

## Table of Contents

- [Altibase 7.1.0.6.1 Patch Notes](#)
  - [New Features](#)
    - [BUG-49279 QUEUE 테이블에 대한 객체 권한을 추출하는 기능을 추가합니다.](#)
  - [Fixed Bugs](#)
    - [BUG-47830 LOB 데이터 타입 바인드 시 SQLBindParameter 함수에서 매개변수 type 에 입/출력 변수\(SQL PARAM INPUT OUTPUT\)를 사용한 경우 Altibase 서버가 비정상 종료합니다.](#)
    - [BUG-48603 재사용된 TABLE OID가 이중화 값에 포함되어 있는 경우 The row already exists in a unique index. 에러가 발생하며 이중화 시작이 실패합니다.](#)
    - [BUG-49168 "PROJ-2749 CTE\(Common Table Expression\) Validate 단계 최적화" 프로젝트를 반영합니다.](#)
    - [BUG-49250 Direct-Execute 방식의 statement를 반복 수행 중 prepare 에러가 발생한 경우 이전에 수행한 statement가 다시 실행될 수 있습니다.](#)
    - [BUG-49263 다수의 파티션이 이중화에 포함된 파티션드 테이블에 연속적으로 TRUNCATE 수행 시 ERR-11041 : A deadlock situation has been detected. 에러가 발생할 수 있습니다.](#)
    - [BUG-49281 분석 함수 OVER절에 ORDER BY를 사용한 컬럼을 alias 사용하고 JOIN 의 ON절에 AND, OR 조건과 함께 사용한 경우 Altibase 서버가 비정상 종료하는 현상을 수정합니다.](#)
    - [BUG-49283 UNIQUE INSERT 수행 시 LOCK WAIT 상황에서 V\\$SESSION WAIT에 no wait event가 발생합니다. wait event가 수집되지 않습니다.](#)
  - [Changes](#)
    - [Version Info](#)
    - [호환성](#)
    - [프로퍼티](#)
    - [성능 분](#)

# Altibase 7.1.0.6.1 Patch Notes

## New Features

**BUG-49279 QUEUE 테이블에 대한 객체 권한을 추출하는 기능을 추가합니다.**

- **module** : ux-aexport
- **Category** : Functional Error
- **재현 빈도** : Always
- **설명** : QUEUE 테이블에 대한 객체 권한을 추출하는 기능을 추가합니다.
- **재현 방법**
  - 재현 절차
  - 수행 결과
  - 예상 결과
- **Workaround**
- **변경사항**
  - Performance view
  - Property
  - Compile Option
  - Error Code

## Fixed Bugs

**BUG-47830 LOB 데이터 타입 바인드 시 SQLBindParameter 함수에서 매개변수 type 에 입/출력 변수(SQL\_PARAM\_INPUT\_OUTPUT)를 사용한 경우 Altibase 서버가 비정상 종료합니다.**

- **module** : qp-dml-execute
- **Category** : Fatal
- **재현 빈도** : Always
- **설명** : LOB 데이터 타입 바인드 시 SQLBindParameter 함수에서 매개변수 type 에 입/출력 변수 (SQL\_PARAM\_INPUT\_OUTPUT)를 사용한 경우 Altibase 서버가 비정상 종료하는 현상을 수정합니다.
- **재현 방법**
  - 재현 절차

```
CREATE TABLE bug47830 (i1 integer, i2 clob );
INSERT INTO bug47830 VALUES(1, 'abc');
/* prepares an SQL string for execution */
rc = SQLPrepare(stmt, (SQLCHAR *)"INSERT INTO bug47830 VALUES(?, ?)",
SQL_NTS);
if (!SQL_SUCCEEDED(rc))
{
    PRINT_DIAGNOSTIC(SQL_HANDLE_STMT, stmt, "SQLPrepare");
    goto EXIT_STMT;
```

```

}
/* binds a buffer to a parameter marker in an SQL statement */
rc = SQLBindParameter(stmt, 1, SQL_PARAM_INPUT,
                      SQL_C_SLONG, SQL_INTEGER,
                      0, 0,
                      &i1,
                      0,
                      NULL);

if (!SQL_SUCCEEDED(rc))
{
    PRINT_DIAGNOSTIC(SQL_HANDLE_STMT, stmt, "SQLBindParameter");
    goto EXIT_STMT;
}
rc = SQLBindParameter(stmt,
                      2,
                      SQL_PARAM_INPUT_OUTPUT,
                      SQL_C_CLOB_LOCATOR, SQL_CLOB,
                      0,
                      0,
                      &c1_loc, 0, &c1_ind);

if (!SQL_SUCCEEDED(rc))
...종락...

```

## ○ 수행 결과

### 1. Altibase 클라이언트 측

어플리케이션에서 아래와 같은 에러 발생합니다.

#### Diagnostic Record 1

```

SQLSTATE      : 08S01
Message text   : Communication link failure. Connection closed
Message len    : 45
Native error   : 0x51043

```

### 2. Altibase 서버 측

Altibase 트레이스 로그 altibase\_error.log 에 아래와 같은 콜스택을 출력합니다.

BEGIN-STACK [CRASH] =====

```

Caller[0] 0000000000418CE1    => mmmSignalHandler
Caller[1] 0000003620C0F500    => not found
Caller[2] 000000000042EB88    => mmtCopyBindParamDataCallback(qciBindParam*,
unsigned char*, unsigned int*)
Caller[3] 00000000004D2054    => qci::setBindParamData(qciStatement*, unsigned
short, unsigned char*, IDE_RC (*)(qciBindParam*, unsigned char*, unsigned int*))
Caller[4] 0000000000430A78    =>
mmtServiceThread::paramDataInProtocol(cmiProtocolContext*, cmpProtocol*, void*,
void*)
Caller[5] 0000000000F94017    => cmiRecv(cmiProtocolContext*, void*,
PDL_Time_value*, void*)
Caller[6] 000000000042AA7F    => mmtServiceThread::executeTask_READY(mmcTask*,
mmcTaskState*)
Caller[7] 000000000042C5F5    => mmtServiceThread::executeTask()
Caller[8] 000000000042CBC2    => mmtServiceThread::multiplexingAsShared()
Caller[9] 000000000042D257    => mmtServiceThread::run()
Caller[10] 0000000000FC9A72    => idtContainer::staticRunner(void*)
Caller[11] 0000003620C07851    => not found
Caller[12] 00000036204E767D    => not found

```

END-STACK =====

## ○ 예상 결과

```
Diagnostic Record 1
SQLSTATE      : HY090
Message text   : Invalid binding to host variables
Message len    : 33
Native error   : 0x31126
Diagnostic Record 2
SQLSTATE      : HY090
Message text   : Invalid binding to host variables
Message len    : 33
Native error   : 0x31126
```

## • Workaround

### • 변경사항

- Performance view
- Property
- Compile Option
- Error Code

**BUG-48603 재사용된 TABLE OID가 이중화 갭에 포함되어 있는 경우 The row already exists in a unique index. 에러가 발생하며 이중화 시작이 실패합니다.**

- **module** : rp
- **Category** : Functional Error
- **재현 빈도** : Frequence
- **설명** :

아래 조건 만족 시 The row already exists in a unique index. 에러가 발생하며 이중화 시작이 실패하는 현상을 수정합니다.

1. 이중화 중지 또는 이중화 갭이 있는 상태
2. 이중화 대상 테이블(A)에 DDL 수행
3. Altibase 서버 재구동
4. 이중화 대상 테이블(B)에 DDL 수행
5. 이중화 객체에서 테이블을 DROP TABLE 후 ADD TABLE 수행
6. 이중화 시작

Altibase 서버 재구동 후 재사용된 TABLE OID가 이중화 갭에 포함되어 있는 경우, 이중화 메타 테이블(SYS\_REPL\_OLD\_ITEMS\_)의 정보가 충돌하며 발생하는 현상입니다. 이 현상 발생 시 altibase\_rp.log에 아래와 같은 로그가 출력됩니다.

ERR-11058(errno=0) The row already exists in a unique index.

ERR-610f8(errno=0) [Sender] Failed to add a XLog [TID:-1603452059, SN:133280338892, Type:2, Log Type:1, Change Log Type:0]

ERR-61014(errno=0) [Sender] Failed to make XLOG in a log file at SN[133280338892]

이 버그에서 메타 테이블(SYS\_REPL\_TABLE\_OID\_IN\_USE\_)이 추가되었습니다.

- 참고: [GeneralReference Manual](#)

이로 인해 메타 버전이 변경되었습니다. 패치 후 롤백이 필요한 경우 [메타 다운그레이드](#)를 수행해야 합니다.

## • 재현 방법

- 재현 절차

1. 이중화 중지 또는 이중화 갭이 있는 상태
2. 이중화 대상 테이블(A)에 DDL 수행
3. Altibase 서버 재구동
4. 이중화 대상 테이블(B)에 DDL 수행
5. 이중화 객체에서 테이블을 DROP TABLE 후 ADD TABLE 수행
6. 이중화 시작

- 수행 결과

이중화 시작이 실패하며 \$ALTIBASE\_HOME/trc/altibase\_rp.log 에 아래 형태의 로그가 출력됩니다.

```
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
[SenderXLog] Table Meta Log [TableOID:3096456->3098776, User Name:SYS, Table
Name:T1, Partition Name:]
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
ERR-11058(errno=0) The row already exists in a unique index.
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
ERR-610f8(errno=0) [Sender] Failed to add a XLog [TID:23297, SN:31532356, Type:2,
Log Type:1, Change Log Type:0]
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
ERR-61014(errno=0) [Sender] Failed to make XLOG in a log file at SN[31532356]
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
SN=<31532356>, MAGIC: 3320, TID: 23297, BE: N, REP: Y, ISVP: N, ISVP_DEPTH: 0,
PLSN=<0, 480, 10071066>, LT: < 38 >, SZ: 49
[2021/02/20 22:37:28] [Thread-140200585975552] [Level-0]
ERR-61013(errno=0) [Sender] Replication failed
```

- 예상 결과

이중화 정상 구동

## • Workaround

이중화 갭 해소 후 이중화 대상 테이블에 DDL 수행

## • 변경사항

- Performance view
- Property
- Compile Option
- Error Code

## BUG-49168 "PROJ-2749 CTE(Common Table Expression) Validate 단계 최적화" 프로젝트를 반영합니다.

- **module** : qp-dml-pvo
- **Category** : Enhancement
- **재현 빈도** : Unknown
- **설명** :

이 버그는 WITH 절을 반복적으로 사용한 긴 문장의 SQL 수행 시 ERR-3111D : There are too many DML statements in the stored procedure, or the SQL query is too long. 에러가 발생하는 현상을 개선한 버그입니다.

- WITH절로 생성한 뷰를 n번 사용 시 내부적으로 statement가 n+1개 생성되는 문제를 개선하기 위해 1개 statement로 압축하는 COMPACT WITH 기법을 적용합니다.
  - SQL 처리 시 내부적으로 사용하는 tuple 수가 줄어듭니다.
  - prepare 시간이 줄어 SQL 전체 수행 시간이 단축됩니다.
  - 비용 기반이 아니기 때문에 비효율적인 실행 계획이 생성될 가능성이 있습니다.
  - 실행 계획에서 COMPACT WITH 기법 사용 여부를 확인하는 방법입니다.

```
ALTER SESSION SET EXPLAIN PLAN = ON;
ALTER SESSION SET TRCLOG_DETAIL_INFORMATION = 1;
```

VIEW 또는 VIEW-SCAN 노드에서 뷰 이름 앞에 \* 표시는 COMPACT WITH 기법이 적용된 것을 의미합니다. 단, 뷰 머징(View Merging)이 발생한 경우는 확인할 수 없습니다.

```
isQL> with query1 as (select /*+ COMPACT_MATERIALIZE */ i1, i2 from t1)
select /*+ no_merge(query1) */ i1, i2 from query1 where i1 > 1;
i1      i2
-----
No rows selected.

-----
PROJECT ( COLUMN_COUNT: 2, TUPLE_SIZE: 8, COST: 161.40 )
VIEW ( * QUERY1, ACCESS: 0, COST: 160.52 )
PROJECT ( COLUMN_COUNT: 2, TUPLE_SIZE: 8, COST: 121.60 )
SCAN ( TABLE: SYS.T1, FULL SCAN, ACCESS: 0, COST: 120.72 )
-----
```

- WITH 절에 ORDER BY 가 사용된 경우 OBYE(Order BY Elimination, 불필요한 ORDER BY를 제거하는 기법)를 적용합니다.
- VIEW 최적화 기능 설정 시 (\_\_OPTIMIZER\_VIEW\_TARGET\_ENABLE = 1) 특정 조건에서 발생하는 결과 오류를 수정합니다.
  - BUG-48045, BUG-48183에서 추가한 VIEW 최적화 기능의 제약 조건을 제거합니다.
- COMPACT WITH 기법과 VIEW 최적화 적용 여부를 결정하는 히든 프로퍼티 \_\_OPTIMIZER\_WITH\_VIEW 추가합니다.
- COMPACT\_MATERIALIZE 힌트를 추가합니다.
  - "COMPACT WITH : WITH(Recursive WITH 제외)뷰가 한 개의 statement를 공유" 기법 적용 여부를 설정합니다.

- 사용 위치
  - WITH 구문 내부
  - WITH 구문 내부에 SET operator( UNION / UNION ALL ..) 을 사용한 경우 제일 처음 SELECT 에 사용
- 힌트 사용 예제
  - 힌트 사용 가능

```
with query1 as (select /*+ COMPACT_MATERIALIZE */ i1, i2 from t1)
select i1, i2 from query1
union all
select i1, i2 from query1;
```

- with 뷰가 아닌 곳에 힌트 사용 : 힌트 무시

```
(1) with 뷰 밖에
with query1 as (select i1, i2 from t1)
select /*+ COMPACT_MATERIALIZE */ i1, i2 from query1
union all
select i1, i2 from query1;

(2) inline-view 안에
select * from ( select /*+ COMPACT_MATERIALIZE */ i1, i2 from t1);

(3) VIEW 안에
create view v1 as ( select /*+ COMPACT_MATERIALIZE */ i1, i2 from t1);
```

- COMPACT\_MATERIALIZE (view\_name)은 지원하지 않습니다.

```
with query1 as (select i1, i2 from t1)
select /*+ COMPACT_MATERIALIZE( query1) */ i1, i2 from query1
union all
select i1, i2 from query1;
```

- COMPACT WITH를 사용하지 않으려면 NO\_MATERIALIZE 힌트를 사용합니다.
- COMPACT\_MATERIALIZE 힌트는 \_\_OPTIMIZER\_WITH\_VIEW 값이 2 또는 4로 설정 시에만 적용됩니다.

## • 재현 방법

- 재현 절차
- 수행 결과
- 예상 결과

## • Workaround

## • 변경사항

- Performance view
- Property
  - 히든 프로퍼티 \_\_OPTIMIZER\_WITH\_VIEW 추가합니다.

## ■ 설명

COMPACT WITH 기법과 VIEW 최적화 적용 여부를 결정합니다.

- 0 : COMPACT WITH 기능을 비활성화합니다. COMPACT\_MATERIALIZE 힌트도 적용되지 않습니다.  
WITH 절로 생성된 뷰에 뷰 최적화(PUSH PROJECTION) 기능을 적용하지 않습니다.
- 1 : WITH 절로 생성된 뷰에 뷰 최적화(PUSH PROJECTION) 기능을 적용합니다.
  - Recursive WITH 뷰는 프로퍼티에 상관없이 뷰 최적화(PUSH PROJECTION)하지 않습니다.
  - COMPACT WITH 경우 뷰 최적화(PUSH PROJECTION) 기능을 적용합니다.
- 2 : COMPACT\_MATERIALIZE 힌트를 사용한 경우만 COMPACT WITH 기법을 적용합니다.
- 3 : 1+2 적용을 의미합니다.
- 4 : 조건을 만족하는 경우 COMPACT WITH 기법을 적용합니다.
  - WITH 절로 생성된 뷰가 뷰 머징(View Merging)이 불가능한 경우
    - WITH 뷰 내부에 View Merging이 안 되는 조건의 힌트(no\_merge, leading 등 힌트) 를 사용한 경우 COMPACT WITH로 유도됩니다.
  - View merge가 가능해도 from절에 뷰가 남아있는 경우
  - COMPACT\_MATERIALIZE 힌트를 사용한 경우
- 기본값
  - 1
- 속성
  - 변경 가능, 비공개
- Compile Option
- Error Code

## BUG-49250 Direct-Execute 방식의 statement를 반복 수행 중 prepare 에러가 발생한 경우 이전에 수행한 statement가 다시 실행될 수 있습니다.

- **module** : mm-jdbc
- **Category** : Functional Error
- **재현 빈도** : Always
- **설명** : Direct-Execute 방식의 statement를 반복 수행 중 prepare 에러가 발생한 경우 이전에 수행한 statement가 다시 실행되는 현상을 수정하였습니다.

Statement가 아래 조건을 만족하며 순차적으로 수행 시 발생할 수 있습니다.

1. direct execute #1
2. direct execute #2 (prepare 에러 발생)
3. prepare execute #3
4. direct execute #4 (#3의 statement가 실행)

- **재현 방법**
  - 재현 절차

```
Statement sStmt;
PreparedStatement sPstmt;
```



```

try {

    sStmt.execute("ALTER SESSION SET LOB_CACHE_THRESHOLD = 1048576");
} catch ( Exception e ) {

    System.out.println( "ERROR MESSAGE : " + e.getMessage() );

}

try {
    sStmt.execute("ALTER SESSION CLOSE DATABASE LINK");
} catch ( Exception e ) {
    System.out.println( "ERROR MESSAGE : " + e.getMessage() );
}

try {

    sPstmt = mConn.prepareStatement("ALTER SESSION SET LOB_CACHE_THRESHOLD =
1048576");
    sPstmt.execute();

} catch ( Exception e ) {

    System.out.println( "ERROR MESSAGE : " + e.getMessage() );

}

try {
    sStmt.execute("ALTER system CHECKPOINT");
} catch ( Exception e ) {

    System.out.println( "ERROR MESSAGE : " + e.getMessage() );

}

```

#### ○ 수행 결과

```

ALTER SESSION SET LOB_CACHE_THRESHOLD = 1048576
-----
ERROR MESSAGE : Property value overflow [LOB_CACHE_THRESHOLD]
ALTER SESSION CLOSE DATABASE LINK
-----
ERROR MESSAGE : SQL syntax error
line 1: parse error
ALTER SESSION CLOSE DATABASE LINK
                        ^
prepare-execute : ALTER SESSION SET LOB_CACHE_THRESHOLD = 1048576
-----
ERROR MESSAGE : Property value overflow [LOB_CACHE_THRESHOLD]
ALTER system CHECKPOINT
-----
ERROR MESSAGE : Property value overflow [LOB_CACHE_THRESHOLD]

```

#### ○ 예상 결과

```

ALTER SESSION SET LOB_CACHE_THRESHOLD = 1048576
-----
ERROR MESSAGE : Property value overflow [LOB_CACHE_THRESHOLD]

```

```

ALTER SESSION CLOSE DATABASE LINK
-----
ERROR MESSAGE : SQL syntax error
line 1: parse error
ALTER SESSION CLOSE DATABASE LINK
                        ^
prepare-execute : ALTER SESSION SET LOB_CACHE_THRESHOLD = 1048576
-----
ERROR MESSAGE : Property value overflow [LOB_CACHE_THRESHOLD]
ALTER system CHECKPOINT
-----

```

- **Workaround**

- **변경사항**

- Performance view
- Property
- Compile Option
- Error Code

**BUG-49263 다수의 파티션이 이중화에 포함된 파티션드 테이블에 연속적으로 TRUNCATE 수행 시 ERR-11041 : A deadlock situation has been detected. 에러가 발생할 수 있습니다.**

- **module** : rp-control
- **Category** : Functional Error
- **재현 빈도** : Rare
- **설명** : 다수의 파티션이 이중화에 포함된 파티션드 테이블에 DDL 수행 시 ERR-11041 : A deadlock situation has been detected. 에러가 발생하는 문제를 수정합니다.
- **재현 방법**
  - 재현 절차
  - 수행 결과
  - 예상 결과
- **Workaround**
- **변경사항**
  - Performance view
  - Property
  - Compile Option
  - Error Code

**BUG-49281 분석 함수 OVER절에 ORDER BY를 사용한 컬럼을 alias 사용하고 JOIN 의 ON절에 AND, OR 조건과 함께 사용한 경우 Altibase 서버가 비정상 종료하는 현상을 수정합니다.**

- **module** : qp-select-execute
- **Category** : Fatal
- **재현 빈도** : Always

- **설명** : 분석 함수 OVER절에 ORDER BY를 사용한 컬럼을 alias 사용하고 JOIN 의 ON절에 AND, OR 조건과 함께 사용한 경우 Altibase 서버가 비정상 종료하는 현상을 수정합니다.
- **재현 방법**
  - **재현 절차**

```
CREATE TABLE bug49281_t1 ( i1 CHAR(3), i2 VARCHAR(16), i3 VARCHAR(12));
INSERT INTO bug49281_t1 SELECT LEVEL, LEVEL, LEVEL FROM DUAL CONNECT BY LEVEL < 10;
CREATE TABLE bug49281_t2 ( i1 VARCHAR(17), i2 CHAR(2), i3 CHAR(3));
INSERT INTO bug49281_t2 SELECT LEVEL, LEVEL, LEVEL FROM DUAL CONNECT BY LEVEL < 10;
SELECT B.i1
  FROM (SELECT i1, i2, ROW_NUMBER() OVER (PARTITION BY i3 ORDER BY i1) AS RN
        FROM bug49281_t1 ) A
 RIGHT OUTER JOIN
    (SELECT i1, '00' AS PCD FROM bug49281_t2) B
 ON (A.RN <> 1 OR ( A.RN = 1 AND A.i2 = B.PCD));
```

- **수행 결과**

Altibase 서버 비정상 종료. 트레이스 로그 altibase\_error.log 에 아래와 같은 콜스택이 남습니다.

```
END-DUMP =====
BEGIN-STACK [CRASH] =====
Caller[0] 0000000000418FA1    => mmmSignalHandler
Caller[1] 00007FE8EFD845F0    => not found
Caller[2] 0000000000A68A9A    => mtdIsNull(mtcColumn const*, void const*)
Caller[3] 0000000000ACB707    => mtfIsNullCalculate(mtcNode*, mtcStack*, int,
void*, mtcTemplate*)
Caller[4] 0000000000AF9944    => mtfOrCalculate(mtcNode*, mtcStack*, int,
void*, mtcTemplate*)
Caller[5] 0000000000767418    => qtc::calculate(qtcNode*, qcTemplate*)
Caller[6] 0000000000767511    => qtc::judge(idBool*, qtcNode*, qcTemplate*)
Caller[7] 0000000000897846    => qmnFILT::doItFirst(qcTemplate*, qmnPlan*,
int*)
Caller[8] 000000000089794F    => qmnFILT::doIt(qcTemplate*, qmnPlan*, int*)
Caller[9] 000000000089C519    => qmnHASH::storeAndHashing(qcTemplate*,
qmnHASH*, qmndHASH*)
Caller[10] 000000000089C81B    => qmnHASH::init(qcTemplate*, qmnPlan*)
Caller[11] 00000000008A7B07    => qmnLOJN::doItLeft(qcTemplate*, qmnPlan*, int*)
Caller[12] 00000000008A7E4F    => qmnLOJN::doIt(qcTemplate*, qmnPlan*, int*)
Caller[13] 0000000000897824    => qmnFILT::doItFirst(qcTemplate*, qmnPlan*,
int*)
Caller[14] 000000000089794F    => qmnFILT::doIt(qcTemplate*, qmnPlan*, int*)
Caller[15] 0000000000899184    => qmnSORT::storeAndSort(qcTemplate*, qmncSORT*,
qmndSORT*)
Caller[16] 0000000000899389    => qmnSORT::init(qcTemplate*, qmnPlan*)
Caller[17] 00000000006CC0C0    => qmnPROJ::init(qcTemplate*, qmnPlan*)
Caller[18] 00000000006D4991    => qmx::executeSelect(qcStatement*)
Caller[19] 00000000004DCA7F    => qci::execute(qciStatement*, smiStatement*)
Caller[20] 000000000047D527    => mmcStatement::executedML(mmcStatement*, long*,
long*)
Caller[21] 00000000004743B6    => mmcStatement::execute(long*, long*)
```

```

Caller[22] 000000000043BEDE =>
_ZL9doExecuteP20mmtCmsExecuteContext.constprop.46
Caller[23] 000000000043CA0E =>
_ZN16mmtServiceThread7executeEP18cmiProtocolContextP12mmcStatement6idBoolPS4_PjP1
S7_.constprop.48
Caller[24] 000000000043E17F =>
mmtServiceThread::executeProtocol(cmiProtocolContext*, cmpProtocol*, void*,
void*)
Caller[25] 0000000000FA92D7 => cmiRecv(cmiProtocolContext*, void*,
PDL_Time_Value*, void*)
Caller[26] 000000000042ACFF => mmtServiceThread::executeTask_READY(mmcTask*,
mmcTaskState*)
Caller[27] 000000000042C875 => mmtServiceThread::executeTask()
Caller[28] 000000000042CE42 => mmtServiceThread::multiplexingAsShared()
Caller[29] 000000000042D4D7 => mmtServiceThread::run()
Caller[30] 0000000000FDED42 => idtContainer::staticRunner(void*)
Caller[31] 00007FE8EFD7CE65 => not found
Caller[32] 00007FE8EEE4788D => not found
END-STACK =====

```

#### ◦ 예상 결과

```

I1
-----
1
2
3
4
5
6
7
8
9
9 rows selected.

```

#### • Workaround

```

ON 절의 쿼리를 변환하여 Altibase 서버 비정상 종료를 회피할 수 있습니다.
ON (A.RN <> 1 OR ( A.RN = 1 AND A.i2 = B.PCD))
->
ON (A.RN <> 1 OR A.RN = 1 ) AND ( A.RN <> 1 OR A.i2 = B.PCD)

```

#### • 변경사항

- Performance view
- Property
- Compile Option
- Error Code

## BUG-49283 UNIQUE INSERT 수행 시 LOCK WAIT 상황에서 V\$SESSION\_WAIT에 no wait event가 발생합니다. wait event가 수집되지 않습니다.

- **module** : sm
- **Category** : Functional Error
- **재현 빈도** : Unknown
- **설명** : UNIQUE INSERT 수행 시 LOCK WAIT 상황에서 V\$SESSION\_WAIT 또는 V\$STATEMENT 에 wait event가 수집되지 않는 현상을 수정합니다.  
이 버그는 디스크 테이블에만 해당합니다. 메모리 테이블 경우 대기 이벤트가 없어 같은 상황에서 no wait event 발생하는 것은 정상 상황입니다.
- **재현 방법**
  - **재현 절차**

```
A 세션
iSQL> CREATE TABLE bug49283 (c1 INTEGER PRIMARY KEY) TABLESPACE
SYS_TBS_DISK_DATA;
iSQL> ALTER SYSTEM SET TIMED_STATISTICS = 1;
iSQL> AUTOCOMMIT OFF;
iSQL> INSERT INTO bug49283 VALUES(1);

B 세션
iSQL> AUTOCOMMIT OFF;
iSQL> INSERT INTO bug49283 VALUES(1);

A 세션
iSQL> SELECT * FROM V$LOCK_WAIT;
iSQL> SELECT SID, EVENT FROM V$SESSION_WAIT WHERE SID <> SESSION_ID();
iSQL> SELECT TX_ID, SESSION_ID, EVENT, SUBSTR(QUERY, 1, 60) QRY FROM V$STATEMENT
WHERE SESSION_ID <> SESSION_ID();
```

### ◦ 수행 결과

```
iSQL> SELECT * FROM V$LOCK_WAIT;
TRANS_ID          WAIT_FOR_TRANS_ID
-----
1408              21633
1 row selected.
iSQL> SELECT SID, EVENT FROM V$SESSION_WAIT WHERE SID <> SESSION_ID();
SID              EVENT
-----
2                no wait event
1 row selected.
iSQL> SELECT TX_ID, SESSION_ID, EVENT, SUBSTR(QUERY, 1, 60) QRY FROM V$STATEMENT
WHERE SESSION_ID <> SESSION_ID();
TX_ID          SESSION_ID  EVENT          QRY
-----
-
```

```

1408                2                no wait event                INSERT
INTO bug49283_m VALUES(1)
1 row selected.

```

#### ○ 예상 결과

```

iSQL> SELECT * FROM V$LOCK_WAIT;
TRANS_ID                WAIT_FOR_TRANS_ID
-----
1408                    21633
1 row selected.
iSQL> SELECT SID, EVENT FROM V$SESSION_WAIT WHERE SID <> SESSION_ID();
SID                EVENT
-----
2                enq: TX - row lock contention, data row
1 row selected.
iSQL> SELECT TX_ID, SESSION_ID, EVENT, SUBSTR(QUERY, 1, 60) QRY FROM V$STATEMENT
WHERE SESSION_ID <> SESSION_ID();
TX_ID                SESSION_ID  EVENT                QRY
-----
-
1408                2                enq: TX - row lock contention, data row  INSERT
INTO bug49283 VALUES(1)
1 row selected

```

- Workaround
- 변경사항
  - Performance view
  - Property
  - Compile Option
  - Error Code

## Changes

### Version Info

altibase version	database binary version	meta version	cm protocol version	replication protocol version
7.1.0.6.1	6.5.1	8.10.1	7.1.7	7.4.6

Altibase 7.1 패치 버전별 히스토리는 [Version Histories](#) 에서 확인할 수 있다.

### 호환성

## Database binary version

데이터베이스 바이너리 버전은 변경되지 않았다.

데이터베이스 바이너리 버전은 데이터베이스 이미지 파일과 로그파일의 호환성을 나타낸다. 이 버전이 다른 경우의 패치(업그레이드 포함)는 데이터베이스를 재구성해야 한다.

## Meta Version

메타 버전이 변경되었다.

패치를 롤백하려는 경우, [메타다운그레이드](#)를 참고한다.

## CM protocol Version

통신 프로토콜 버전은 변경되지 않았다.

## Replication protocol Version

Replication 프로토콜 버전은 변경되지 않았다.

## 프로퍼티

### 추가된 프로퍼티

- \_\_OPTIMIZER\_WITH\_VIEW

### 변경된 프로퍼티

### 삭제된 프로퍼티

## 성능 뷰

### 추가된 성능 뷰

### 변경된 성능 뷰

### 삭제된 성능 뷰