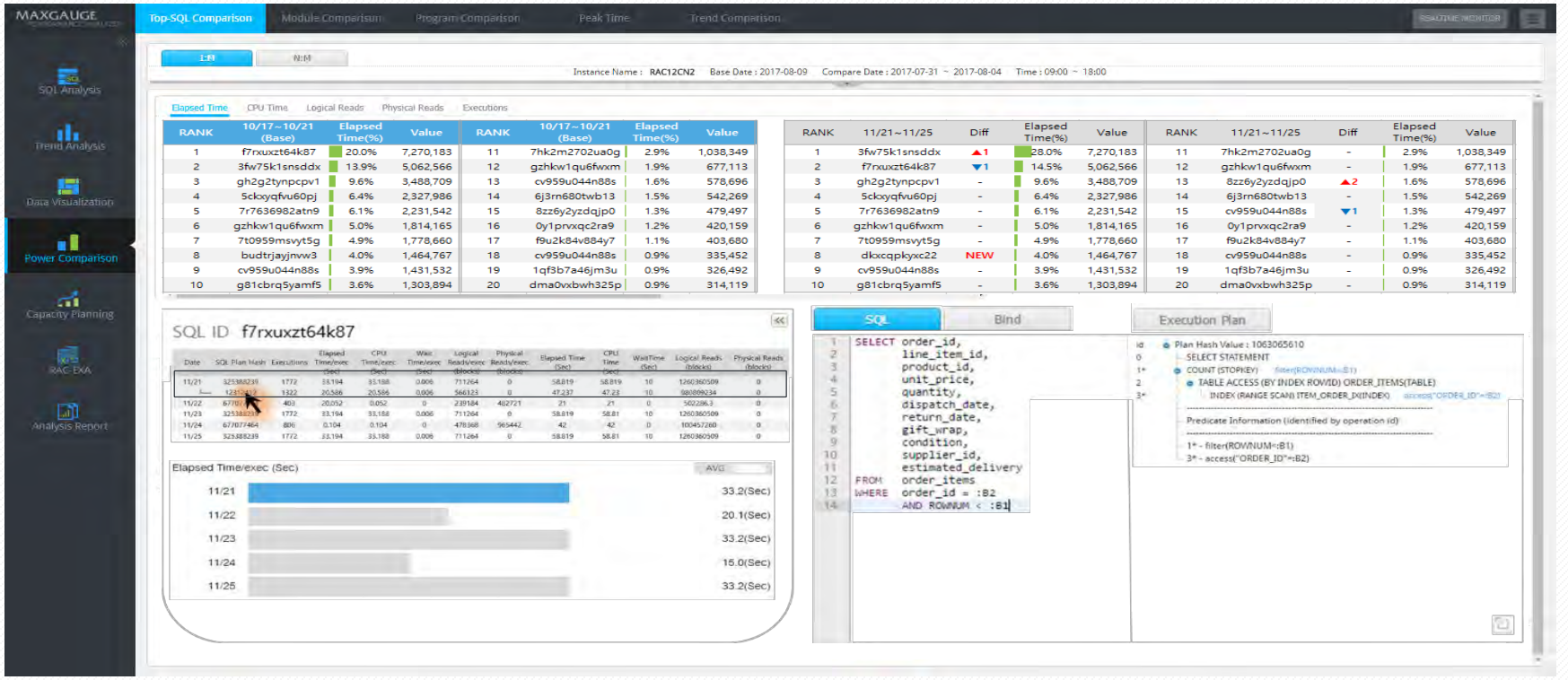
✓ 해당 SQL은 이전 달보다 Elapsed Time의 비중이 감소(**20.0% -> 14.5%**)하여 Top-1에서 Top-2으로 변경 된 것을 확인 할 수 있습니다.

"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



✓ 선택한 SQL의 Plan 정보와 실행 횟수(Execution)당 Elapsed Time의 SUM / AVG 정보를 확인 할 수 있습니다.

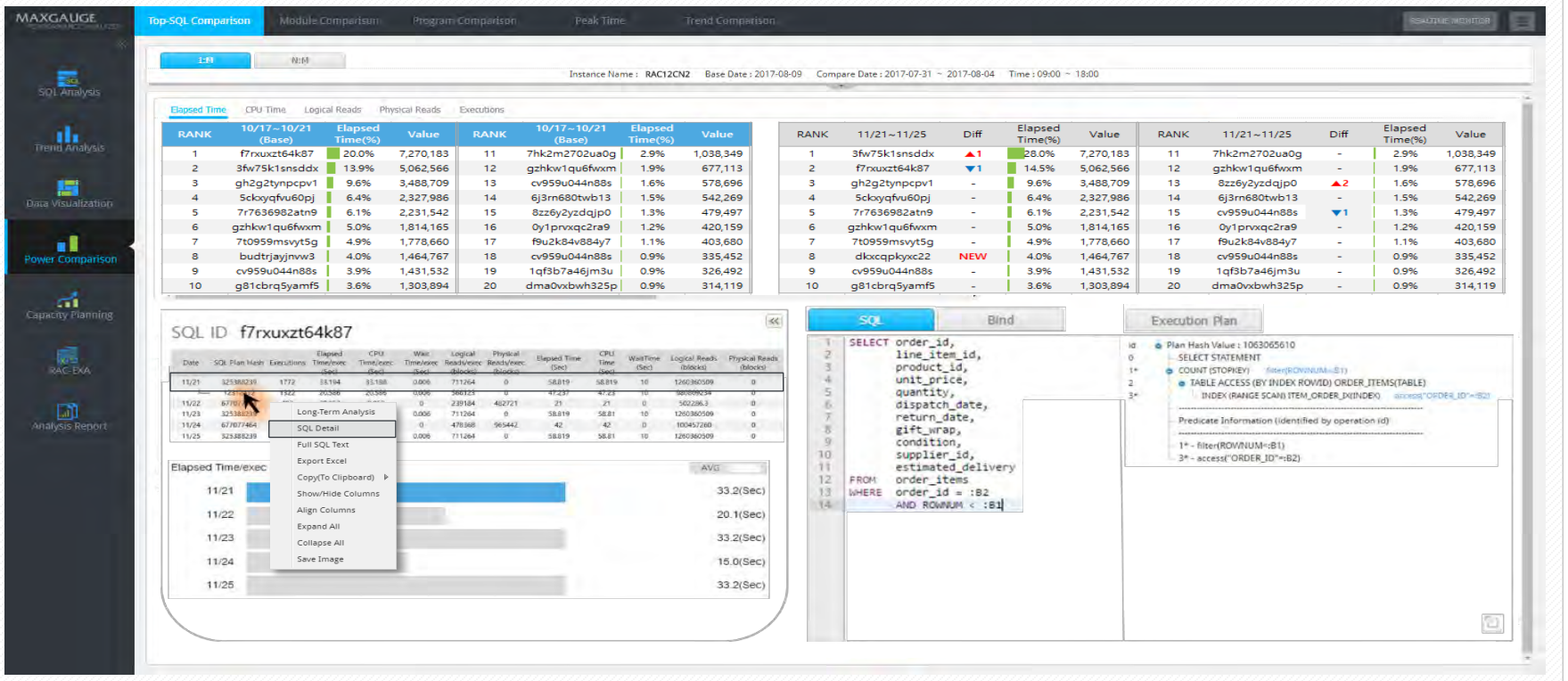
✓ 해당 SQL의 Plan Hash Value가 2개 뜨는 것을 확인 할 수 있는데, 이를 통해 SQL의 Plan이 변경 되었다는 것을 확인 할 수 있습니다.

❖ **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

"지난 달과 이번 달의 월 단위 성능을 비교하고 싶어요 ! "

» Top-SQL Comparison N:M 조회화면



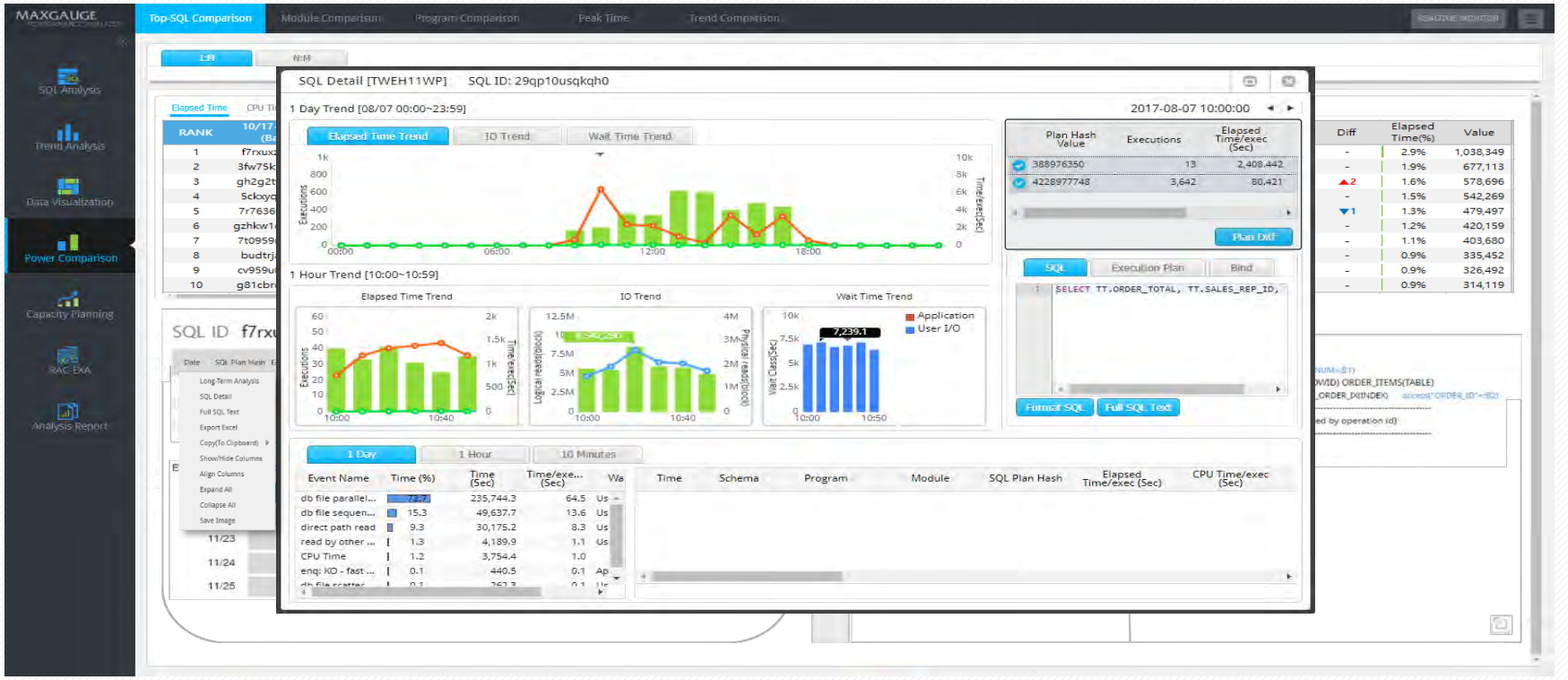
✓ 선택한 SQL Plan의 자세한 정보를 확인하기 위해 마우스 오른쪽 클릭 후, SQL Detail을 선택합니다.

"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 조회화면



✓ SQL Detail을 통해 선택한 SQL의 상세한 정보를 확인 할 수 있습니다.

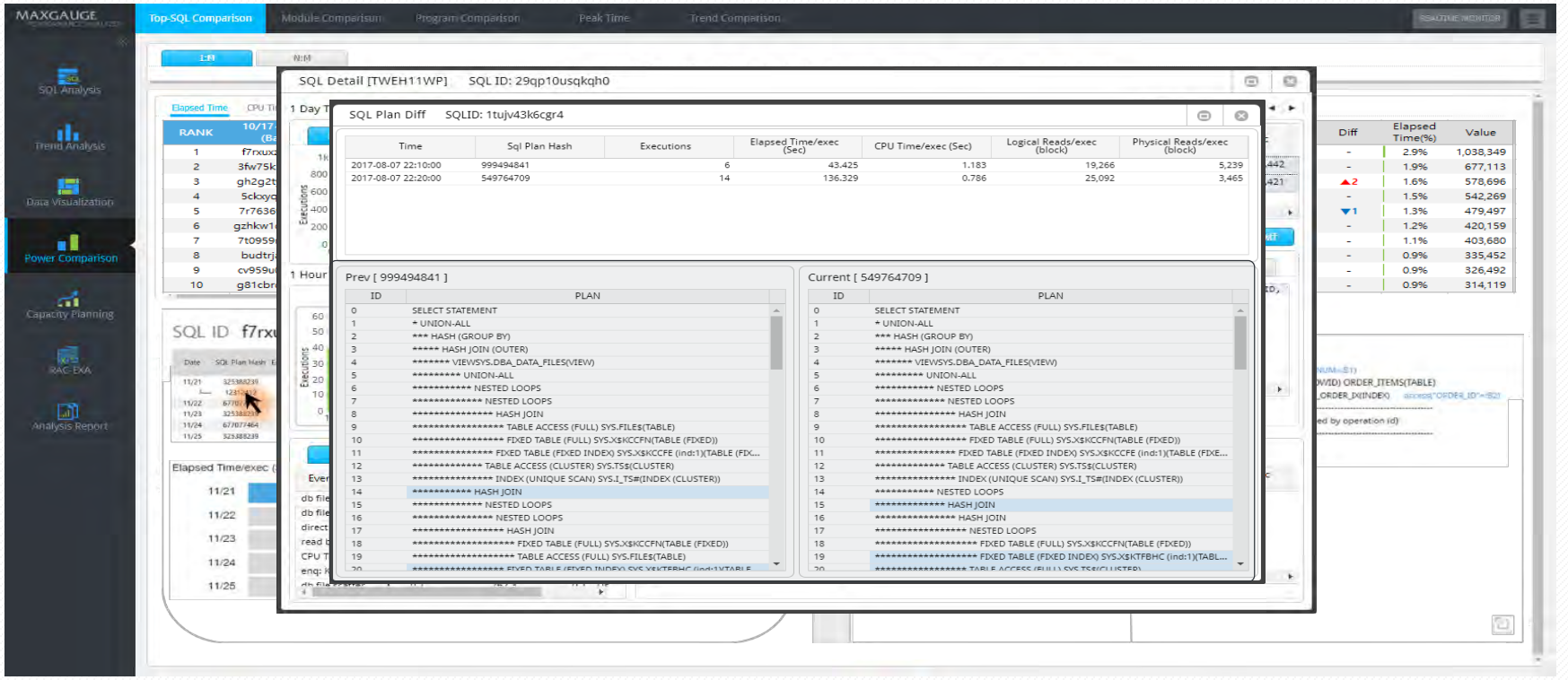
✓ 변경 된 Plan의 정보를 오른쪽 상단에서 확인 할 수 있으며, 변경 된 Plan 간의 비교를 위해 Plan을 선택 후 Plan diff를 클릭합니다.

"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 조회화면



✓ 이전 Plan과 변경 된 현재 Plan을 동시에 확인하여 어떻게 Plan이 변경 되었는지 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.

Plan Change Histroy

MAXGAUGE Practical Guide

Trend Comparison [Performance Analyzer]

PRODUCT
DOCUMENTATION

Practical Guide

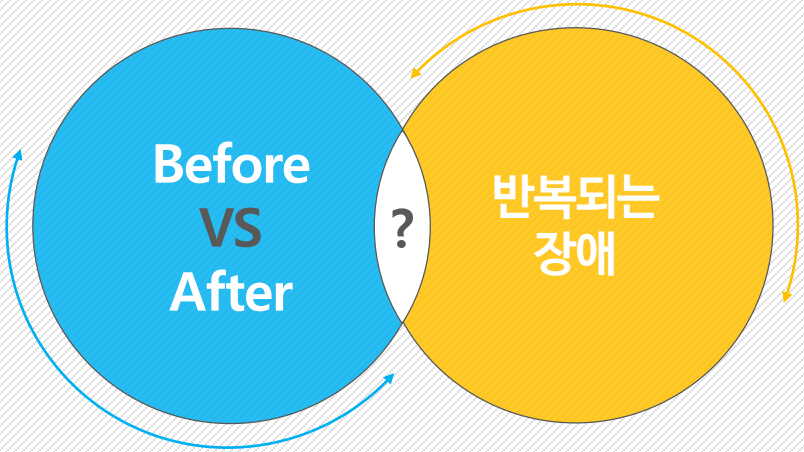
Trend Comparison

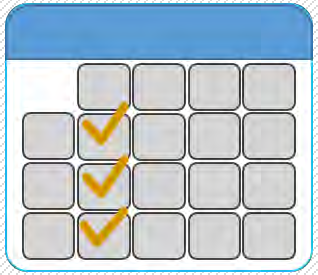
Contents

Trend Comparison?

Trend Comparison은 언제 쓰나요?

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





» **매주 같은 시간 반복**적으로
장애를 발생시키는 원인을 알고 싶을 때

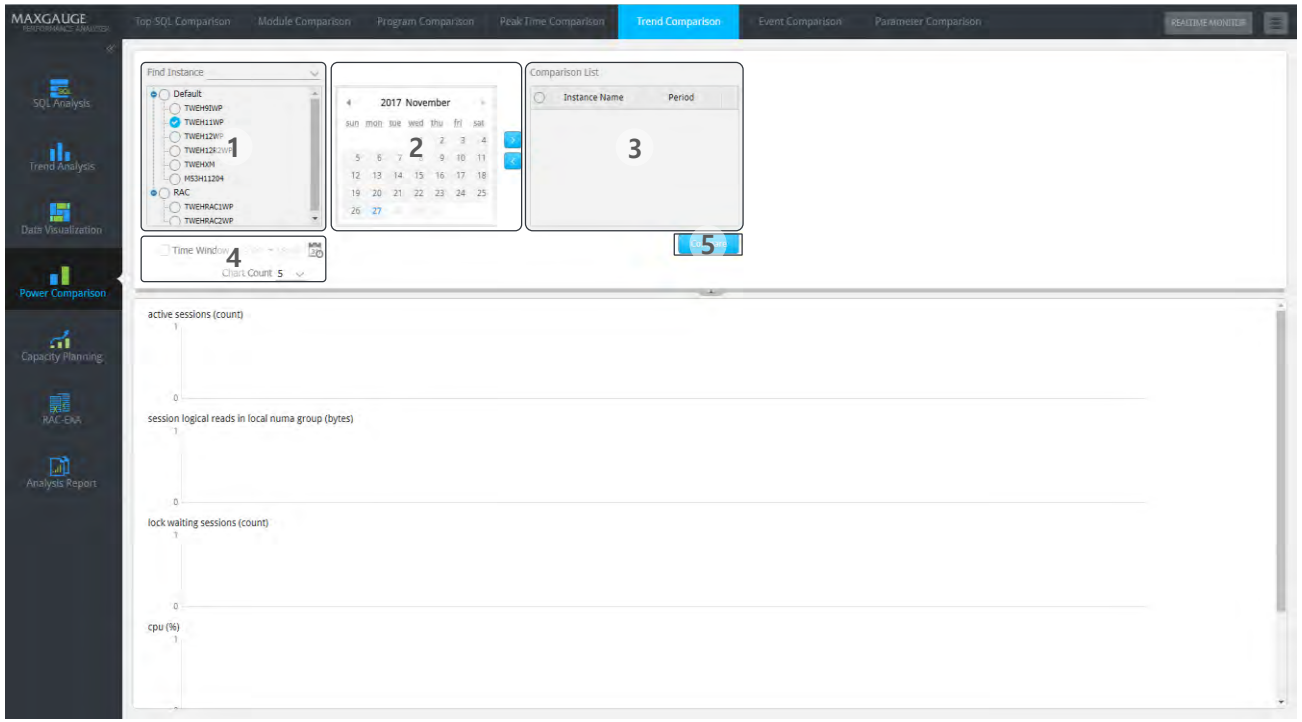
Trend Comparison

- 여러 개의 인스턴스, 혹은 여러 시점의 동일한 시간 축으로 **Trend Performance 간 비교 기능**이다.
- **최대 6일**까지의 로그에 대한 Stat, Wait, Ratio, OS Stat에 대한 지표에 대해 비교 분석 및 Active 세션에 대한 겹쳐진 이미지를 제공한다.
- Default 로 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 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 Comparison를 실행**합니다.

Trend Comparison의 사용 방법

» Trend Comparison 조회 화면

1 선택한 분석 대상의 정보 및 Trend 표시 색상을 나타냅니다.

2 Default 로 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 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 조회 화면

1 초기 화면의 Performance Trend Area 에서 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.

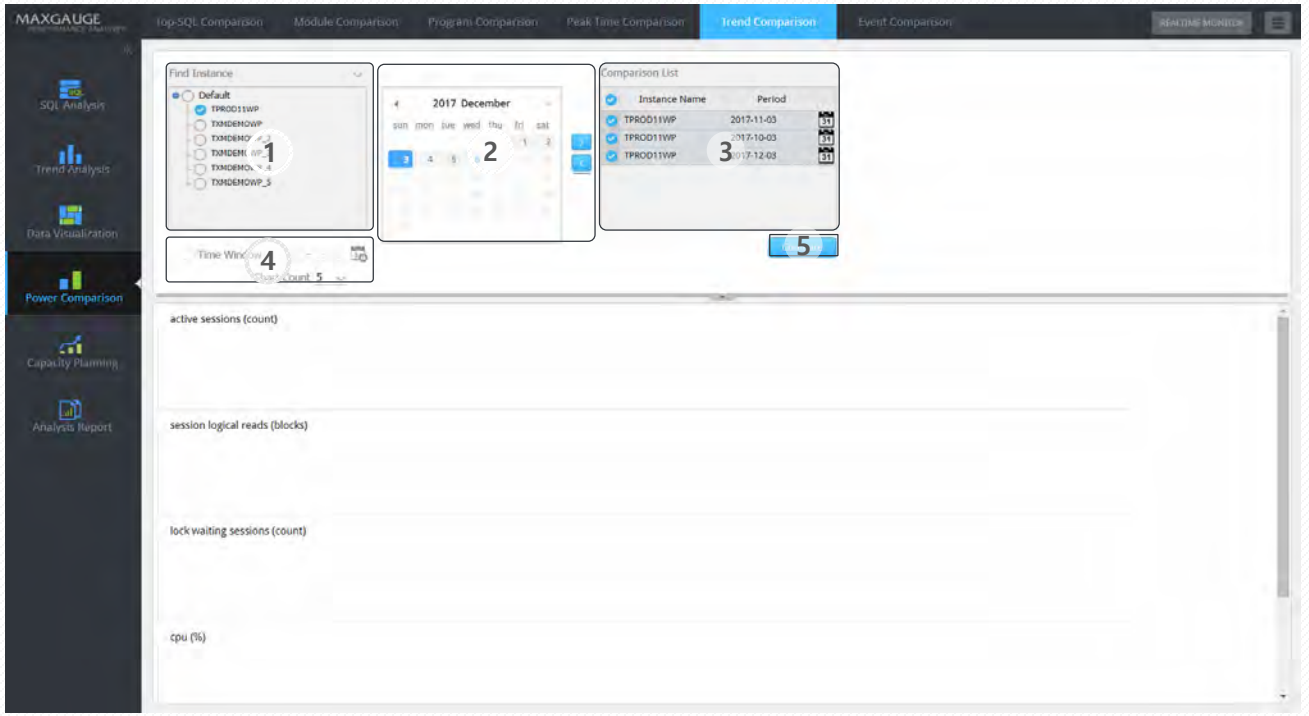
Storage 이전 작업 전후의 성능을 비교하고 싶어요.

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 전후의 성능을 비교하고 싶어요 ! "

Trend Comparison 검색 화면



✓ 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

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.**

" Storage 이전 작업 전후의 성능을 비교하고 싶어요 ! "

» Trend Comparison 결과 조회 화면

✓ 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 향상**된 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

» Trend Comparison 결과 조회 화면

✓ 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.

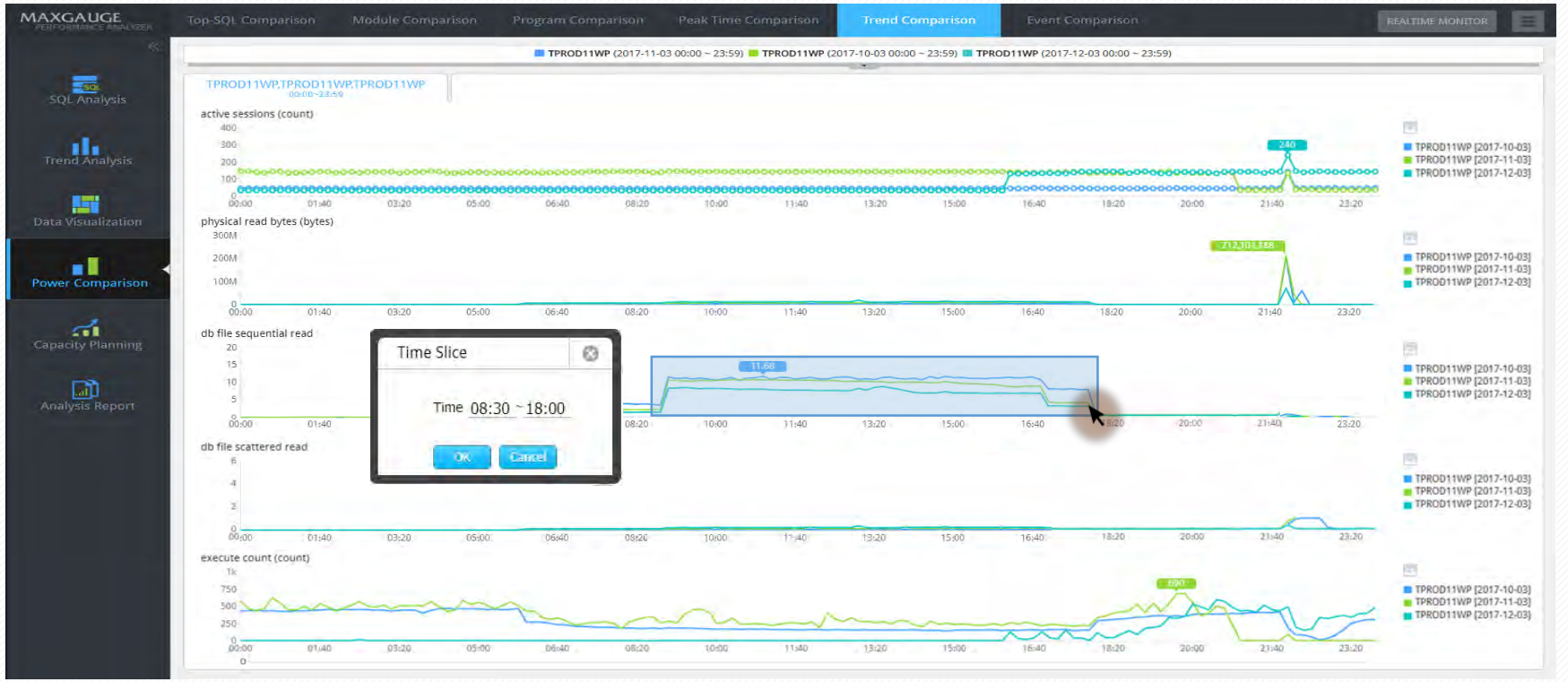
✓ 또한, Physical reads, db file scattered read 등 다른 I/O 오퍼레이션의 처리 시간의 경우, 서버 교체 전후 모두 비슷한 추이를 나타내고 있습니다.

" 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

» Trend Comparison 상세 분석 화면



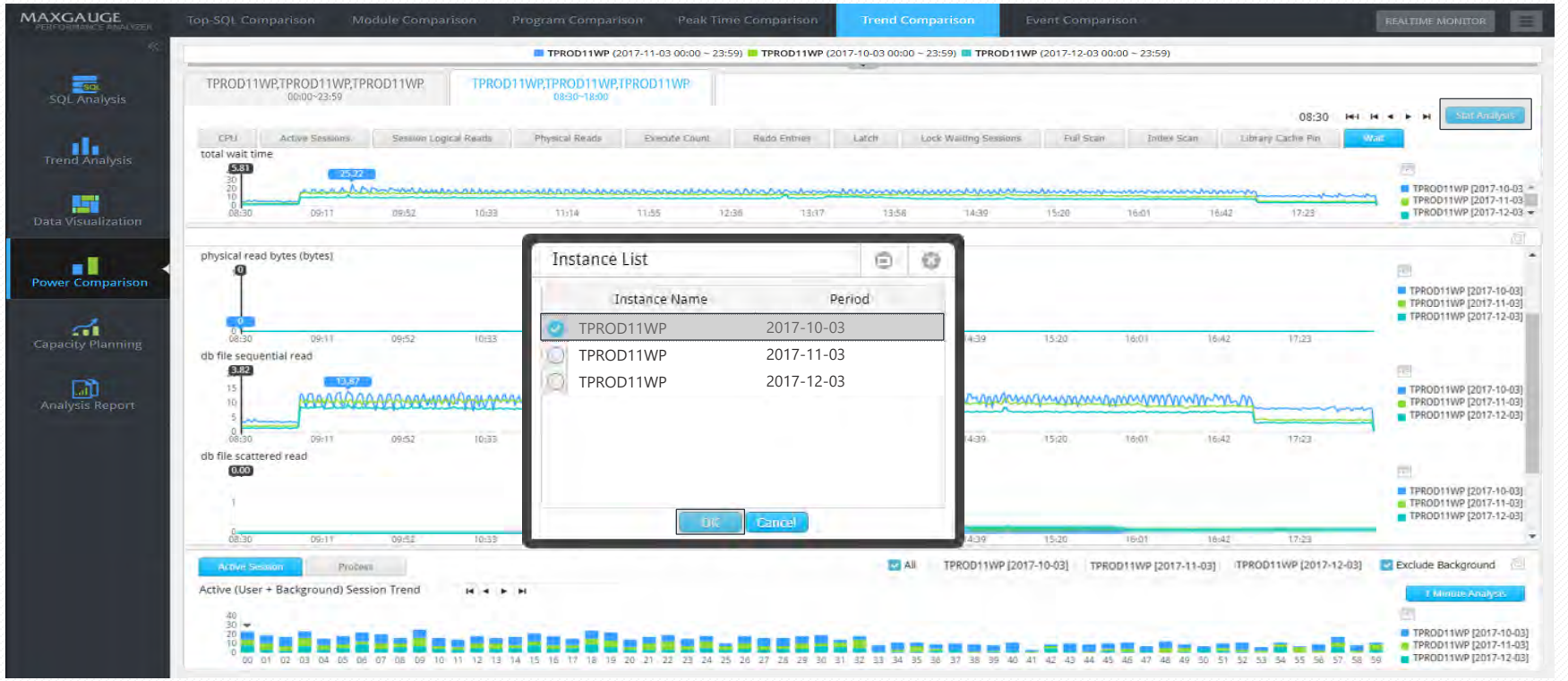
✓ 업무 시간대에 대한 상세 분석을 위해, 분석 구간을 마우스로 Drag 후 시간 설정하여 "OK" 버튼을 클릭합니다.

" 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

» Trend Comparison 상세 분석 화면



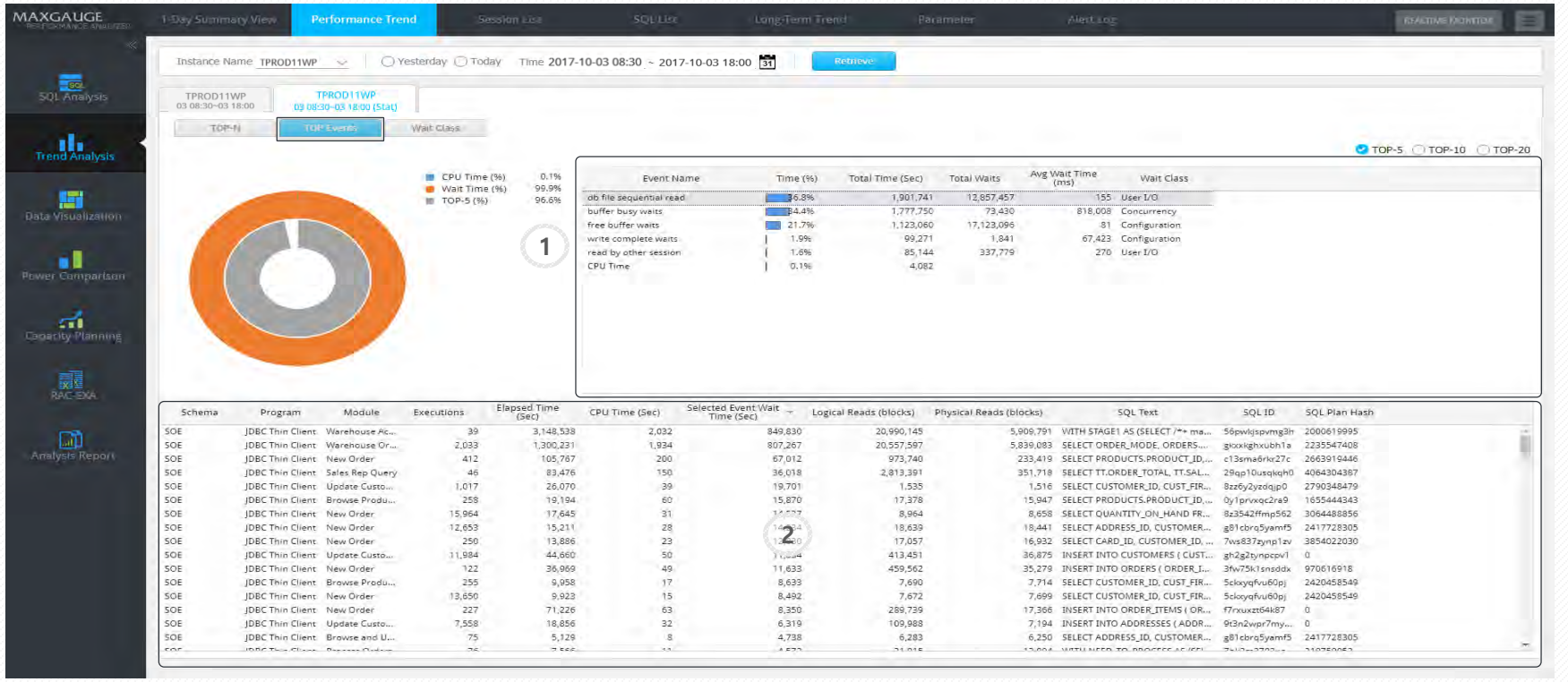
✓ 작업 전후의 Total wait time 지표를 비교해보고, 각 날짜별 분석을 위해 Stat Analysis 를 클릭하여 해당 인스턴스 명, 날짜를 선택합니다.

" 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

» Performance Trend(Stat) 화면



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

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

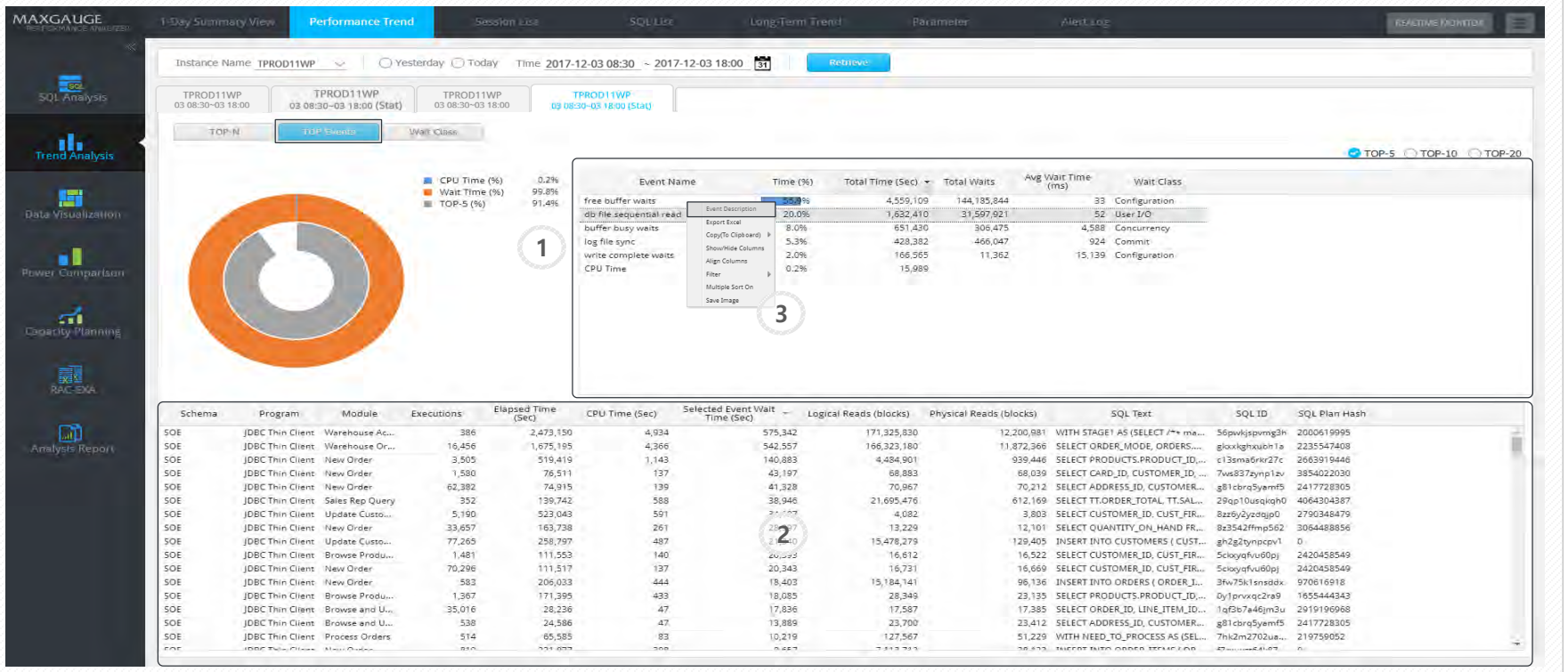
2 추출 된 TOP Events 중 대기 이벤트 선택 시, 2번 영역에 해당 대기 이벤트를 대기한 SQL 목록을 보여줍니다.

" 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

» Performance Trend(Stat) 화면



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.

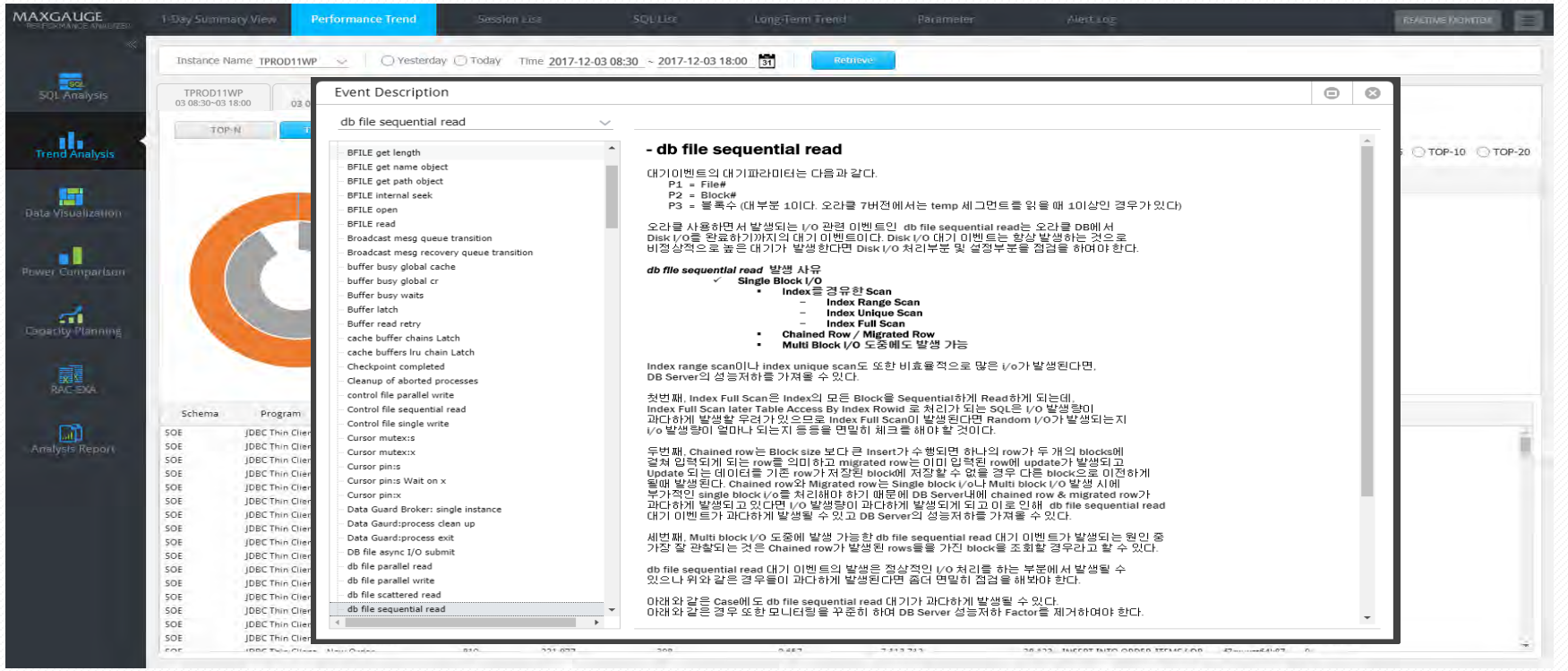
2 해당 이벤트에 대한 상세 정보를 얻고 싶을 경우, 오른쪽 마우스 클릭하여 **Event Description**으로 연계합니다.

" 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



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

실전 분석 사례 Case 2.

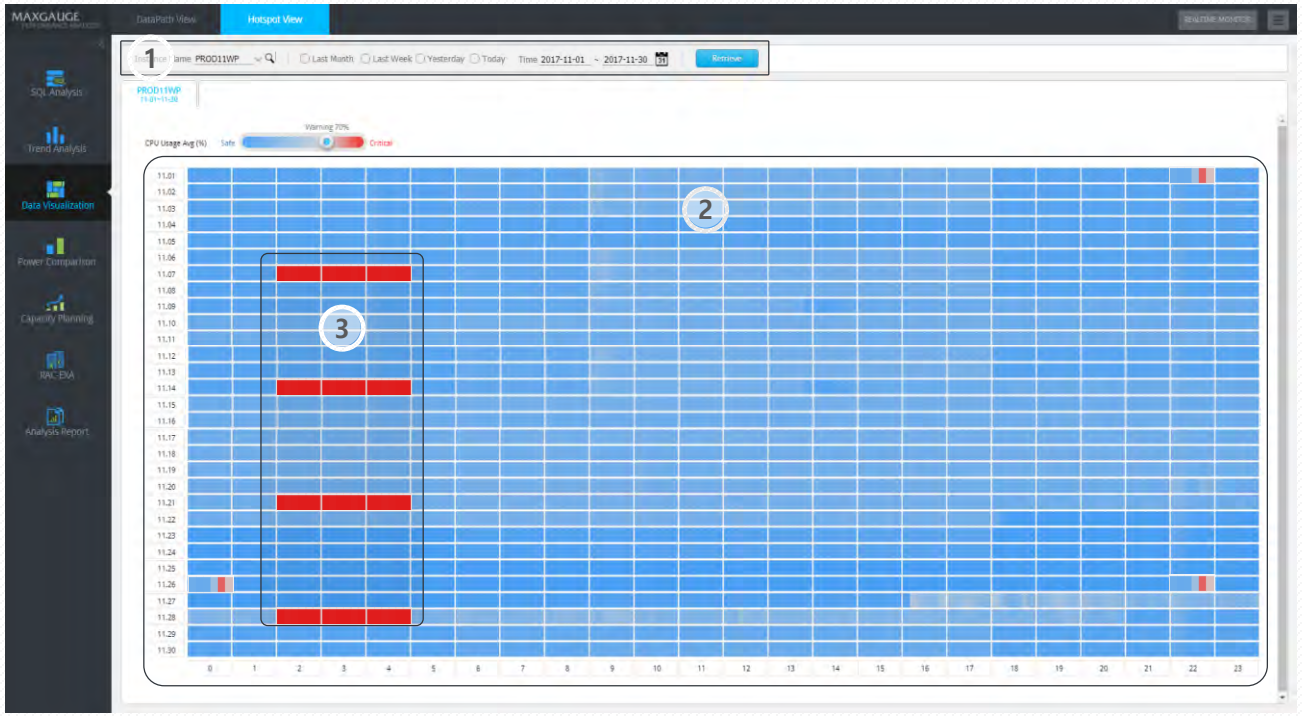
(매주 같은 시간)
반복적으로 장애를 발생시키는
쿼리를 찾고 싶어요.

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

" 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! "

HotSpot View 조회 화면



✓ 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.

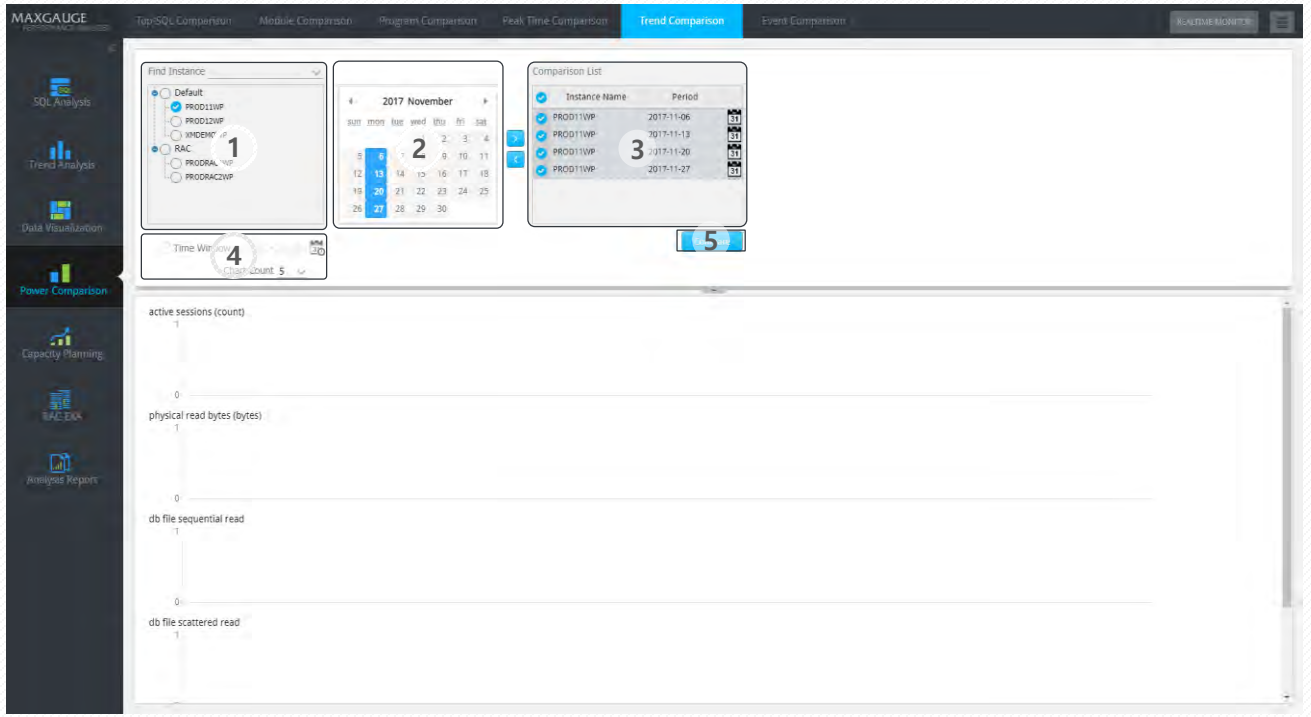
1 비교 대상 Instance (**PROD11WP**) 에 대하여 한달 간의 CPU 사용률 분석을 위한 구간 (**2017.11.01 ~ 2017.11.30**) 선택합니다.

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 대**에 CPU 사용률이 높은 것을 확인할 수 있습니다.

" 반복적으로 장애를 발생시키는 쿼리를 찾고 싶어요! "

Trend Comparison 검색 화면



✓ 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

» Trend Comparison 결과 조회 화면

✓ 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.

" 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

» Trend Comparison 결과 조회 화면

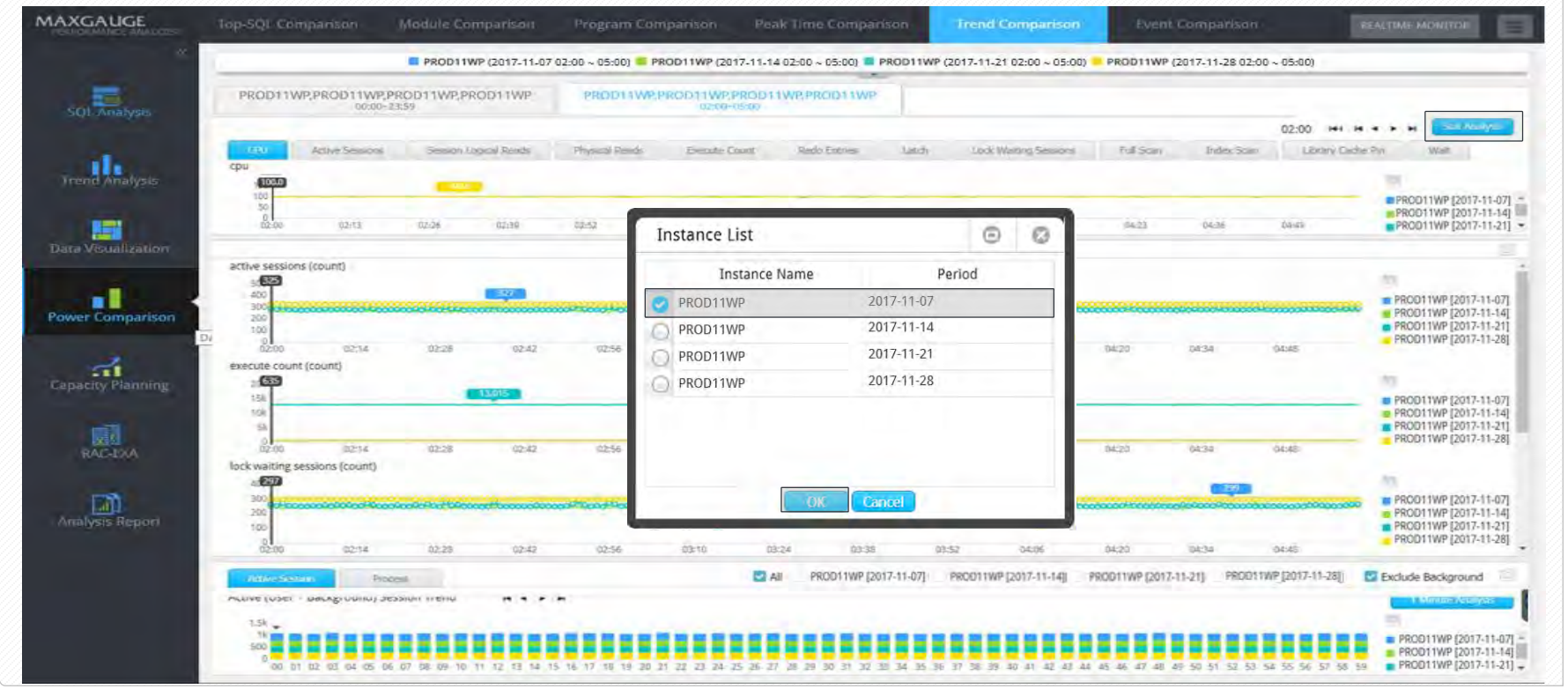
✓ 장애 발생 구간(**02:00 ~ 05:00**) 에 대한 상세 분석을 위해, 분석 구간을 마우스로 Drag 후 시간 설정하여 "OK" 버튼을 클릭합니다.

" 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

» Trend Comparison 상세 분석 화면



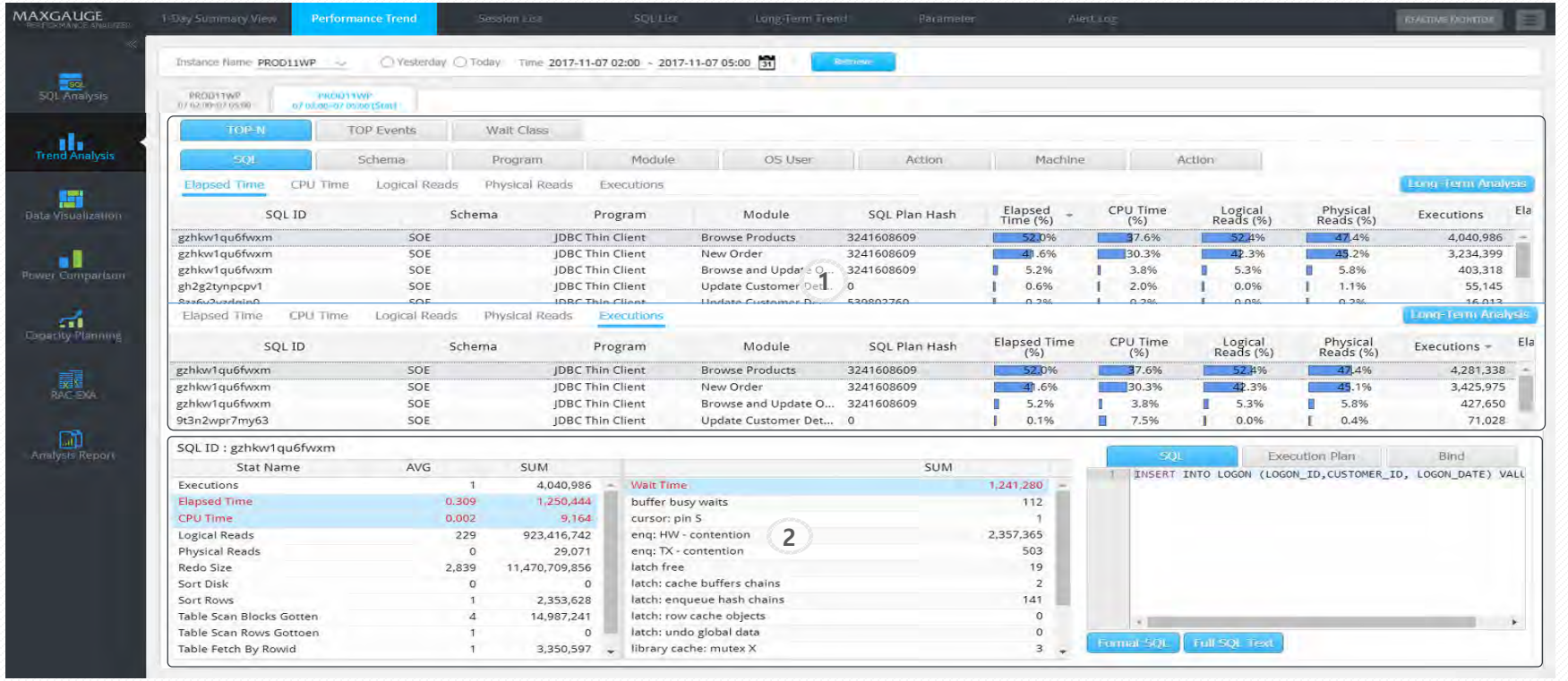
✓ 장애 발생 구간(**02:00 ~ 05:00**)에 대한 상세 분석을 위해 Stat Analysis 를 클릭하여 인스턴스 명, 날짜를 선택합니다.

" 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



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.

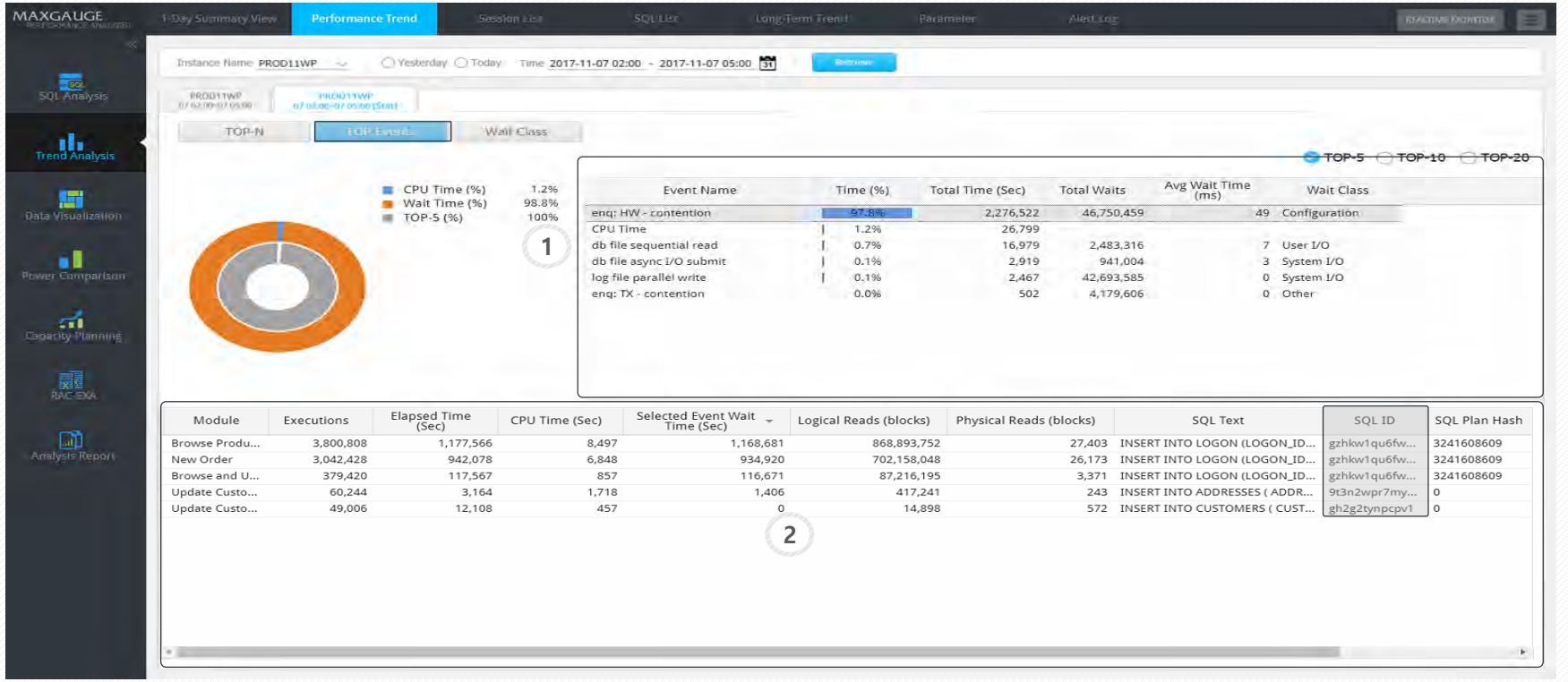
2 추출 된 TOP SQL 중 선택 시, 2번 영역에 해당 SQL에 대한 STAT 정보 등 상세한 정보를 확인할 수 있습니다.

" 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
     L Stat Analysis 연계
   - TOP SQL 정보 확인
     L TOP-N ▶ SQL
   - **TOP Event 및 문제 SQL 확인**
     L TOP Event ▶ SQL Detail 연계

» Performance Trend(Stat) 화면



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.

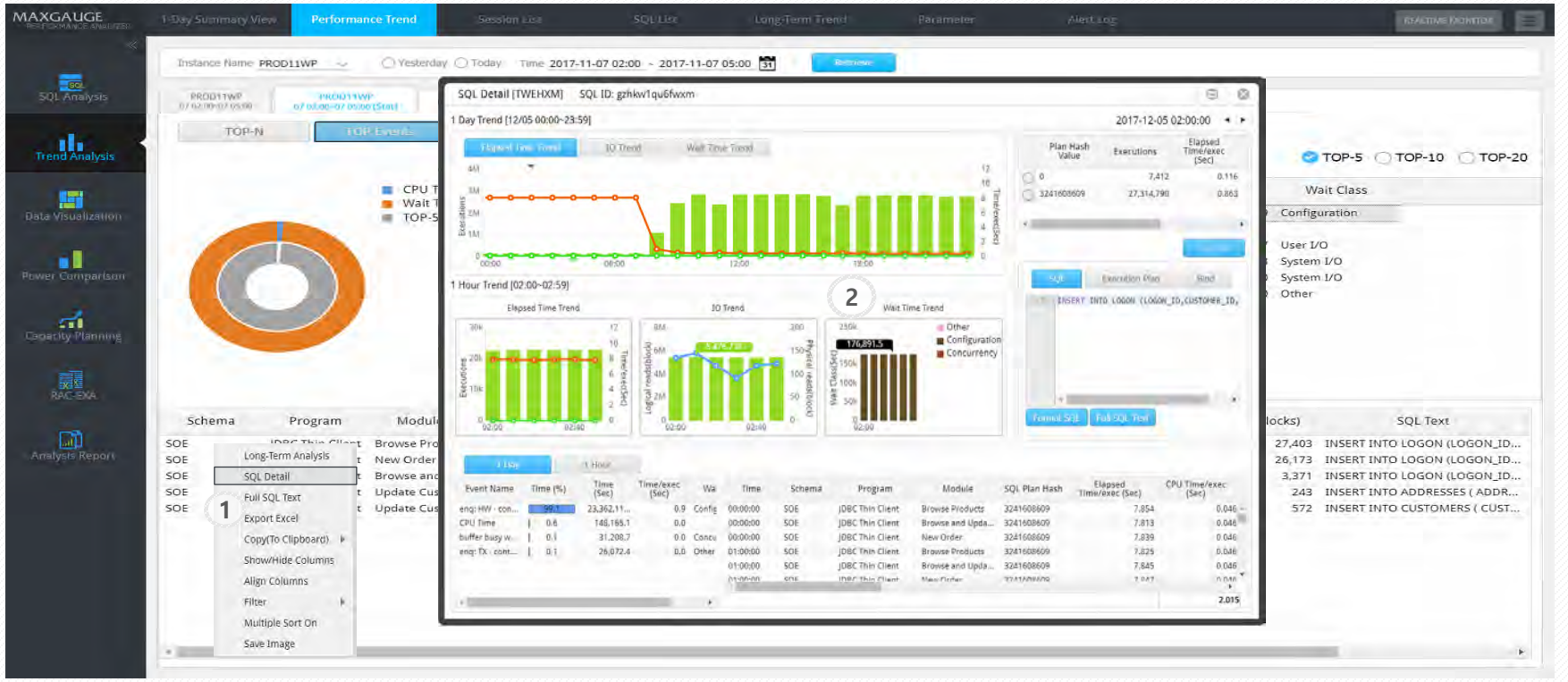
» 또한, 해당 이벤트를 대기하는 SQL이 TOP SQL과 동일한 것을 알 수 있습니다.

" 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
     L Stat Analysis 연계
   - TOP SQL 정보 확인
     L TOP-N ▶ SQL
   - **TOP Event 및 문제 SQL 확인**
     L TOP Event ▶ SQL Detail 연계

» Performance Trend(Stat) 화면



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

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

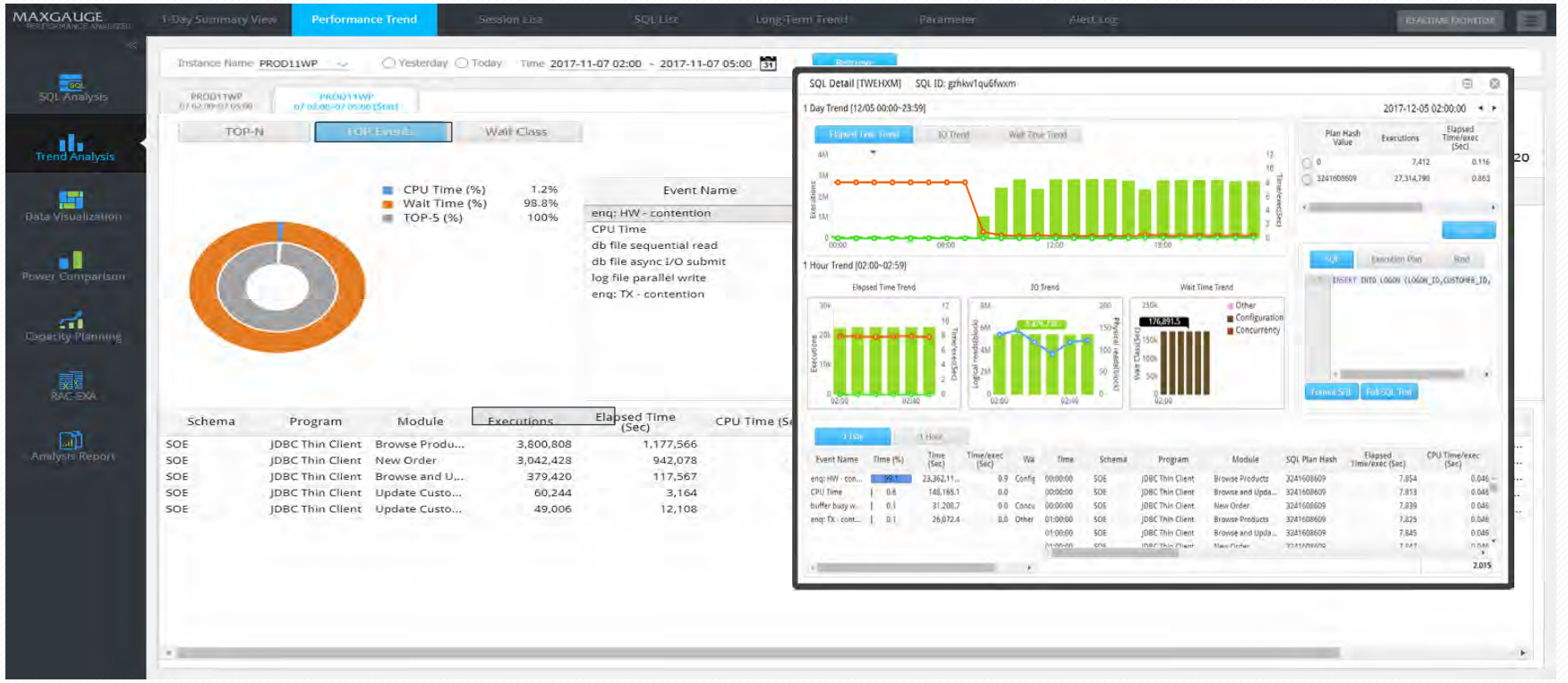
» 해당 SQL의 시간 별 수행 정보를 통해 성능 지연의 원인을 파악할 수 있습니다.

" 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**

» Performance Trend(Stat) 화면



2

1

1 선택한 SQL 의 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

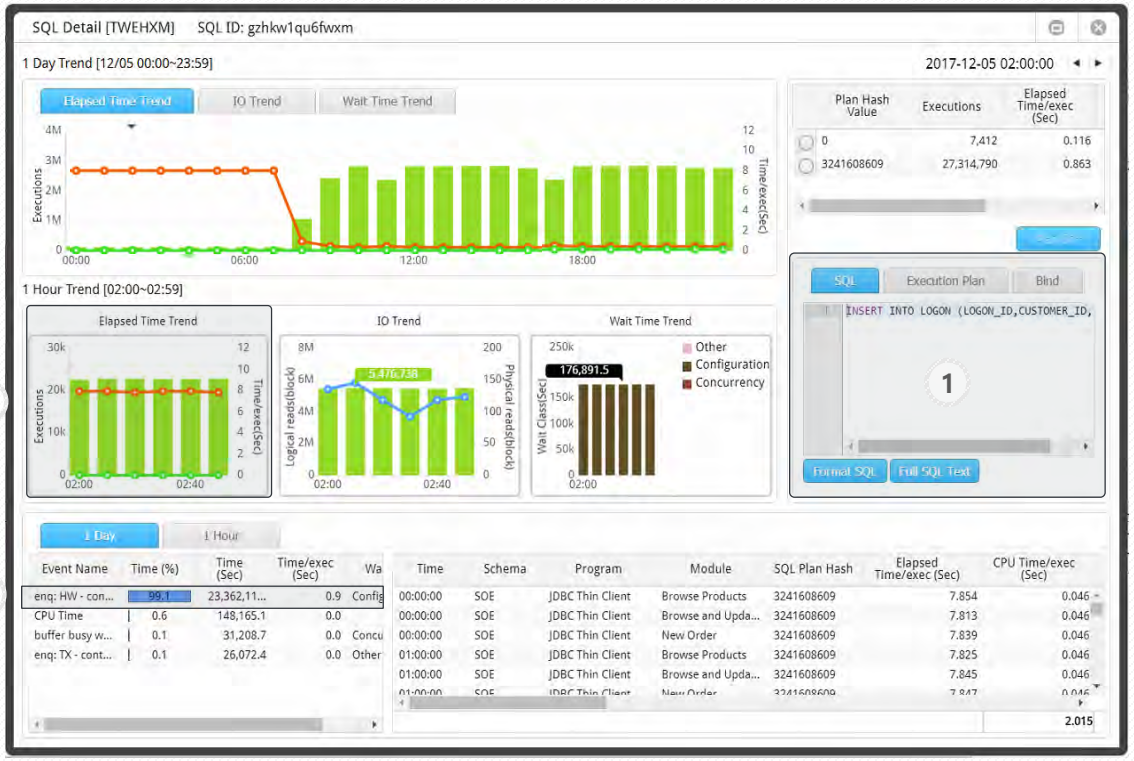
1. 장애 구간 파악

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**

» 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]

PRODUCT
DOCUMENTATION

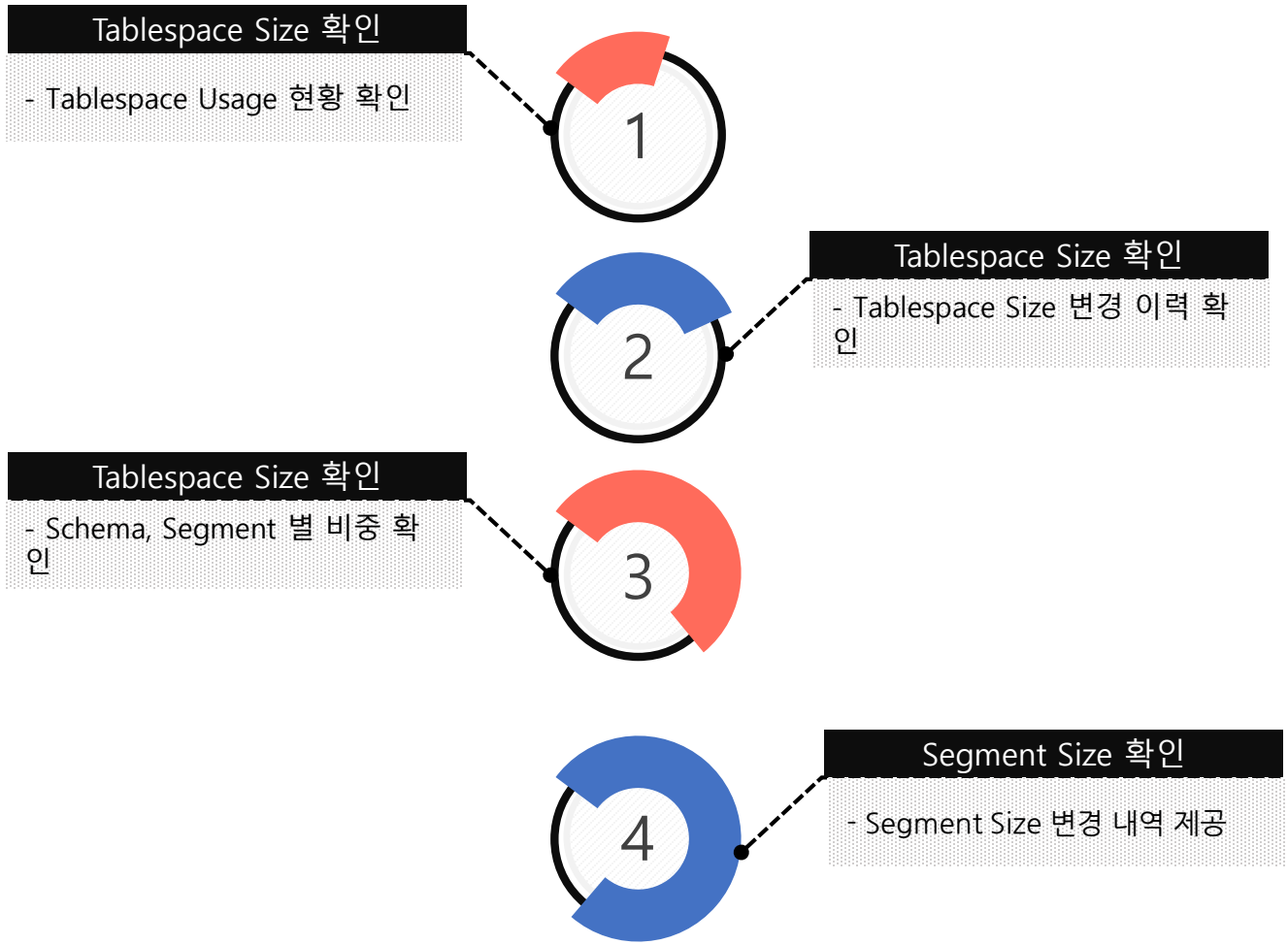
Practical Guide

Capacity Planning

Contents

Capacity Planning?

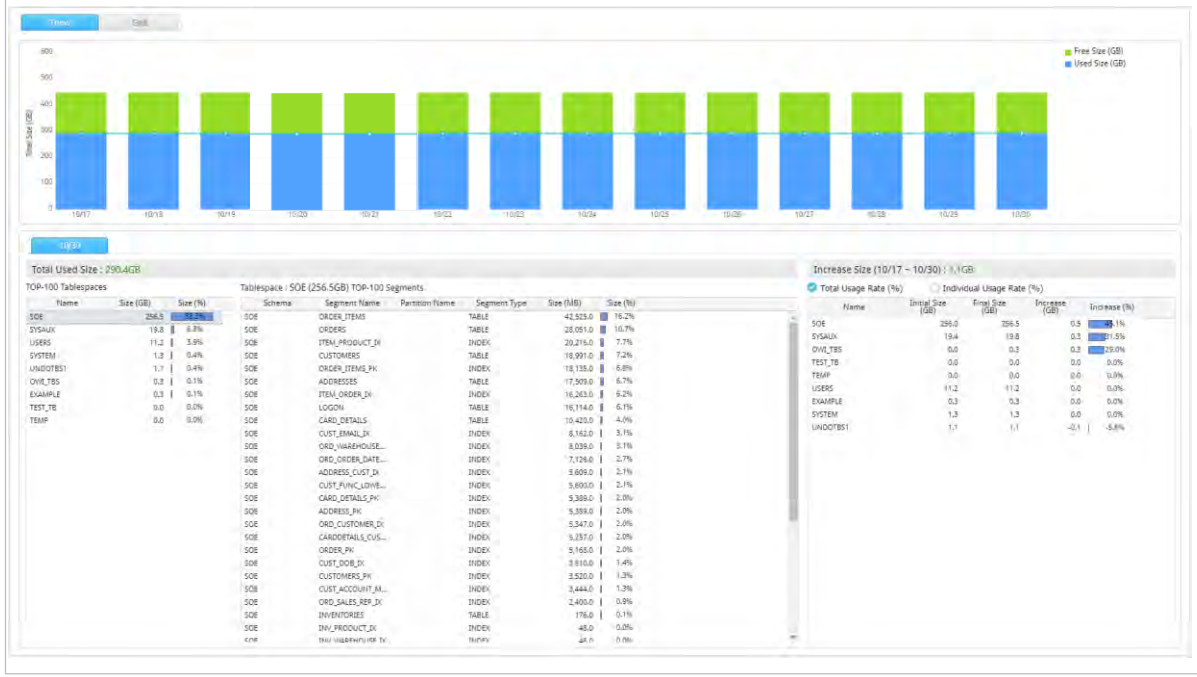
Capacity Planning의 주요 기능은 무엇인가요?



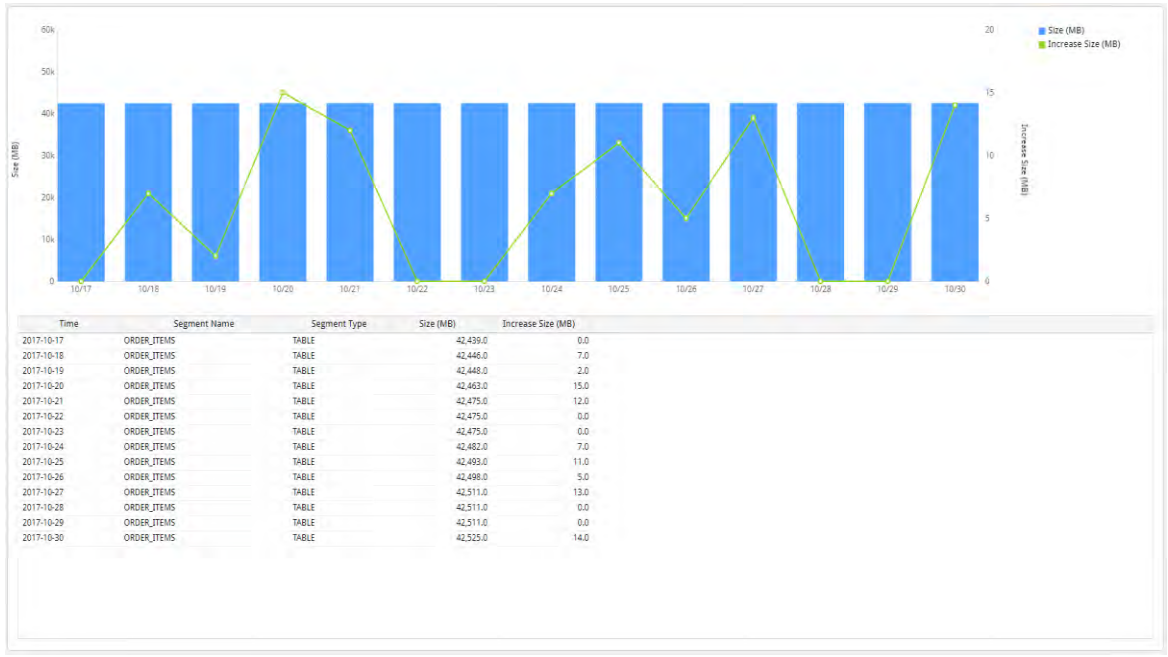
Capacity Planning

- 테이블스페이스의 **사용량** 및 **여유 공간**에 대한 추이를 제공한다.
- **사용량 증가 비율**이 높은 테이블 스페이스 내의 **Top-N 세그먼트 정보**를 제공한다.
- 세그먼트에 대한 **일별 크기** 및 **증가 크기**를 제공한다.
- 테이블스페이스와 세그먼트 정보를 일자 별 **Trend**와 **Grid**로 제공하기 때문에 효율적인 용량관리가 가능하다.
- 해당 기능은 Performance Analyzer ▸ **Capacity Planning** ▸ **Tablespace** 경로를 통해 사용할 수 있다.

Tablespace

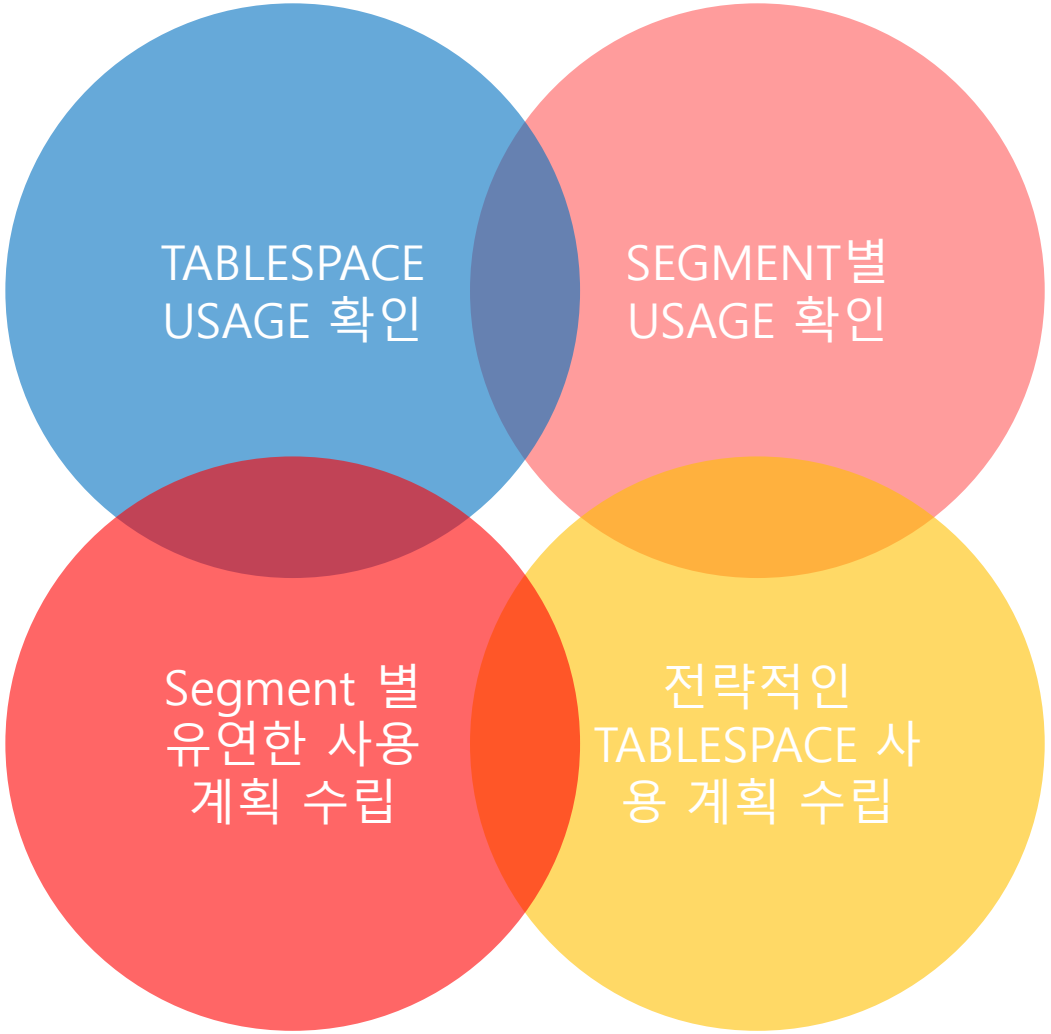


Segment



Time	Segment Name	Segment Type	Size (MB)	Increase Size (MB)
2017-10-17	ORDER_ITEMS	TABLE	42,439.0	0.0
2017-10-18	ORDER_ITEMS	TABLE	42,446.0	7.0
2017-10-19	ORDER_ITEMS	TABLE	42,448.0	2.0
2017-10-20	ORDER_ITEMS	TABLE	42,463.0	15.0
2017-10-21	ORDER_ITEMS	TABLE	42,475.0	12.0
2017-10-22	ORDER_ITEMS	TABLE	42,475.0	0.0
2017-10-23	ORDER_ITEMS	TABLE	42,475.0	0.0
2017-10-24	ORDER_ITEMS	TABLE	42,482.0	7.0
2017-10-25	ORDER_ITEMS	TABLE	42,493.0	11.0
2017-10-26	ORDER_ITEMS	TABLE	42,498.0	5.0
2017-10-27	ORDER_ITEMS	TABLE	42,511.0	13.0
2017-10-28	ORDER_ITEMS	TABLE	42,511.0	0.0
2017-10-29	ORDER_ITEMS	TABLE	42,511.0	0.0
2017-10-30	ORDER_ITEMS	TABLE	42,525.0	14.0

안정적인 TABLESPACE 관리가 가능합니다 !



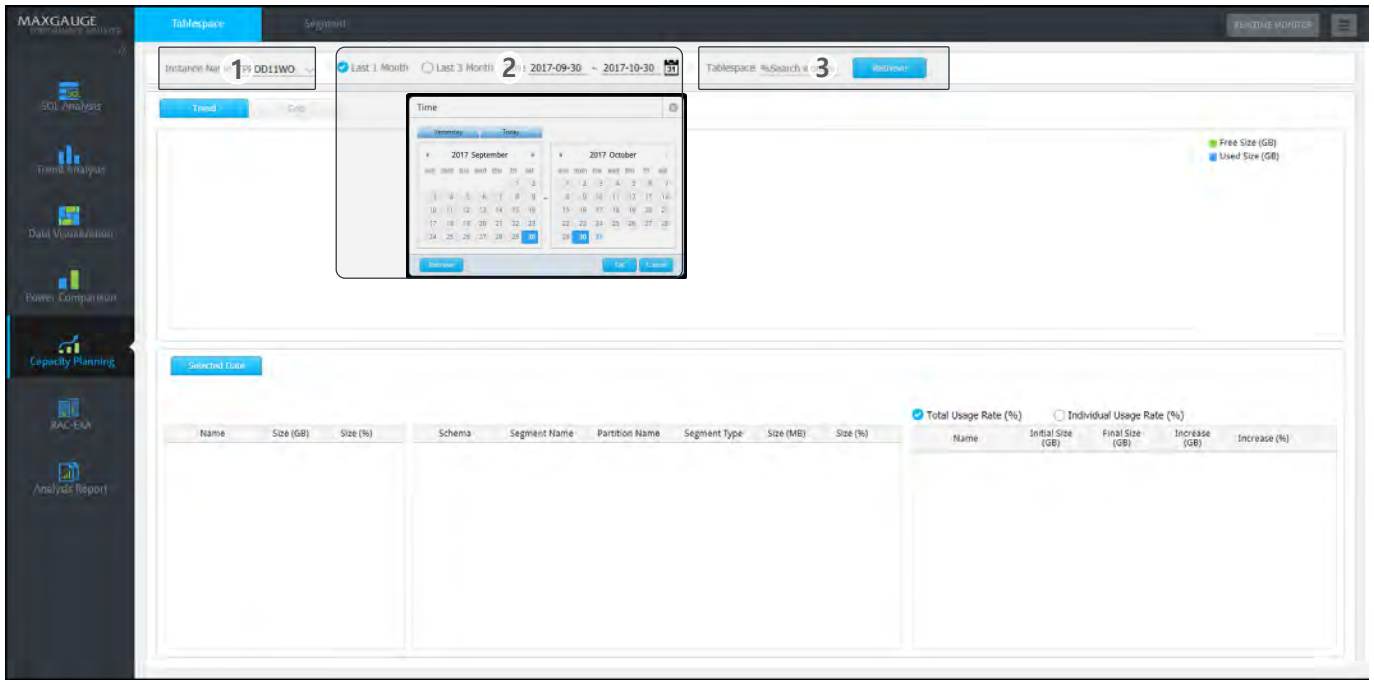
Capacity Planning의 활용

Capacity Planning의 사용 방법 (Tablespace)

Capacity Planning (Tablespace-Trend)

- 검색 조건 설정
- 데이터 분석

» Capacity Planning Tablespace-Trend 검색 화면



1 대상 **Instance**를 선택합니다.

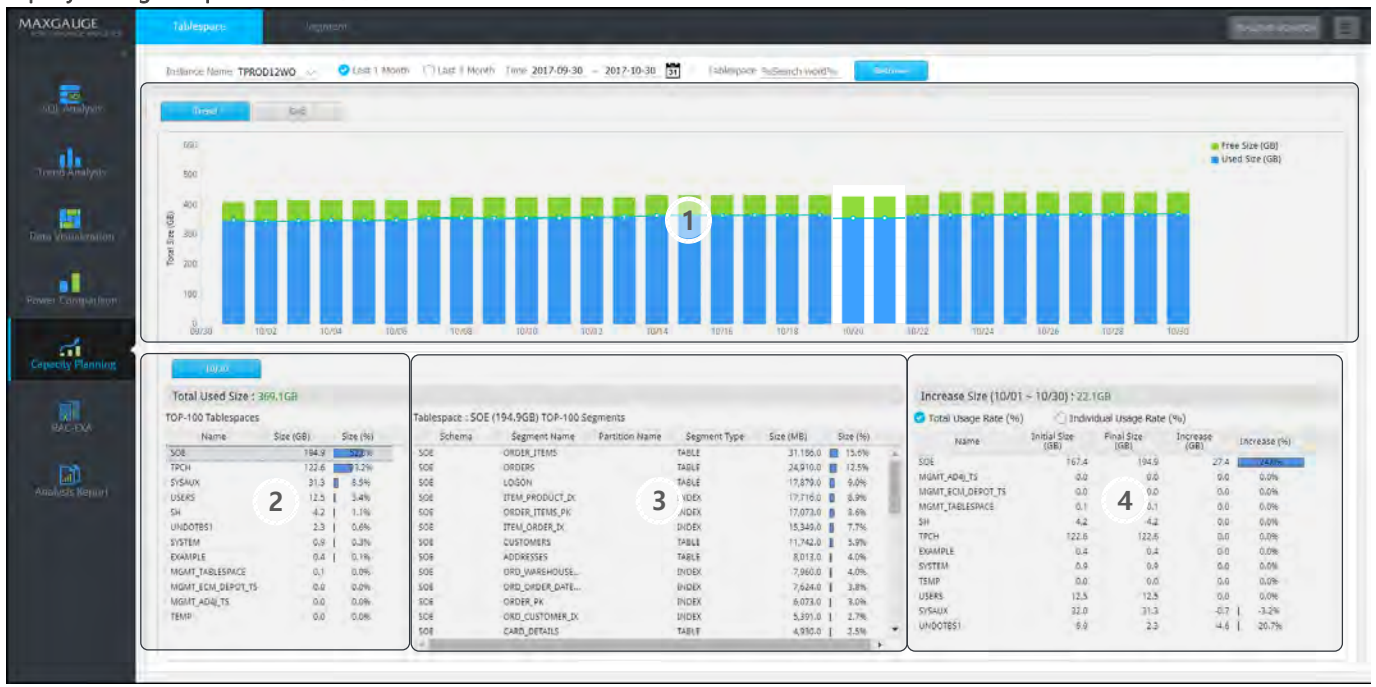
2 **Last 1 Month, Last 3 Month**로 기간을 선택 할 수 있으며, 달력 UI를 이용하여 **일(Day) 단위**로 기간을 선택 할 수 있습니다.

3 찾고자 하는 Tablespace의 **Full Name 또는 일부**를 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

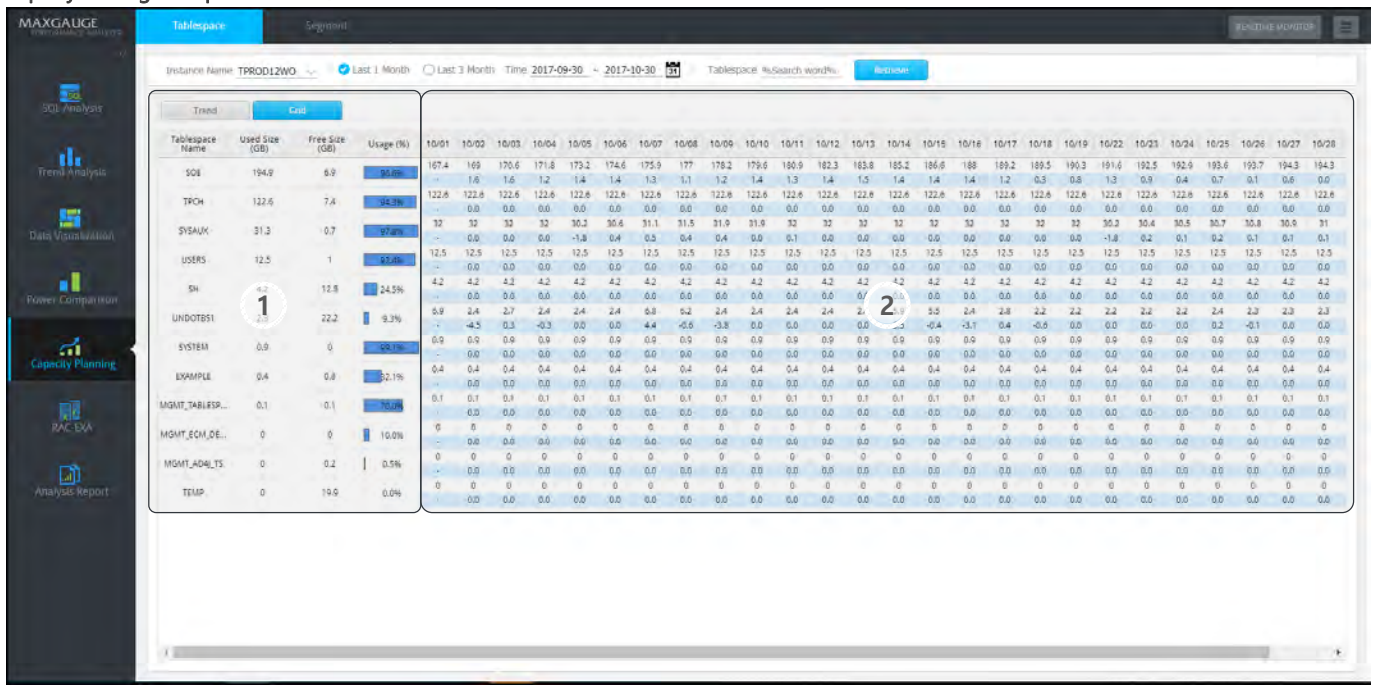
3 Top-100 Tablespace에서 선택한 Tablespace에 속해 있는 **Segment**를 Size기준으로 **Top-100 Segment** 정보를 제공합니다.

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 조회

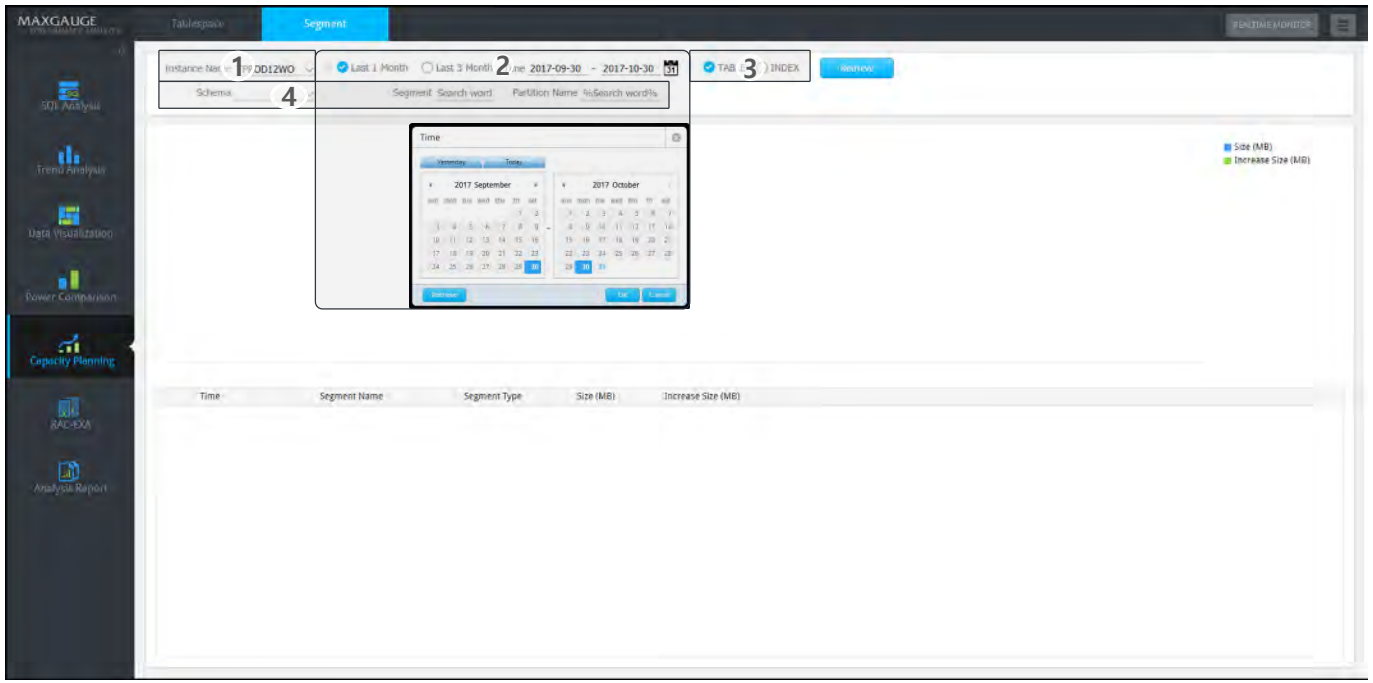


1 지정한 기간 마지막 일자의 **Tablespace 상세 정보**를 제공 합니다.
- Used Size(GB) : 마지막 날짜의 Tablespace 사용 공간입니다.
- Free Size(GB) : 마지막 날짜의 Tablespace 여유 공간 입니다.
- Usage(%) : Tablespace의 사용률 (FreeSize/UsedSize * 100) 입니다.

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 화면



1 대상 **Instance**를 선택합니다.

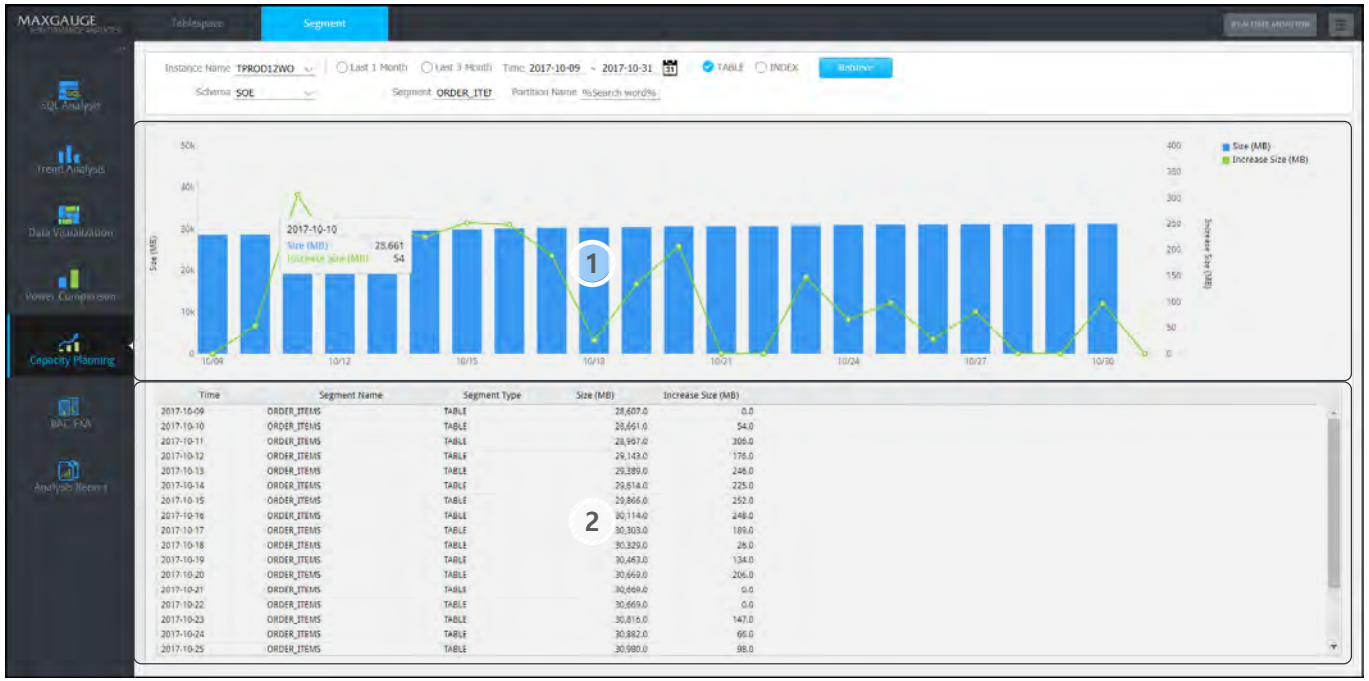
2 **Last 1 Month, Last 3 Month**로 기간을 선택 할 수 있으며, 달력 UI를 이용하여 **일(Day) 단위**로 기간을 선택 할 수 있습니다.

3 Segment 유형을 **TABLE,INDEX** 중 선택 할 수 있습니다.

4 **Schema와 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 조회 화면



1 선택한 Segment에 대한 **일별 크기 변화**를 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.

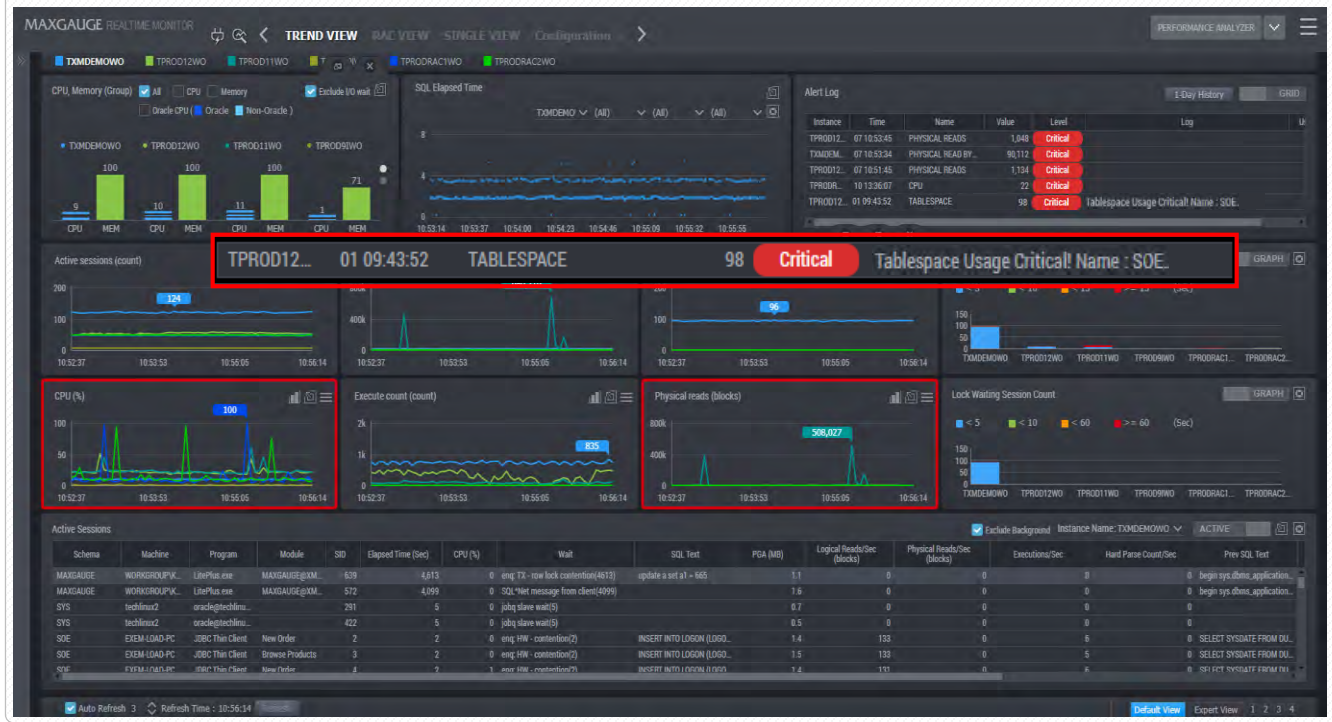
TABLESPACE ALARM이 발생 했는데 원인을 알고 싶어요.

STEP

1. 상황 발생 및 검색 조건 설정
   - 상황발생
   - 검색 조건 설정
2. 데이터 분석
   - 출력 데이터 확인
   - Top-100 Segments 확인

"TABLESPACE ALARM이 발생 했는데 원인을 알고 싶어요"

Real-Time Monitor Alarm 화면



✓ 상황 발생

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

✓ CRITICAL ALARM이 발생한 INSTANCE NAME, TABLESPACE NAME을 확인 합니다.

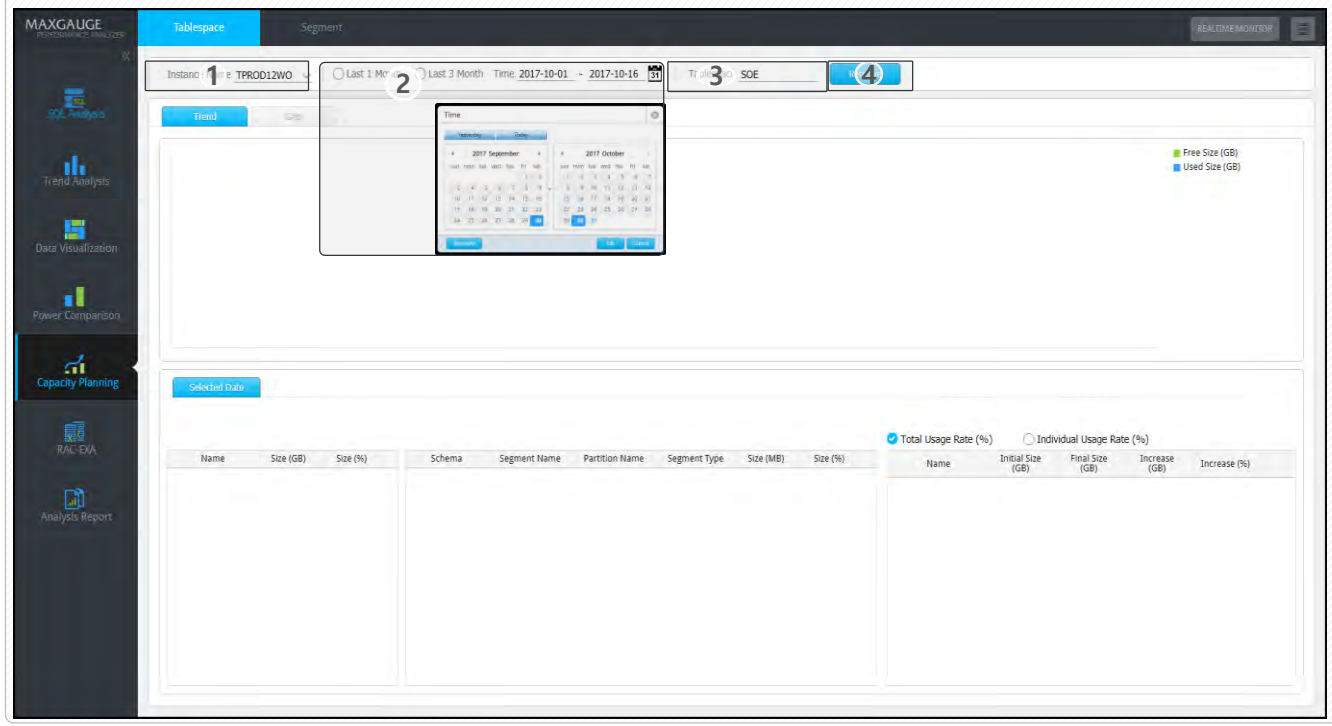
✓ 발생한 ALARM을 확인해 보면 INSTANCE NAME : "TPROD12WO" / TABLESPACE NAME : "SOE"를 확인 할 수 있습니다.

"TABLESPACE ALARM이 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

Capacity Planning-Tablespace 검색 화면



✓ **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

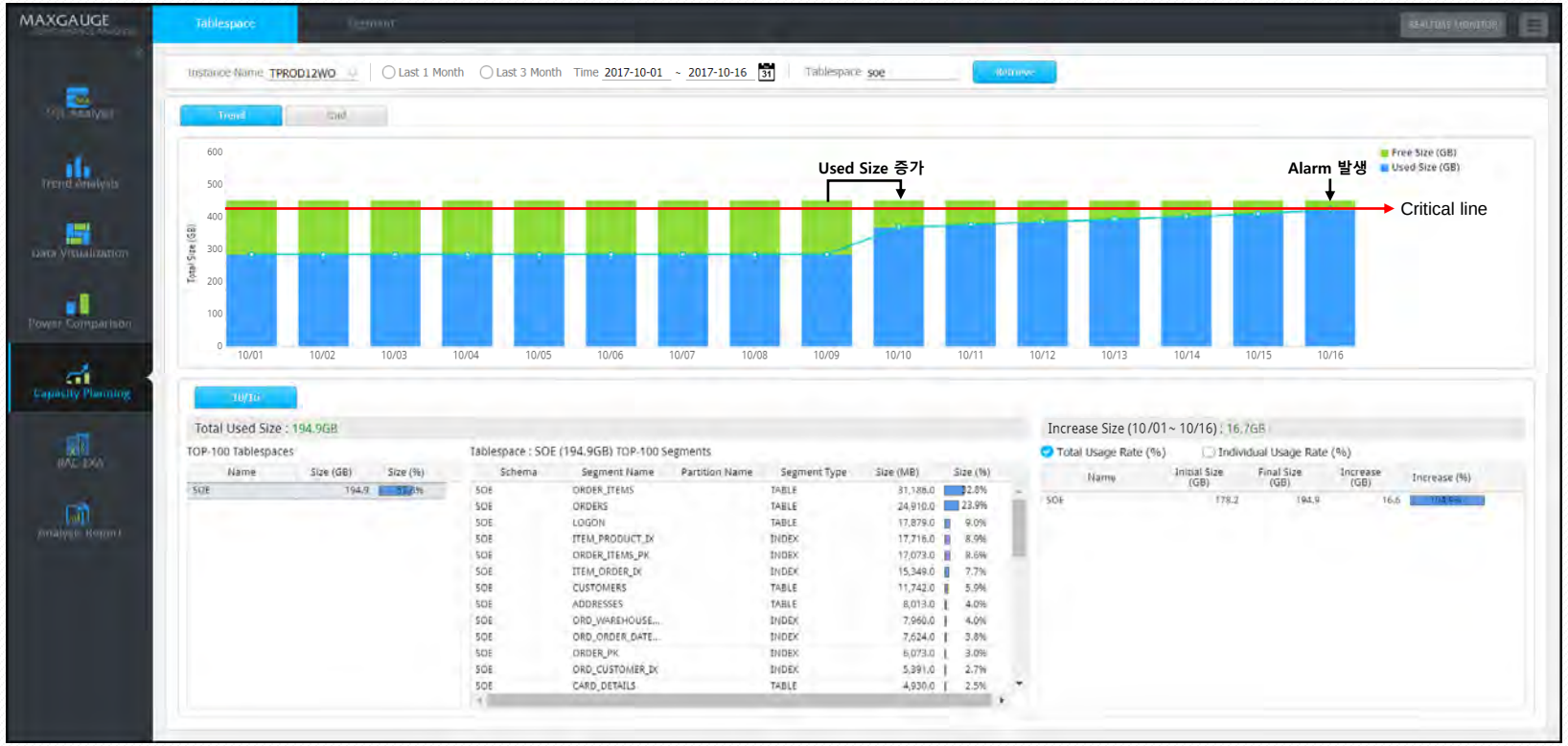
1. 대상 Instance (**TPROD12WO**) 를 선택합니다.
2. **Last 1 Month, Last 3 Month, 일(Day)** 단위로 기간을 선택 할 수 있습니다
3. Critical Alarm의 대상인 **SOE(Tablespace Name)를 Tablespace**란에 입력 합니다.
4. 위의 설정한 조건으로 **Tablespace를 조회합니다.**

"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

» Capacity Planning-Tablespace 조회 화면



✓ 막대 그래프를 통하여 일자 별 Size 증가 추이를 확인 합니다.

✓ "SOE" Tablespace 사용량이 10월 9일 이후로 점차 증가하여 **10월 16일**에 **"Tablespace Usage Critical" Alarm"**이 발생한 것을 확인할 수 있습니다.

✓ **Used Size가 급격히 증가 했던 날짜는 10월 10일이므로**, 10월 9일과 사용량을 비교하여 원인을 파악할 필요가 있습니다.

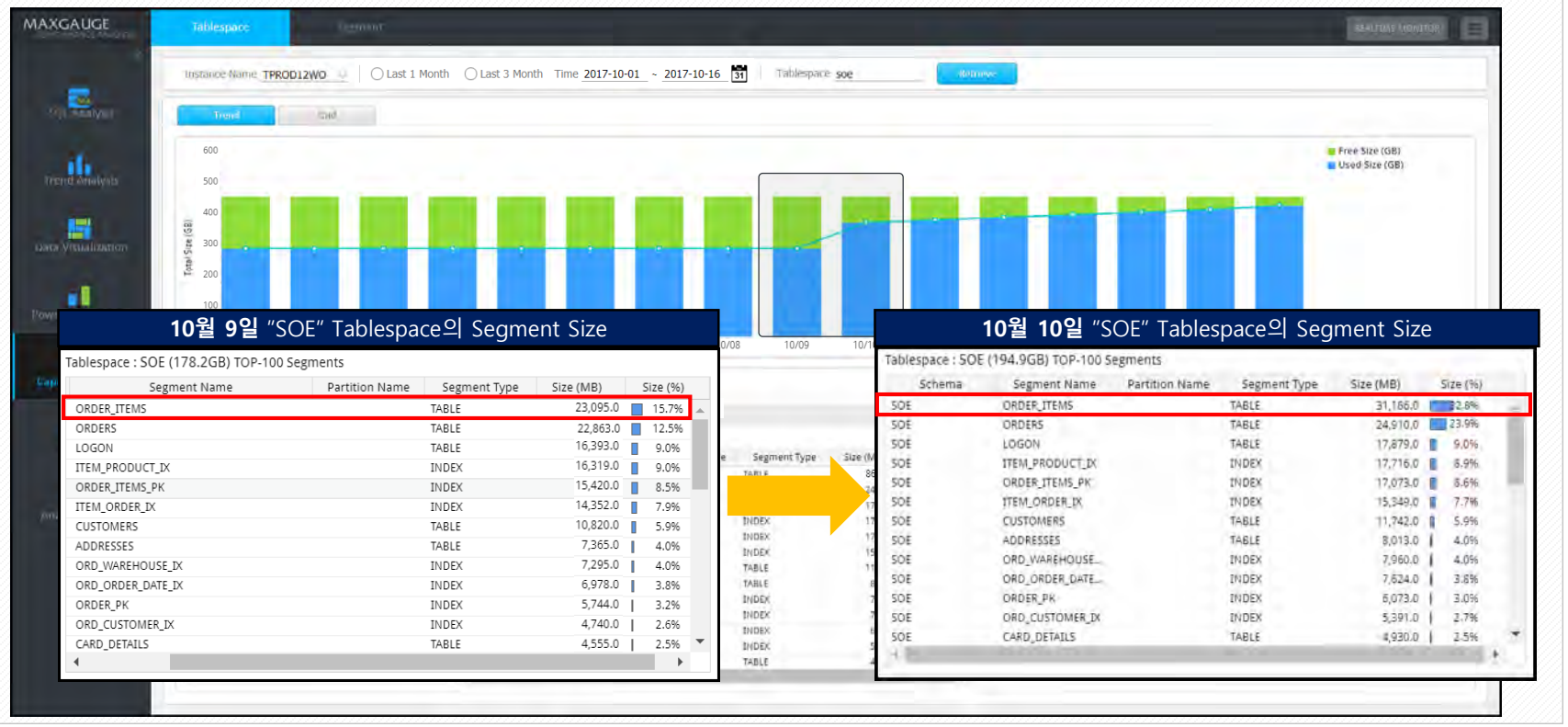
✓ Top-100 Segments에는 선택한 날짜의 **Segment Size 정보를 상세히 제공** 하므로 10월 9일과 **10월 10일**의 해당 Segment를 비교하여 파악할 필요가 있습니다.

"TABLESPACE ALARM이 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**

» Capacity Planning-Tablespace 조회 화면



✓ 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" Schema**를 확인 할 수 있으며, 해당 Segment에 대한 증가 이력을 살펴 볼 필요가 있습니다.

실전 분석 사례 Case 2.

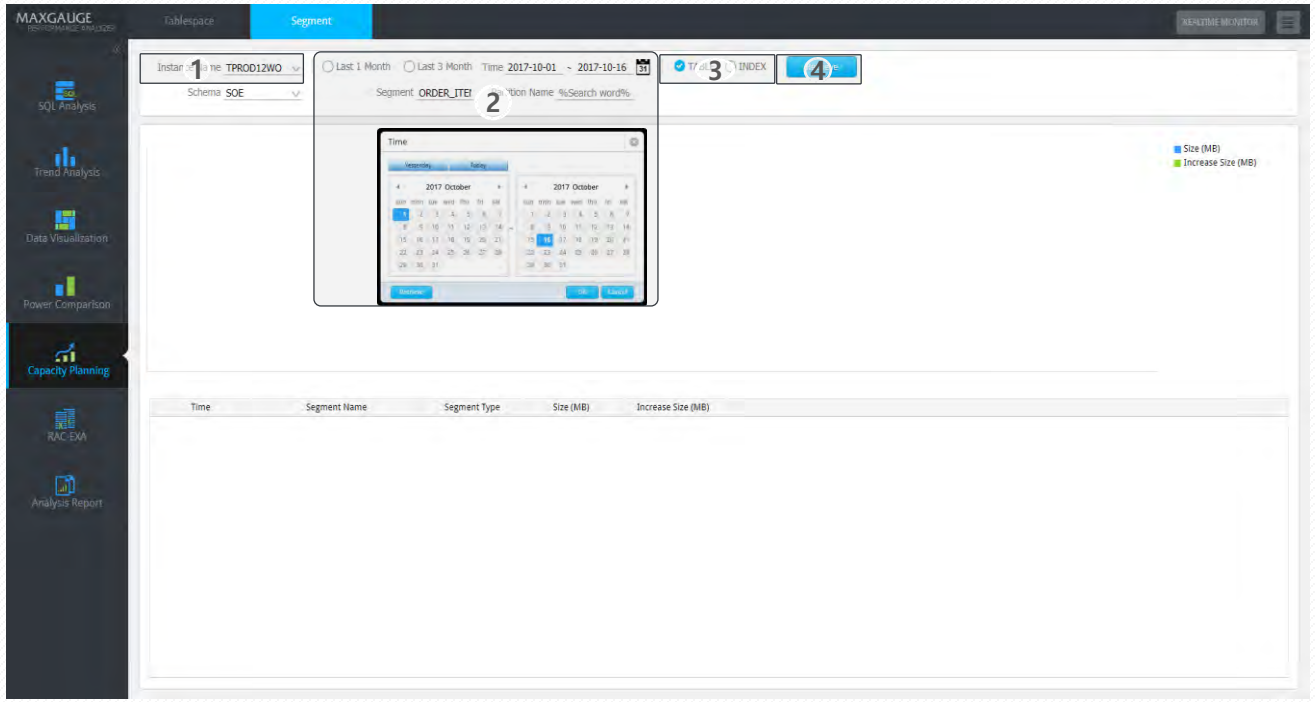
특정 Segment Size가 크게 증가한 시점에 자주 수행된 SQL을 알고 싶어요.

"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

Capacity Planning-Tablespace 검색 화면



✓ **시나리오**

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

✓ **목표**

Case1 에서 Tablespace 증가에 큰 영향을 준 Segments를 찾았다.
이제 Segment Size 증가 이력을 살펴 보면서 급격히 증가한 시점을 찾고, 영향도가 높은 SQL을 추적하자.

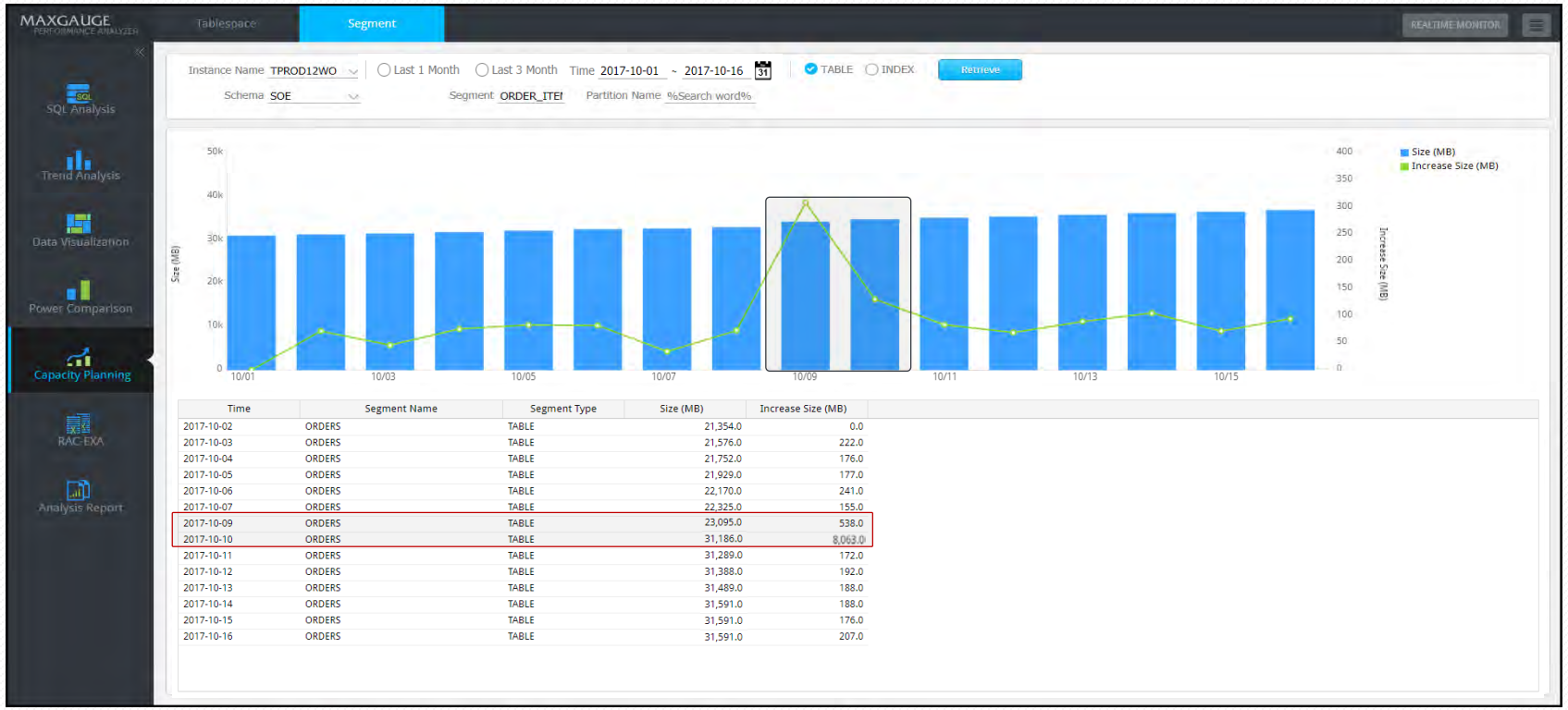
1. 대상 Instance (**TPROD12WO**) 를 선택합니다.
2. **Last 1 Month, Last 3 Month, 일(Day)** 단위로 기간을 선택 할 수 있습니다
3. **Segment Type**을 선택 합니다.
4. **Schema, Segment는 필 수 입력**해야 하므로 입력 후 Segment 정보를 조회 합니다.

"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

» Capacity Planning-Segment 조회 화면



Time	Segment Name	Segment Type	Size (MB)	Increase Size (MB)
2017-10-02	ORDERS	TABLE	21,354.0	0.0
2017-10-03	ORDERS	TABLE	21,576.0	222.0
2017-10-04	ORDERS	TABLE	21,752.0	176.0
2017-10-05	ORDERS	TABLE	21,929.0	177.0
2017-10-06	ORDERS	TABLE	22,170.0	241.0
2017-10-07	ORDERS	TABLE	22,325.0	155.0
2017-10-09	ORDERS	TABLE	23,095.0	538.0
2017-10-10	ORDERS	TABLE	31,186.0	8,063.0
2017-10-11	ORDERS	TABLE	31,289.0	172.0
2017-10-12	ORDERS	TABLE	31,388.0	192.0
2017-10-13	ORDERS	TABLE	31,489.0	188.0
2017-10-14	ORDERS	TABLE	31,591.0	188.0
2017-10-15	ORDERS	TABLE	31,591.0	176.0
2017-10-16	ORDERS	TABLE	31,591.0	207.0

✓ 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.

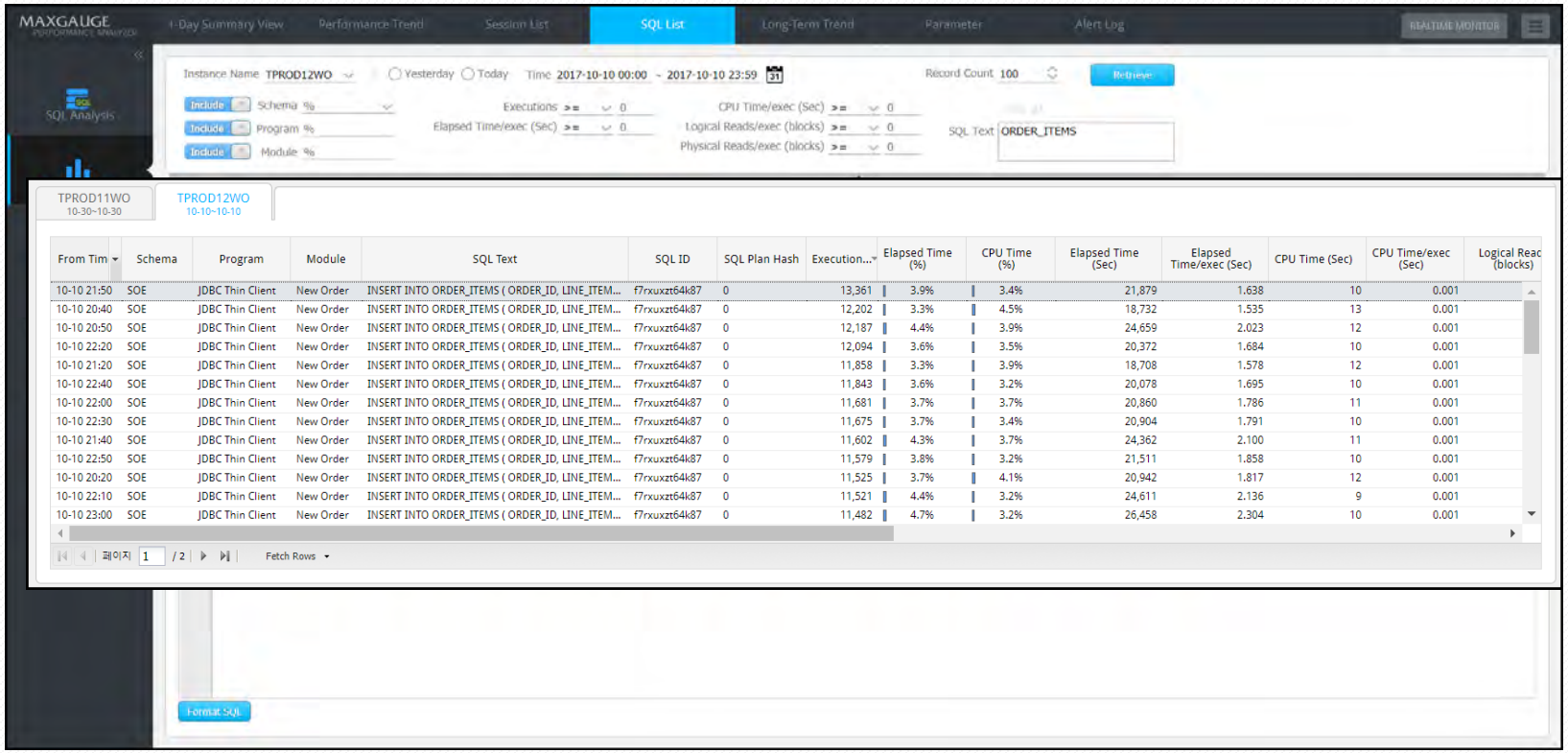
✓ 10월 09일과 10월 10일 사이에 Size가 급격히 증가하는데 **영향을 끼친 "SQL"**이 무엇인지 확인 해 볼 필요가 있습니다.

"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



From Tim	Schema	Program	Module	SQL Text	SQL ID	SQL Plan Hash	Execution...	Elapsed Time (%)	CPU Time (%)	Elapsed Time (Sec)	Elapsed Time/exec (Sec)	CPU Time (Sec)	CPU Time/exec (Sec)	Logical Reac (blocks)
10-10 21:50	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	13,361	3.9%	3.4%	21,879	1.638	10	0.001	
10-10 20:40	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	12,202	3.3%	4.5%	18,732	1.535	13	0.001	
10-10 20:50	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	12,187	4.4%	3.9%	24,659	2.023	12	0.001	
10-10 22:20	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	12,094	3.6%	3.5%	20,372	1.684	10	0.001	
10-10 21:20	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,858	3.3%	3.9%	18,708	1.578	12	0.001	
10-10 22:40	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,843	3.6%	3.2%	20,078	1.695	10	0.001	
10-10 22:00	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,681	3.7%	3.7%	20,860	1.786	11	0.001	
10-10 22:30	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,675	3.7%	3.4%	20,904	1.791	10	0.001	
10-10 21:40	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,602	4.3%	3.7%	24,362	2.100	11	0.001	
10-10 22:50	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,579	3.8%	3.2%	21,511	1.858	10	0.001	
10-10 20:20	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,525	3.7%	4.1%	20,942	1.817	12	0.001	
10-10 22:10	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,521	4.4%	3.2%	24,611	2.136	9	0.001	
10-10 23:00	SOE	JDBC Thin Client	New Order	INSERT INTO ORDER_ITEMS (ORDER_ID, LINE_ITEM...	f7rxuxzt64k87	0	11,482	4.7%	3.2%	26,458	2.304	10	0.001	

페이지 1 / 2 Fetch Rows

✓ 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]

PRODUCT
DOCUMENTATION

Contents

Script Alert ?

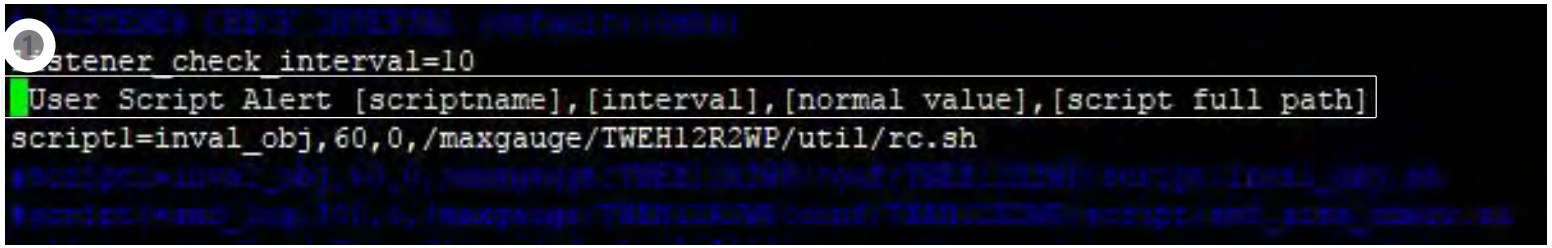
Script Alert 는 언제 쓰나요?

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

Script Alert 의 활용

RTS User Script Alert 의 사용 방법

» 화면구성



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 User Script 추가 화면

```
[weheheh:/maxgauge/TWEH12R2WP/util]$ls -lrt
total 224
drwxr-xr-x. 3 maxgauge oinstall   4096 Nov 18  2016 ZnPlug
-rwx------. 1 maxgauge oinstall   1267 Nov 18  2016 MXG DiskInfo.vbs
drwxr-xr-x. 6 maxgauge oinstall   4096 Nov 18  2016 monbar
-rwx------. 1 maxgauge oinstall    153 Nov 18  2016 get base.sql
-rwx------. 1 maxgauge oinstall    192 Nov 18  2016 get base.sh
-rwx------. 1 maxgauge oinstall    152 Nov 18  2016 get adm.sql
-rwx------. 1 maxgauge oinstall    188 Nov 18  2016 get adm.sh
drwxr-xr-x. 2 maxgauge oinstall   4096 Apr 27  2017 sh load
drwxr-xr-x. 2 maxgauge oinstall   4096 Sep 14 15:43 sh setup
-rw-r--r--. 1 maxgauge oinstall   1124 Nov 13 17:08 ts.sh
-rw-r--r--. 1 maxgauge oinstall   1276 Nov 13 17:09 ts.sql
-rwxr-xr-x. 1 maxgauge oinstall   1004 Nov 14 14:14 resource limit.sql
-rw-r--r--. 1 maxgauge oinstall    403 Nov 14 16:20 tmp11.txt.old
-rw-r--r--. 1 maxgauge oinstall    402 Nov 14 16:20 tmp22.txt.old
-rw-r--r--. 1 maxgauge oinstall      7 Nov 14 18:37 rc1.sh
-rwxr-xr-x. 1 maxgauge oinstall   1348 Nov 14 19:31 rc.sh
-rw-r--r--. 1 maxgauge oinstall    402 Nov 15 08:52 tmp22.txt
-rw-r--r--. 1 maxgauge oinstall    484 Nov 15 08:52 tmp11.txt
-rw-r--r--. 1 maxgauge oinstall 149056 Nov 15 08:52 shlog.log
[weheheh:/maxgauge/TWEH12R2WP/util]$
```

✓ **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 User Script 추가 화면

✓ **시나리오**

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

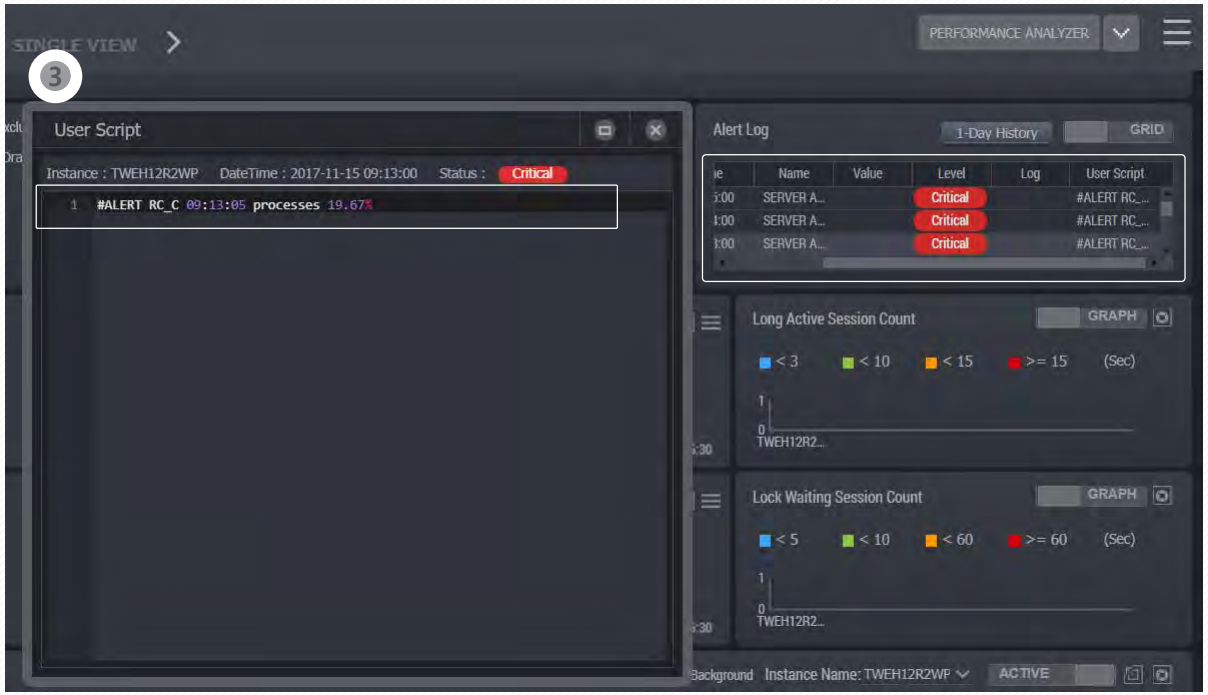
✓ **목표**

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

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

"평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!"

Alert User Script 추가 화면



✓ **시나리오**

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

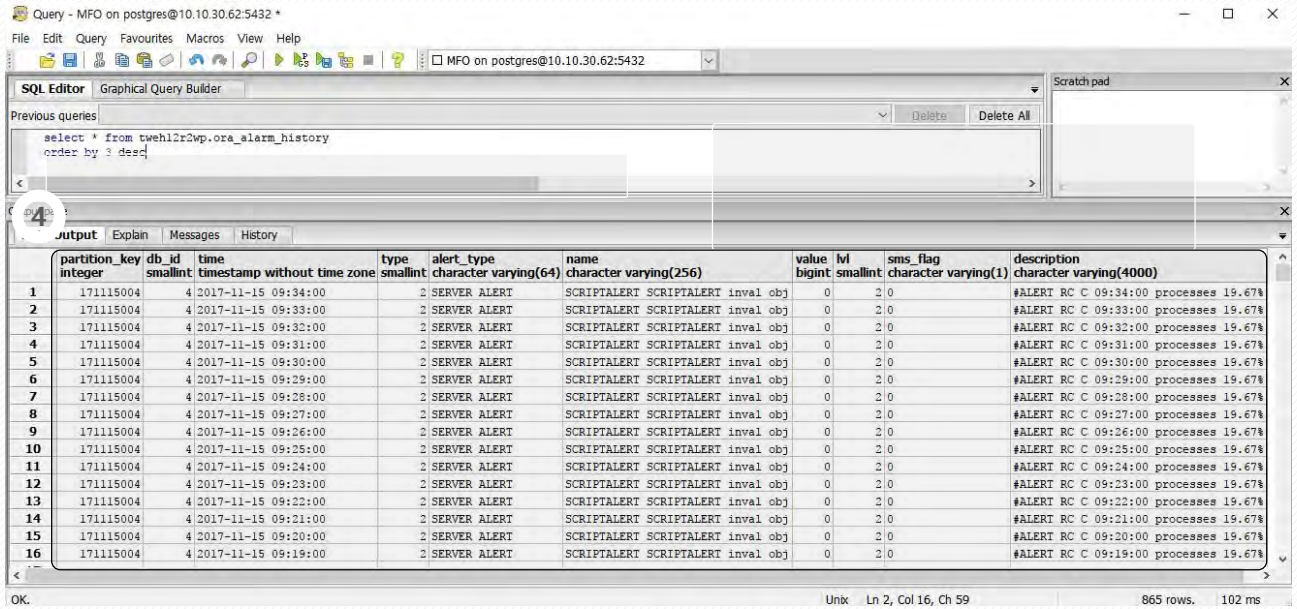
✓ **목표**

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

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

"평소 사용하는 Script를 이용하여 Alert을 등록하고 싶어요!"

Alert User Script 추가 화면



✓ 시나리오

INSTANCE : TWEH12R2WP

결과 확인 방법 : ora_alarm_history Table

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

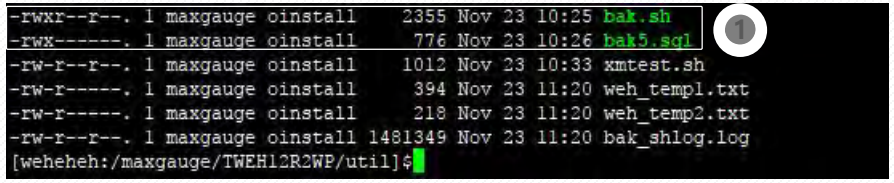
1. Script 파일의 실행권한을 확인합니다.
2. rts.conf 파일에 사용할 script 정보를 등록시킵니다.
3. RTM 화면에서 알람 발생 여부를 확인합니다.
4. Repository > ora_alarm_history Table Data를 확인합니다.

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 User Script 추가 화면



✓ **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 User Script 추가 화면

```
db_id = 4
daemon_port=5083
wr_host=10.10.30.62
wr_port=7001
stat_interval=1
send_sysstat_interval=1
session_list_interval=1
send_session_list_interval=1
send_lock_info_interval=1
top_process_interval=3
xview_limit=1000
sqllog_gathering_count=20
listener=10.10.30.250:1521
listener_check_interval=10
script1=backup_status,60,0,/maxgauge/TWEH12R2WP/util/bak.sh
```

②

✓ **시나리오**

INSTANCE : TWEH12R2WP

결과 확인 방법 : rts.conf 내용 확인

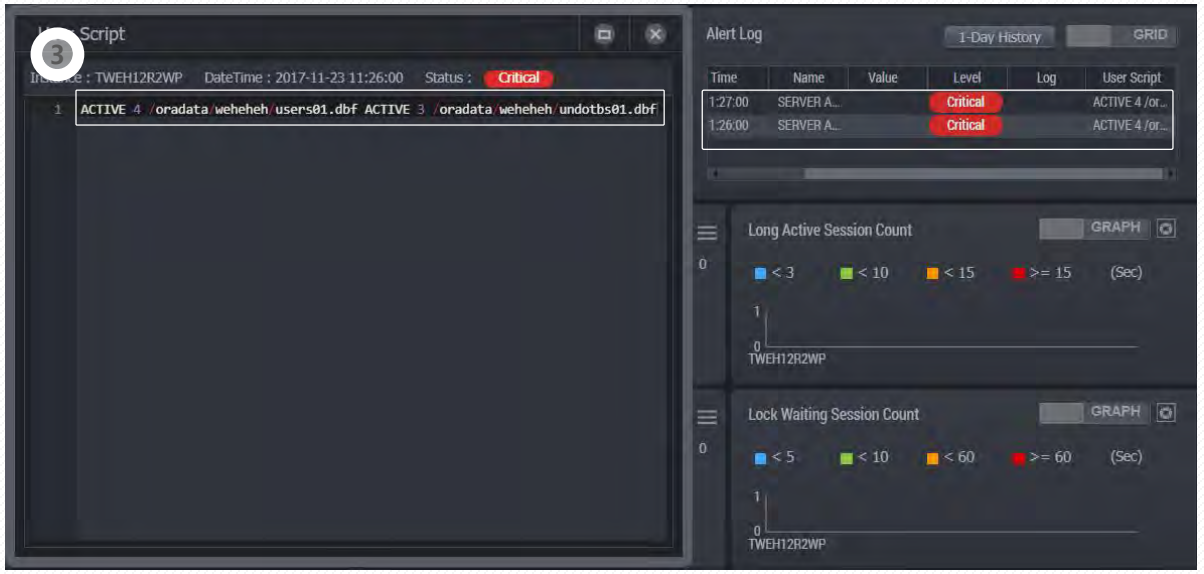
✓ **목표**

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

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

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

» Alert User Script 추가 화면



✓ **시나리오**

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

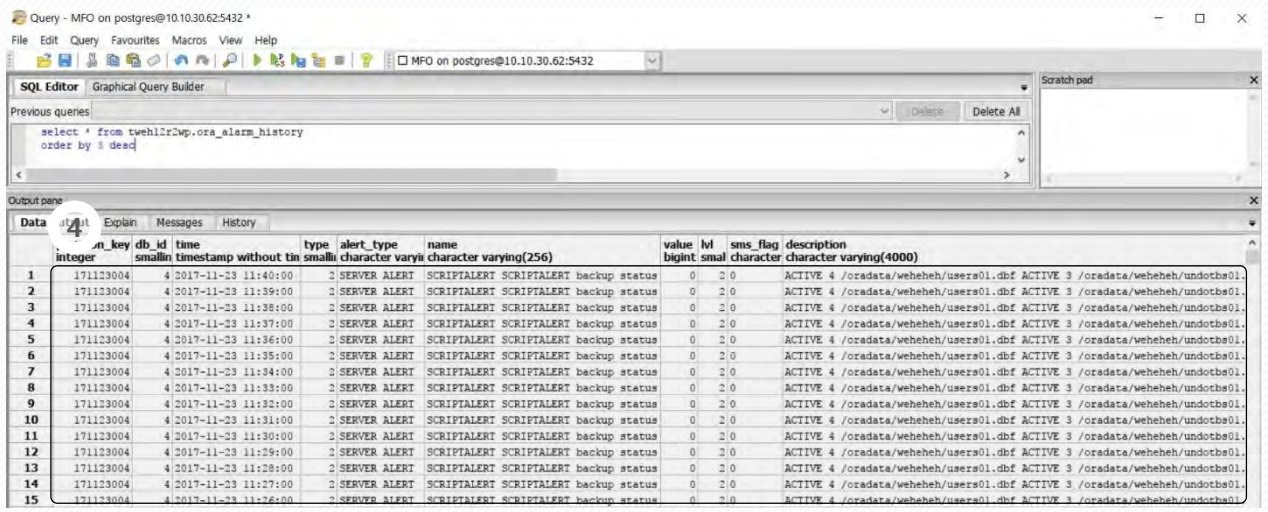
✓ **목표**

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

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)

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

Alert User Script 추가 화면



✓ 시나리오

INSTANCE : TWEH12R2WP

결과 확인 방법 : ora_alarm_history Table

✓ 목표

사용자가 평소 사용하는 script를 이용하여 알람을 등록할 수 있다.

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.

MAXGAUGE Practical Guide

Script Manager [Real Time Monitor]

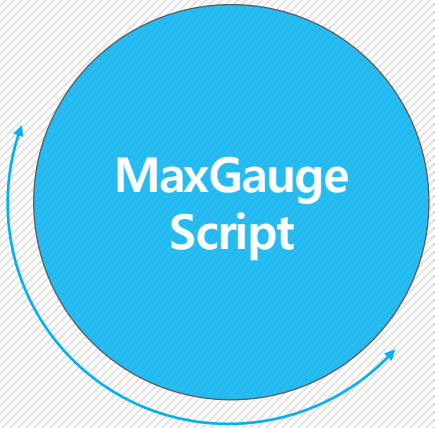
PRODUCT
DOCUMENTATION

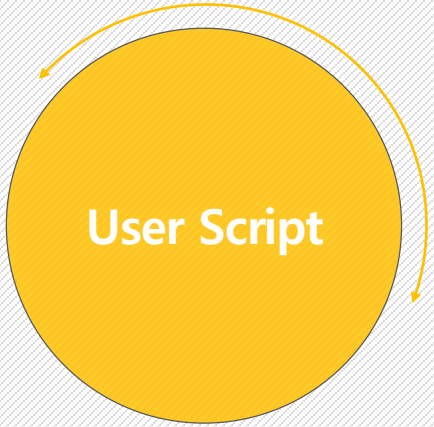
Contents

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

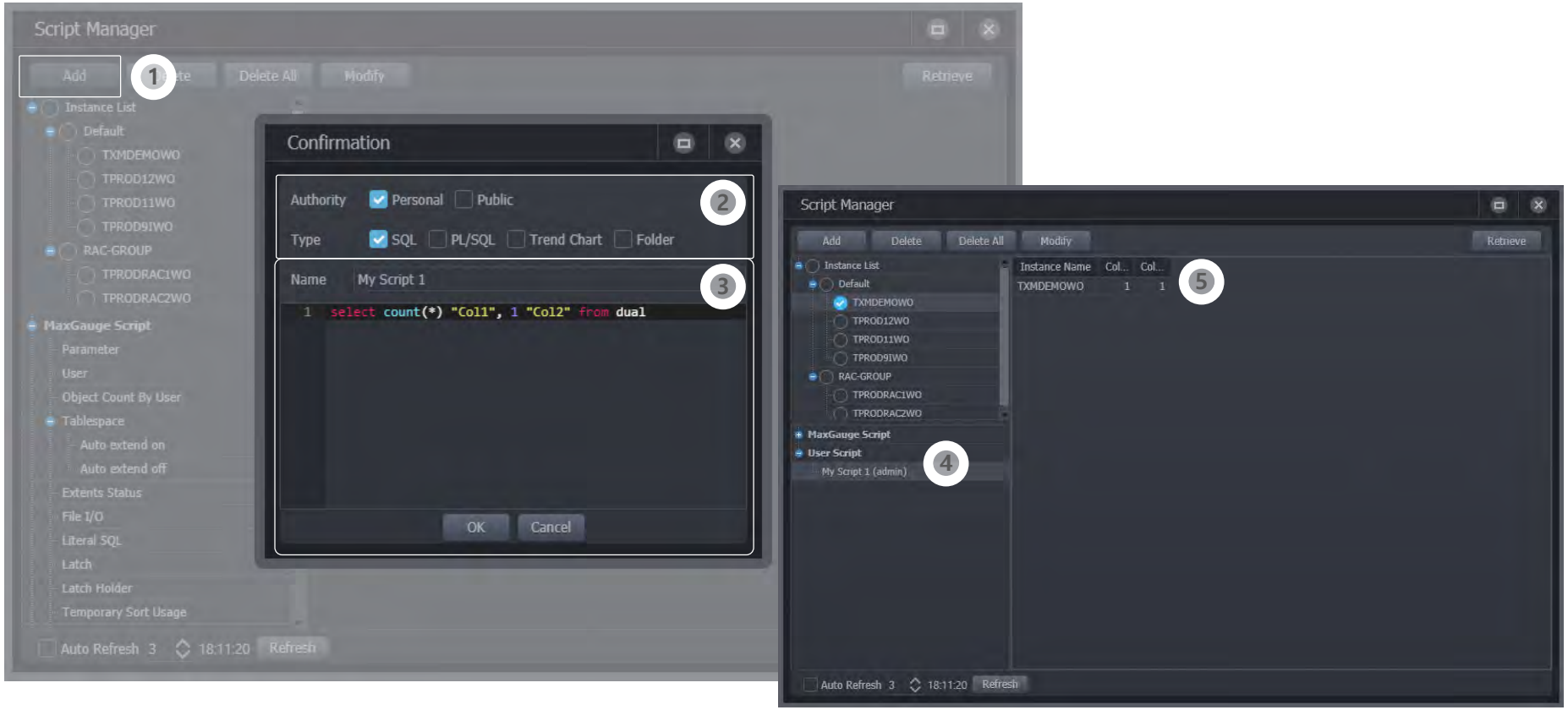
Script Manager 의 사용 방법

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

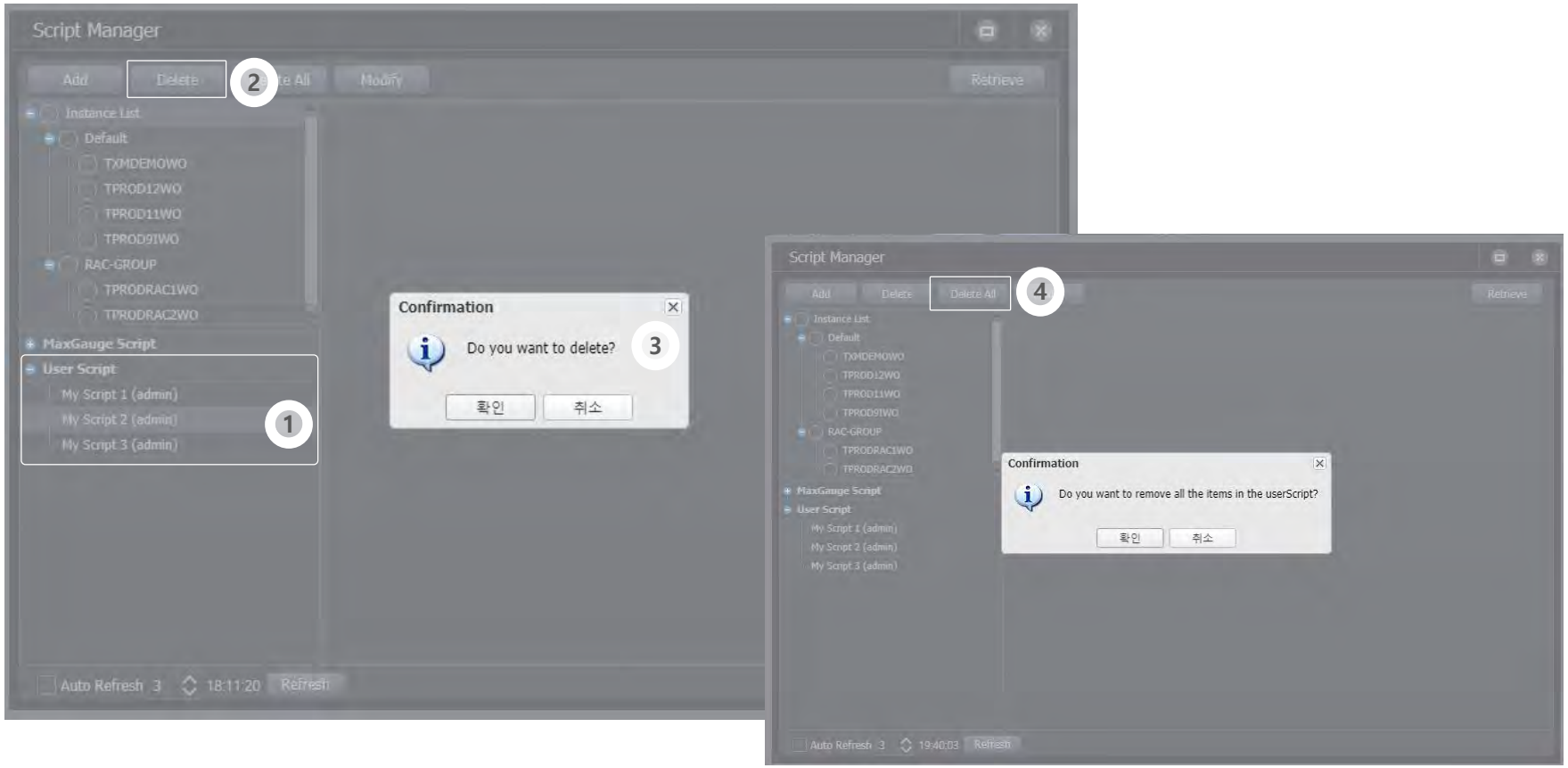
User Script 등록



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 List 에서 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

User Script 삭제



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

User Script 수정

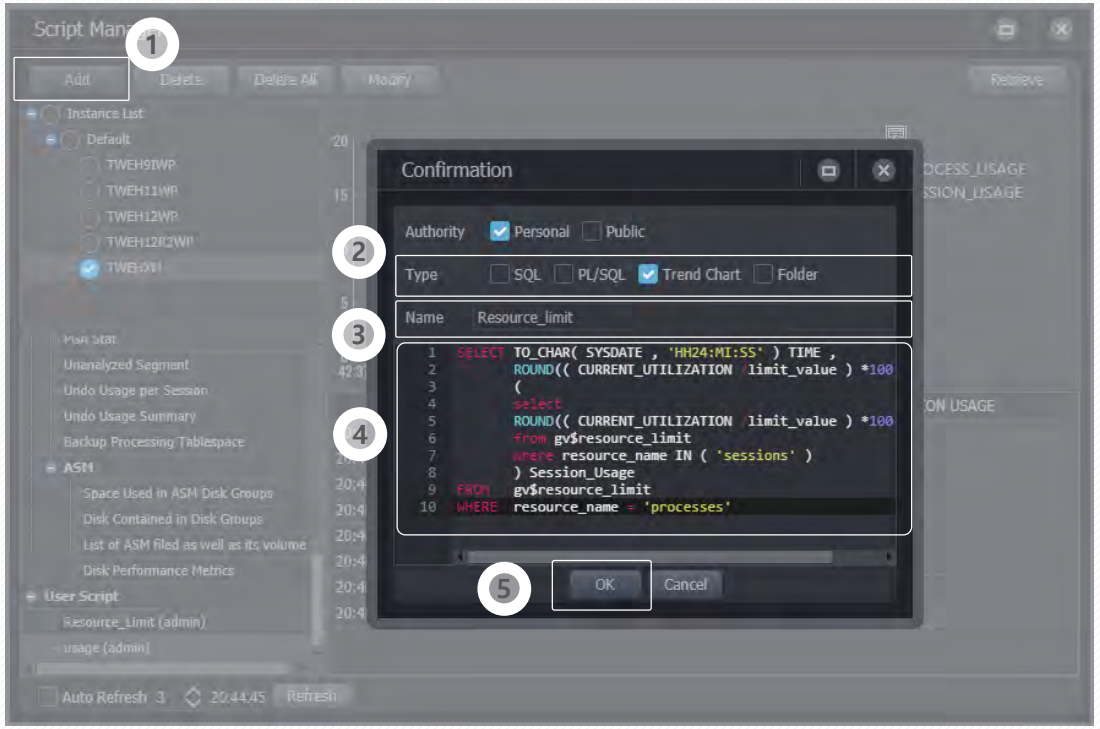
1. 수정할 User Script 를 선택합니다.
2. Modify 버튼를 클릭합니다.
3. 내용 수정 후 OK 버튼을 클릭합니다.

실전 활용 사례 Case 1.

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

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

User Script추가 화면



✓ **시나리오**

INSTANCE : TWEHXM
결과 확인 방법 : Trend Chart (그래프)

✓ **목표**

Process 의 사용량과,
Session 의 사용량 확인 하기

1. Add 버튼을 클릭하여, Script 등록 창을 실행 시킵니다.
2. Type 에서 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.

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

» Trend Chart 유형 결과 화면

✓ **시나리오**

INSTANCE : TWEHXM
결과 확인 방법 : Trend Chart (그래프)

✓ **목표**

Process 의 사용량과,
Session 의 사용량 확인 하기

1 Instance List 영역에서 'TWEHXM' Instance 를 'Check' 합니다.

2 Auto Refresh (Default 3) sec 를 선택 하거나, 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.

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

» SQL유형 결과 화면

✓ **시나리오**

INSTANCE : TWEHXM, TWEH12R2WP
결과 확인 방법 : Trend Chart 유형
: SQL 유형

✓ **목표**

Process 의 사용량과,
Session 의 사용량 확인 하기

1 Instance List 영역에서 'TWEHXM' , 'TWEH12R2WP' Instance 2개를 'Check' 합니다.

2 Auto Refresh (Default 3) sec 를 선택 하거나, Guide Refresh 를 선택 합니다.

3 Instance 별 수행 결과가 순차적으로 출력 됩니다.

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.

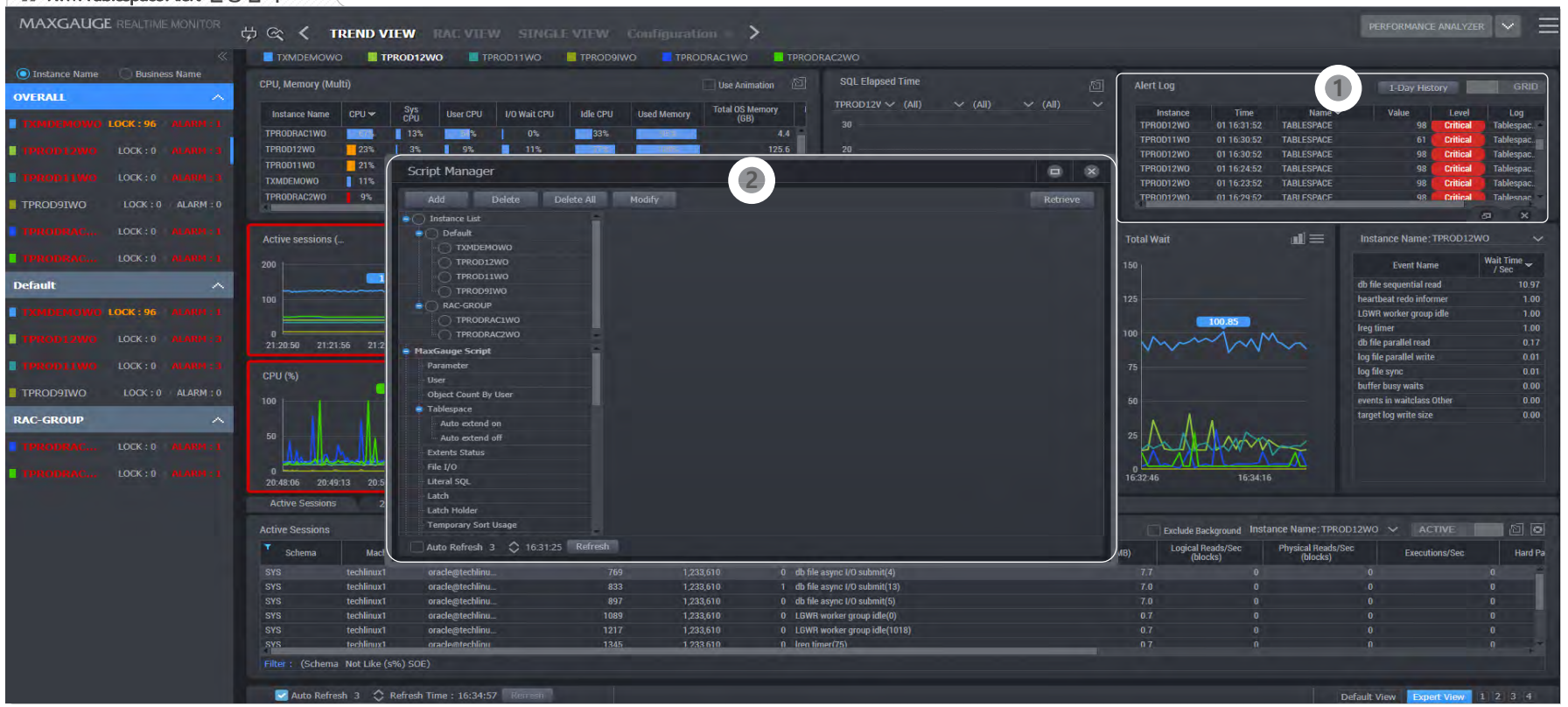
STEP

1. RTM 에서 Tablespace Alert 감지

2. Script Manger Tablespace 정보 확인

- 해당 인스턴스 선택
- MaxGauge Script 선택
- Auto extend on 더블클릭
- Tablespace 사용정보 확인

RTM Tablespace Alert 발생 감지



1 Real- Time Monitor에서 Tablespace Alert이 감지 되었습니다.

2 메뉴 -> Script Manager를 실행합니다.

Tablespace Alert 발생 시 현재 시점의 Tablespace 사용정보를 확인하고 싶습니다.

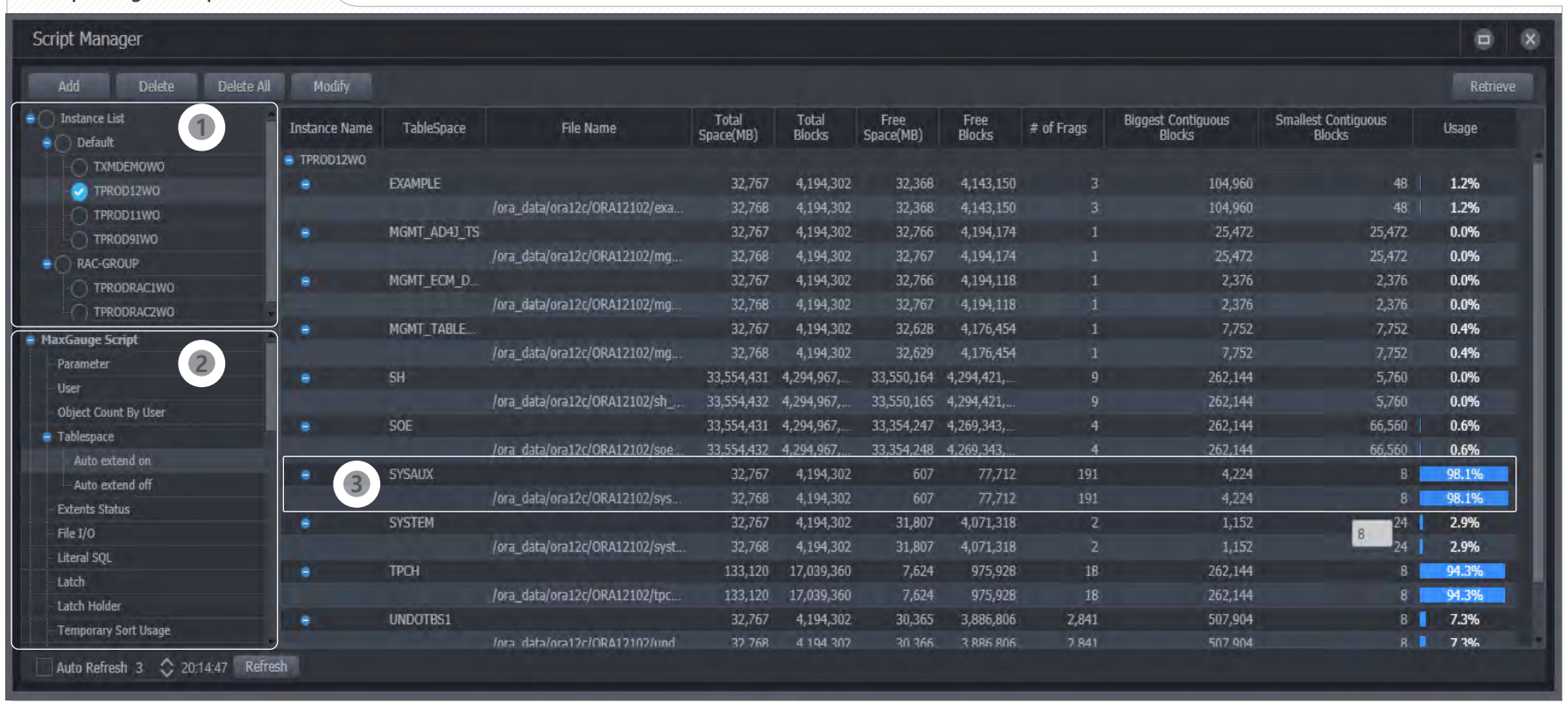
STEP

1. RTM 에서 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



1 대상 Instance 를 선택합니다.

2 MaxGauge Script 중 Tablespace – auto extend on 더블클릭 합니다.

3 Tablespace 정보 확인합니다.

Thank You