SQL 실행계획 가이드

태그

SQL Trace 기능 이용하여 tbPROF로 수행결과 보기
SQL 실행 정보를 profiling 하기 위한 SQL trace 파일은 정적 또는 동적으로 SQL_TRACE 파라미터를 적용한다.
SQL 실행과정을 trace로 남기게 하는 기능
SQL Trace를 이용 Plan상에서 튜닝 대상을 찾귀 쉬움
SQL Trace 내용을 파일에 저장하기 때문에 쿼리 처리가 느려질 수 있음

SQL_Trace 파라미터 사용
SQL_TRACE 파라미터 사용
SQL_TRACE는 SQL_trace 정보를 기록할지 여부를 결정한다. SQL trace는
실행정보의 자료 수집을 가능하게 한다. 그러나 이 기능을 사용하면 서버 성능에
영향을 미치기 때문에 실제 서버 운영환경에서 이 기능을 사용하지 않도록 권장한다

Boolean 타입
기본값 N / 속성 Optional, Adjustable, Dynamic, Session
설정 방법 : TIP파일을 설정 한 후 재가동 하거나 ALTER 문으로 변경한다.
문법
TIP 파일
SQL_TRACE = {Y|N}
ALTER 문
ALTER SYSTEM SET SQL_TRACE+{Y|N}
ALTER SESSION SET SQL_TRACE = {Y|N}

```
SELLECT SID, SERIAL#, USERNAME, SQL_TRACE, TO_CHAR(LOGON_TIME,'DD HH24:MI:SS')
LOGON_TIME
FROM V$SESSION
ORDER BY LOGON_TIME;
```

```
SELECT e.employee_id, e.first_name||' ' ||e.last_name emp_name, d.dpartment_id, d.depa
rtment_name
FROM department d
LEET OUTER JOIN employees e ON(e.department_id=d.department_id);
```

생성된 SQL Trace 파일 확인
SQL Trace 작업 후 trace file 은 *.trc라는 확장자로 다음 위치에 존재한다.
생성파일 위치 - $TB_HOME/instance/$TB_SID/log/sqltrace 디렉터리
파일명생성규칙
파일명은 해당 세션의 PID, SID, serial#정보를 이용하여 생성한다. ex)tb_sqltrc_PID_SID_serial#.trc
-v$session 뷰에서 해당 세션의 PID, SID, serial# 정보 조회가 가능하다.

```
SELECT PID, SID, SERIAL#, USERNAME, SQL_TRACE, TO_CHAR(LOGON_TIME, 'DD HH24:MI:SS') LO
GON_TIME
FROM V$SESSION
ORDER BY LOGON_TIME;
```

예를 들면 tb_sqltrc_3181_19_567.trc 파일생성을 확인
cd tibero6/instance/tibero/log/sqltrace
tbPROF 각각의 SQL에 대해 parse, execution, fetch 단계 별 정보를 제공해주며 사용자가 지정하는 필드에 대해 결과를 정렬해서 보여준다. tbPROF 명령어를 통한 파라미터 및 실행방법은 다음과 같다.

```
tbprof
tbprof tracefile outfile [print= ] [sort= ] [aggregate= ]
```

tracefile SQL Trace로 생성된 통계정보를 가진 파일명
outputfile tbPROF를 통해 읽기 가능한 텍스트파일로 생성할 파일명
print 지정된 수의 SQL문에 대해서만 TRACE결과를 PRINT함
sys SYS user가 내부적인 작업을 위해 실행한 SQL문을 출력시 포함 여부
aggregate 같은 SQL에 대한 정보를 합산할지 여부를 지정하는 파라미터
sort 결과 정렬방식을 지정하는 파라미터 SORT_OPTION에 따라 내림차순으로 정렬함
여러개 동시 지원 저장 가능하며 필드는 , 로 구분하고 공백이 있으면 안됨
tbPROF 실행 후 OUTPUTFILE 분석
tbPROF를 실행하여 실행한 SQL정보를 분석한다. outputfile 파일에서, 통계정보, 실행계획등의 정보를 확인한다.

```
tbprof tbprof tb_sqltrc_3181_19_567.trc tb_sqltrc_3181_19_567,out sys=no
```

tbPROF 통계 정보
parse 테이블, 칼럼,참조 오브젝트 권한 체크를 포함하여 SQL문장을 실행계획으로 변환한다.
Execute 실행계획에 따라 데이터 변경을 발생시키는 INSERT ,UPDATE, DELETE를 처리한다. SQL구문 처리 시SELECT된 건수를 나타낸다.
Fetch 쿼리 결과 반환되는 ROW 건수를 나타낸다. fetch는 SELECT 구문 처리 시 수행된ㄷ.
Count SQL문의 파싱된 횟수, 실행된 횟수, FETCH가 수행된 횟수
CPU PARSE, EXECUTE, FETCH가 실제로 사용한 CPU 시간 (seconds단위)
Elapsed 작업의 시작에서 종료 시까지 실제 소요된 총 시간(seconds 단위)
Current 세션에서 작업한 내용을 Commit하지 않아 오로지 자신에게만 유효한 블록 (Dirty Block)을 엑세스한 블록 수
주로 UPDATE, INSERT, DELETE 작업 시 많이 발생
SELECT 문에서는 거의 없으나 아주 적은 양의 경우가 대부분임
Query 메모리 내에서 변경되지 않은 블록을 읽거나 다른 세션에 의해 변경되었으나 Commit되지 않아 복사해 둔 스냅샷 블록을 얻은 블록의 수
ㅇSELECT 문에서는 거의가 여기에 해당하며 UPDATE , DELETE< INSERT시에는 소량만 발생됨

Disk 디스크에서 읽혀진 데이터 블록의 수
Rows SQL문을 수행 결과에 의해 최종적으로 엑세스된 ROW의 수 / 서브쿼리에 의해서 추출된 ROW는 제외됨
et 해당 노드에서 수행된 시간(usec)

```
cr 해당 노드에서 읽은 cr block 개수
cu 해당 노드에서 읽은 current block개수
co optimizer에서 계산된 노드의 cost
ro optimizer에서 예측한 노드의 row 개수
et 해당 노드에서 수행된 시간(usec)
```

```
SET_SQL_TRACE_IN_SESSION 프로시저 사용
SQL_TRACE 파일 생성 여부를 설정
특정 세션의 SQL추적 로그 작성을 시작하거나 중지할 수 있다. 세션의 식별자 및 시리얼번호는 V$SESSION뷰를 통
해 조회할 수 있다.
SQL 추적 로그는 $TB_HOME/instance/$TB_SID/log/sqltrace 경로에 생성된다.
DBMS_SYSTEM.SET_SQL_TRACE_IN_SESSION

  SID      NUMBER    세션의 식별자 번호
SERIAL#    NUMBER    세션의 시리얼 번호
  SQL_TRACE   BOOLEAN  SQL추적 로그를 작성하려면 TRUS, 중지하려면 FALSE를 입력한다

실행문법
EXEC SYS.DBMS_SYSTEMSET_SQL_TRACE_IN_sESSION(SID, SERIAL#, SQL_TRACE);
BEGIN SYS.DBMS_SYSTEM.SET_SQL_TRACE_IN_SESSION(SID, SERIAL#, SQL_TRACE);
END
```

```
tbsql sys/tibero
SET_SQL_TRACE_IN_SESSION 정보 확인

DESC DBMS_SYSTEM
V$SESSION에서 파라미터 값 조회
SELECT SID, SERIAL#, USERNAME, SQL_TRACEM TO_CHAR(LOGON_TIME,'DD HH24:MI:SS')
LOGON_TIME
FROM V$SESSION
ORDER BY LOGON_TIME.

SQL_TRACE=Y로 변경
EXEC DBMS_SYSTEM.SET_SQL_TRACE_IN_SESSION(19 , 567, TRUE);

SELECT e.employee_id, e.first_name || ' ' e.last_name emp_name, d.department_id, d.dep
artment_name
FROM department d
LEET OUTER JOIN employees e
ON(e.department_id=d.department_id);

SELECT PID, SID, SERIAL$, USERNAME, SQL_TRACE, TO_CHAR(LOGON_TIME, 'DD HH24:MI:SS') LO
GON_TIME
FROM V$SESSION
ORDER BY LOGON_TIME;

cd tibero6/instance/tibero/log/sqltrace
ls

>>outputfile todtjd
tbprof tbprof tb_sqltrc_3181_19_567.trc tb_sqltrc_3181_19_567.out sys=no
```

```
cat tb_Sqltrc_3181_19_567.out
```

```
TBSQL에서 AUTOTRACE 기능 이용
SET AUTOT[RACE] {OFF|ON|TRACE[ONLY]| [EXP[LAIN]] [STAT[STICS]] [PLANS[TAT]]

SET AUTOTRACE OFF : AUTOTRACE를 수행하지 않는다. (기본값)
SET AUTOTRACE ON 실행 계획, 실행 결과, 통계 자료 모두 출력
SET AUTOTRACE TRACEONLY 실행 계획, 통계 자료를 출력하고 속도 빠름
SET AUTOTRACE ON EXPLAIN 실행 결과, 실행 계획을 출력
SET AUTOTRACE ON STATISTICS 실행 결과, 통계 자료를 출력
SET AUTOTRACE ON PLANSTAT  실행 결과, 노드 별 쿼리 수행정보를 출력 (수행 시간, 처리 ROW 개수, 수행 횟수)

AUTOTRACE 사용시 {OFF | ON|TRACE[ONLY]}은 3개 중 하나를 꼭 사용한다.
[EXP[LAIN] [STAT[STICS] [PLAN[TAT]]은 실행계획 / 실행통계/ 노드 별 쿼리 수행정보 (수행 시간, 처리 ROW개수, 수행 횟수)를 보여줄지 정하는 옵션으로 덧붙여 써도 되고, 골라 써도 되고, 안 써도 된다.
AUTOTTRACE는 옵션에 따라 쿼리 수행 결과, 실행 계획, 실행통계를 선택적으로 출력할 수 있다.
쿼리 수행결과와 실행통계는 수행 후 알 수 있는 결과이고 실행 계획은 예측 결과이다.

통계자료
db block gets: 현재의 블록이 요구된 횟수
consistent gets: cosistent mode에서 읽은 논리 블록 수를 누적한 시스템 통계 정보
physical reads: 디스크로부터 읽은 데이터 블록의 총 개수
redo size: redo log가 만들어진 크기(size)
sorts(disk): disk에서 일어난 sort tn
sorts(memory): memory에서 일어난 sort수
rows processed:연산을 하는 동안 처리한 row 수
```

```
권한 -DBA권한 또는 PLUSTACE 권한이 있어야 사용 가능하다

SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
```

```
tbsql sys/tibero
SET AUTOTRACE ON;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
SET AUTOTRACE TRACEONLY;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
SET AUTOTRACE ON EXPLAIN;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEOT_CD;
SET AUTOTRACE ON STATISTICS;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
SET AUTOTRACE ON PLANSTAT;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
SET AUTOTRACE OFF;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
```

```
V$SQL_PLAN 이용 : SQL문을 실행하는 물리적인 계획에 대한 정보를 표시한다.
HASH_VALUE   NUMBER 물리적 계획의 hash 값
SQL_ID     NUMBER SQL식별자
OPERATION   VARCHAR(128)  operaion job의 이름
OBJECT#  NUMBER Identifier of object accessed by the job
OBJECT_OWNER  VARCHAR(128) object를 가진 사용자의 이름
OBJECT_NAME  VARCHAR(128) object name
OBJECT_TYPE VARACHAR(20)  object type
ID   NUMBER  물리적 계획의 각 작업에 주어진 번호
PARENT_ID  NUMBER  물리 계획의 입력으로 현재 작업의 출력을 얻을 수 있는 작업의 ID
DEPTH  NUMBER  물리적 계획의 tree level
POSITION  NUMBER 같은 PARENT_ID를 가진 모든 작업의 위치
SEARCH_COLUMNS  NUMBER 인덱스 검색에 사용되는 KEY의 수
COST  NUMBER 쿼리 최적화에 예상되는 작업 비용
CPU_COST  NUMBER  최적화에 추정되는 작업의 CPU 비용
IO_COST  NUMBER   최적화에 의해 예상되는 I/O 비용
CARDINALITY  NUMBER 쿼리 최적화에 의해 예상되는 출력 결과의 수
PSTART VARCHAR(38) 파티션 테이블에서 엑세스를 하기 위한 기동 파티션
PEND VARCHAR(38)  분할된 테이블에서 엑세스를 위한 최종 파티션
OTHERS  VARCHAR(4000) 사용자가 유용하게 사용할 수 있는 실행단계에 관한 기타 정보
ACCESS_PREDICATES  VARCHAR(4000) index 접근이나 join을 위한 predicate 정보
FILTER_PREDICATES   VARCHAR(4000) filter처리를 위한 predicate 정보
```

```
SELECT JOB_ID, round(AVG(SALARY)) AVG FROM EMPLOYEES
GROUP BY JOB_ID order by 2 desc;

select sql_id, sql_text
from v$sqltext
where sql_text like '%FROM EMPLOYEES';

select sql_id, aggr_concat(sql_text, '' order by piece)as sql
from $sqltext
where sql_id=521 group by sql_id;
```

```
SQL_TRACE_DEST 파라미터
SQL_TRACE_DEST는 SQL trace파일이 저장 될 디렉토리를 설정한다. 디렉터리를 지정 할 때 절대 경로로 해야
 한다. 일반 디렉터리는 $TB_HOME/instance/$TB_SID/log/sqltrace이고, 이 파라미터 설정을 해서 다른 디
렉토리고 변경 가능하다.
타입 String
기본 값 ""
속성 Optional, Adjustable, Dynamic, System
설정방법 TIP파일을 설정한 후 재기동하거나 ALTER문으로 변경한다.
문법 TIP파일 SQL_TRACE_DEST=<디렉터리>

ALTER SYSTEM SET SQL_TRACE_DEST=<디렉터리>
```

```
set autotrace on explain;
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
@plutrace.sql
SELECT AVG(SALARY) AVG FROM employee GROUP BY DEPT_CD;
```

```
SYSTEM FILES
tibero 가 normal mode상태까지 기동되기 위한 system 관련 파일들을 말한다. tibero가 설치가 완료되면 co
ntrolfile, system, undo, syssub, usr, temp, redolog, passwd파일들이 기본적으로 구성된다. 정상
적으로 티베로가 기동된 상태에서 SQL명령어로 위치 및 정보를 확인할 수 있다.

Controfile : 데이터베이스 자체의 메타데이터를 보관하고 있는 바이너리 파일
select * from v$controlfile;
datafile
SYSTEM : DATA Dictionary(Meta DATA)정보가 들어있는 테이블 스페이스
UNDO : commit/rollboack 을 위한 테이블스페이스
SYSSUB: TPR용 테이블 스페이스
USR : 일반 유저테이블스페이스(기본생성)
select * from dba_data_files where tablespace_name in ('SYSTEM','UNDO','SYSSUB','USR')
order by tablespace_name;
```

```
Tempfiles : 메모리 가용 공간이 부족할때 swap 용도로 사용하는 테이블스페이스
select * from dba_temp_files;

Redo Log 데이터베이스에서 발생하는 모든 변경내용을 저장하는 Log파일
select * from v$log;
select * from v$logfile;

passwd nomount와 mount모드로 부트시 인증에 사용
DB_CREATE_FILE_DEST에 위치

#Controlfile
set lines 200
col name for a30
select * from v$controlfile;

#SYSTEM Datafiles
set lines 200
col file_name for a32
col tablespace name for a10
col maxbytes for 99999999
select * from dba_data_files where tablespace_name in ('SYSTEM','UNDO','SYSSUB','USR')
order by tablespace_name;

#SYSTEM Tempfile
set lines 200
col file_name for a32
col tablespace_name for a10
col maxbytes for 999999
select * from dba_temp_Files;
```

```
#RedoLog files
select * from v$log;
select * from v$logfile;

#passwd
cd home/tb6/tbdata ls -al
```

```
INSTANCE LOGS
TIBERO가 설치되면 $TB_HOME/instance/$TB_SID/log 디렉토리에 로그들이 생성된다. 파라미터(tip)를 통해서 위치를 바꿀 수 있다.
slog 디버깅을 위한 팡ㄹ이다. 서버가 하는 중요한 일이 기록되는 파일이며 서버 성능이 저하되는 원인을 찾거나 Tibero 자체의 버그를 해결하는데 사용 될 수 있다.
dlog 시스템 로그 파일에 기록되는 정보보다 좀 더 중요한 정보가 기록되는 파일이며 서버 기동 및 종류, DDL 문장의 수행 등이 기록되는 파일이다.
llog 스레드별로 설정된 이벤트에 대한 시스템 로그가 기록되는 파일이며,internal 로그를 보려면 tbv를 이용해야 한다.
lsnr Listener의 디버깅을 위한 파일이다. 리스너에서 일어난 중요한 일이 기록되는 파일이며 리스너의 버그를 해결하는데 사용될 수 있다.

cd $TB_HOME/instance/$TB_SID/log
```

```
Tibero Client Connection
tbsql : Tibero에서 제공하는 SQL문장을 처리하는 대화형 유틸리티이다.
tbdsn.tbr alias usage
tbdsn.tbr : client 에서 Tibero서버에 접근하기 위한 접속 정보를 설정하는 환경파일이다.
Parameter
HOST    : 서버의 IP주소
PORT   : 서버의 포트번호
DB_NAME  : 데이터베이스 이름
```

```
Tibero Process Startup & Shutdown
티베로 기동과 종료 절차이다. 티베로는 Single /TAC/TSC로 구성이 가능하며 각각의 구성방식에 따라 기동/종료 절차가 다르다. 각자의 구성에 맞는 기동방법을 확인해야 한다.

Single : 하나의 Instance로 구성된DB이다.

1) startup
tbboot
2) shutdown
tbdown immediate

TAC : 하나의 Database에 복수의 Instance로 구성된 DB이다. TAS(Tibero Active Storage)를 쓰는 경우와 아닌 경우로 나뉘어진다.

TAC - without TAS
1) Startup
Clust Manager : tbcm -b
```

```
Tibero : tbboot


root > tbcm -b
root > su -tibero
home/tibero > tbboot

root > tbcm -b
root > su - tibero
home/tibero > tbboot

2) shutdown
cluster manaber : tbcm -d
tibero : tbdown immediate

node1
tbdown immediate
exit
tbcm -d

node2
tbdown immediate
exit
tbcm -d
```

```
TAC - using TAS
1)Startup
1. CM : Common  * tbcm -b
2. TAS :
cmrctl 사용 * cmrctl start as -name <AS1_NAME>
Tibero command 사용 * export TB_SID=<AS1_NAME>
tbboot
Remote other nodes
* cmrctl start as --name <AS2_NAME> --remote
<CM2_SID>@<Cluster_Name>
3. TAC
cmrctl 사용 * cmrctl start db --name <DB1_NAME>
Tibero command 사용 * export TB_SID=<DB1_NAME> * tbboot
Remote other nodes cmrctl start db --name <DB2_Name> --remote
<CM2_SID>@<Cluster_Name>
```