Altibase 7.2 New Features Guide

Altibase New Features Guide

Release 7.2

㈜알티베이스

08378 서울시 구로구 디지털로 306 대륭포스트타워II 10층

전화: 02-2082-1114 팩스: 02-2082-1099

고객서비스포털: http://support.altibase.com

homepage: http://www.altibase.com

Table of Contents

개요

Altibase 7.2 릴리즈 버전에 추가되거나 향상된 기능들을 소개한다.

Altibase 7.2 릴리즈 버전은 Altibase 7.2.0.0.1 이다. Altibase 7.2 릴리즈 이후 패치 버전에 추가되거나 향상된 기능들은 Altibase 7.2 패치 버전 별 새로운 기능 및 특징 에서 확인할 수 있다.

기능 개선

SQL 확장

CREATE QUEUE 및 ALTER QUEUE 구문에 DELETE 절 추가

큐(QUEUE) 테이블에 DELETE 문 허용 여부를 설정하는 DELETE 절이 추가되었다.

DELETE OFF로 DELETE 문을 허용하지 않으면 DELETE 문을 허용한 경우보다 DEQUEUE 병렬 수행 성능이 향상된다. 구문 사용 방법은 Altibase 7.2 SQL Reference 매뉴얼 을 참고한다. 관련하여 성능 뷰 V$QUEUE_DELETE_OFF 가 추가되었다.

범위 파티션드 객체에 파티션 추가 연산 지원

범위 파티셔드 테이블에 파티션 추가(ADD PARTITION) 구문을 지원한다. 이 기능 추가로 기본 파티션 없는 범위 파티션드 테이블 생성이 가능하다.

Altibase 7.2 에서 범위 파티션드 테이블 생성 시 주의 사항

- Altibase 7.2 에서는 기본 파티션이 없는 범위 파티션드 테이블을 생성할 수 있다.

  참고로 기본 파티션이 없는 범위 파티션드 테이블을 생성하면 SYS_TABLE_PARTITIONS_에서 PARTITION_NAME 이 없는 파티션이 추가로 생성된다.
- 범위 파티션드 객체에서 파티션 추가는 기본 파티션이 없는 범위 파티션드 테이블에서만 사용할 수 있다.
- 기본 파티션이 없는 범위 파티션드 테이블은 기본 파티션 추가 연산을 수행할 수 없다.
- 기본 파티션이 있는 범위 파티션드 테이블은 기본 파티션 삭제 연산을 수행할 수 없다.
- 기본 파티션이 없는 범위 파티션드 테이블은 CONJOIN/DISJOIN 구문을 사용할 수 없다.
- 범위 파티션드 테이블이 이중화 대상 테이블인 경우 파티션 추가 연산을 수행할 수 없다.

응용 프로그램 개발 인터페이스

JDBC API Specification 4.2 부분 지원 (PROJ-2707)

Altibase 7.2 에서 JDBC API Specification 4.2를 부분적으로 지원한다.

Altibase 7.2 JDBC 드라이버는 JRE 1.8 이상에서 동작한다. Altibase 7.2 JDBC 드라이버에서 지원하는 JDBC 4.2 API는 Altibase 7.2 JDBC User's Manual 에서 확인할 수 있다. 변경 사항 및 호환성 이슈는 Altibase JDBC 7.2 변경 사항 및 호환성 이슈에서 확인할 수 있다.

- **Auto-loading of JDBC driver class**

명시적으로 Class.forName() 클래스를 로딩할 필요없이 META-INF/services/java.sql.Driver 파일을 이용한 자동 드라이버 로딩 기능 지원

- **Wrapper Pattern Support**

  프록시에서 구현 객체에 대한 참조를 얻는 JDBC 4.0 표준 인터페이스를 지원한다. 커넥션풀 등에서 생성하는

```
try (Connection sWrappedCon = dbPool.getConnection()) {
    if (sWrappedCon.isWrapperFor(AltibaseConnection.class)) {
        AltibaseConnection connection = sWrappedCon.unwrap(AltibaseConnection.class);
        ...
        ...
}
```

- **National Character Set Support**

  JDBC 4.0 스펙인 표준 다국어 처리 인터페이스 지원

- **Aborting Connections**

  비동기적으로 데이터베이스와의 물리적 연결을 종료하는 Connection.abort() 인터페이스 지원

- **Standard Socket Network Timeout API Support**

  데이터베이스 서버로부터 소켓 응답 대기 시간을 설정하는 표준 인퍼페이스 Connection.setNetworkTimeout() 지원

- **Connection Management Enhancements**

  Validation Query없이 Connection 객체에서 유효성 검사를 수행하는 Connection.isValid() 지원

- **Large Update Counts Support**

  대용량 레코드 업데이트를 위한 executeLargeUpdate(), executeLargeBatch() 지원

- **Set Client Information Support**

  Connection.setClientInfo()를 이용한 클라이언트 어플리케이션 속성(name) 설정 지원

- **java.sql.SQLType interface Support**

  JDBC 4.2 표준 인터페이스 java.sql.SQLType을 구현한 AltibaseJDBCType 지원

- **Try-with-resources 구문을 통한 자동 JDBC 리소스 해제**

```
try (Statement stmt = con.createStatement()) {
      ResultSet rs = stmt.executeQuery(query);
      while (rs.next()) {
        String coffeeName = rs.getString("aaa");
        int supplierID = rs.getInt("bbb");
      }
    }
}
```

- **SQLException에 Enhanced for-each loop 사용**

```
catch(SQLException ex) {
     for(Throwable e : ex ) {
        LOG.error("Error occurred: " + e);
     }
}
```

- **커넥션풀 등에서 생성되는 proxy객체에서 실체 JDBC 객체 획득**

```
try (Connection sWrappedCon = dbPool.getConnection()) {
    if (sWrappedCon.isWrapperFor(AltibaseConnection.class)) {
        AltibaseConnection connection = sWrappedCon.unwrap(AltibaseConnection.class);
        ...
        ...
}
```

유틸리티

altiComp 커밋 카운트 설정 기능 추가

커밋(commit) 카운트를 설정할 수 있는 프로퍼티 COUNT_TO_COMMIT가 추가되었다. 관련 내용은 Altibase 7.2 Utilities Manual 에서 확인할 수 있다.

성능 및 안정성 향상

OLTP Scalability 성능 향상(TASK-7073)

- Linux x86-64 CPU 코어 수 24코어 이상에서 조회 트랜잭션 성능 저하 현상 개선
- 메모리 DB 삭제(DELETE) 트랜잭션 성능 향상을 위해 로깅 구조 개선
- 디스크 DB 변경 트랜잭션 성능 향상을 위해 In-place MVCC 동작 방식 개선
- 테이블 잠금(TABLE LOCK) 병목 개선
- INSERT/UPDATE 트랜잭션 처리 시 불필요한 트랜잭션 로그 기록을 제거
- 트랜잭션 로그파일 압축 시 메모리 할당/해제 병목 개선
    - 이와 관련한 영향도 확인
- 휘발성(Volatile) 메모리 DB 트랜잭션 성능 향상
- 커밋 병목 및 가비지 콜렉션 쓰레드 병목 개선
    - 트랜잭션 커밋 후 테이블 정보 업데이트 병목 개선
- 메모리 DB 트랜잭션 성능 향상
    - 디스크 읽기를 유발하는 함수의 병목을 제거
    - Group Commit Log 기능 추가

트랜잭션 로그 기록 성능 향상(TASK-6983)

로그 압축 알고리즘을 압축 속도가 빠른 LZ4 로 변경하였다.

휘발성/비휘발성 메모리 DB 트랜잭션 성능 향상

메모리 테이블 객체 식별자 추적 단계를 간소화하여 휘발성/비휘발성 메모리 DB 트랜잭션 성능이 향상되었다.

언두(undo) 테이블스페이스 재사용 안정성 향상

언두 테이블스페이스와 디스크 인덱스의 불필요한 관계를 제거하여 버그 발생 위험 요소 제거하였다. 디스크 페이지 공간 효율 개선으로 관련 프로퍼티들의 기본값 및 최대값이 변경되었다.

- INDEX_INITTRANS 최대값이 30에서 50으로 변경
- INDEX_MAXTRANS 기본값과 최대값이 30에서 50으로 변경

PARTITIONED TABLE에 대한 LIMIT FOR UPDATE 성능 개선

파티션드 테이블에 대한 LIMIT 연산 시 불필요한 범위 읽기 연산을 제거하여 성능을 개선하였다.

아래 조건을 모두 만족하는 경우에 성능 향상이 있다.

- LIMIT 절의 START VALUE가 1인 경우

```
-- START VALUE가 1 의 예
SELECT * FROM T1 LIMIT 1;       -- 내부적으로 limit start 1, count 1
SELECT * FROM T1 LIMIT 100      -- 내부적으로 limit start 1, count 100
```

- PROJECT 하위 노드가 파티션드 테이블의 각각의 파티션에 대한 스캔을 관리하는 노드(PARTITION-COORDINATOR) 이며
- PARTITION-COORDINATOR 노드의 모든 노드가 SCAN 인 경우

조건을 모두 만족하는 실행 계획은 아래와 같은 형태이다.

```
------------------------------------------------------------
PROJECT ( COLUMN_COUNT: 2, TUPLE_SIZE: 8, COST: 467.05 )
 PARTITION-COORDINATOR ( TABLE: SYS.T1, PARTITION: 4/4, FULL SCAN, ACCESS: 1, COST:
467.05 )
  SCAN ( PARTITION: P4, FULL SCAN, ACCESS: 0, COST: 116.76 )
  SCAN ( PARTITION: P3, FULL SCAN, ACCESS: 0, COST: 116.76 )
  SCAN ( PARTITION: P2, FULL SCAN, ACCESS: 0, COST: 116.76 )
  SCAN ( PARTITION: P1, FULL SCAN, ACCESS: 1, COST: 116.76 )
------------------------------------------------------------
```

서브쿼리의 인라인 뷰에 ORDER BY절 사용 시 SQL 성능 개선

조건절(WHERE, HAVING 절)에서 사용한 서브쿼리의 인라인뷰에 ORDER BY절이 있는 경우 OBYE(Order By Elimination, 불필요한 ORDER BY 제거) 쿼리 변환을 적용하여 SQL 성능이 향상되었다.

- SQL 사용 예

```
SELECT *
  FROM T1
 WHERE I1 IN (SELECT /*+ NO_UNNEST */I1
                FROM (SELECT *
                        FROM T2
                       ORDER BY I2, I3));
```

> *이 영향을 받는 SQL의 실행 계획에 변화가 있다. SUBQUERY FILTER 안에 SORT 플랜 노드 없어진다.*

스칼라 서브쿼리(Scalar subquery) 성능 개선

스칼라 서브쿼리 결과가 단일 레코드인지 확인 과정에서 발생하는 오버헤드를 제거하여 성능을 개선하였다.

Altibase 이중화 성능 향상

이중화 Sender 성능 향상

- 압축 로그에서 이중화에 필요한 로그만 압축 해제하는 기능 추가
- xLog 압축 알고리즘을 LZO에서 LZ4로 변경

기타

프로퍼티 추가

- DBLINK_GLOBAL_TRANSACTION_LEVEL

메타 테이블 추가

- SYS_REPL_TABLE_OID_IN_USE

성능 뷰 추가

- V$REPL_REMOTE_META_INDEX_COLUMNS
- V$QUEUE_DELETE_OFF

Altibase 7.2 패치 버전 별 새로운 기능 및 특징

Altibase 7.2.0.0.2 의 새로운 기능 및 특징

디스크 트랜잭션 동시 수행 수 증가

동시 수행할 수 있는 디스크 트랜잭션 수가 512에서 16384로 변경되었다. 이에 따른 변경 사항은 아래와 같다.

- 관련 프로퍼티 TRANSACTION_SEGMENT_COUNT의 최댓값이 512에서 16384로 변경되었다.
- TRANSACTION_SEGMENT_COUNT 프로퍼티의 설정 값이 900을 초과하면 세그먼트 헤더 정보 파일(txSegEntry.hdr) 생성된다.
- 프로퍼티의 속성이 '변경 가능'에서 '읽기 전용'으로 변경되었다.

TRANSACTION_SEGMENT_COUNT 속성 변경

TRANSACTION_SEGMENT_COUNT 프로퍼티의 속성이 '변경 가능' 에서 '읽기 전용' 으로 변경되어 ALTER SYSTEM 으로 변경을 시도하면 ERR-0104E : The property [TRANSACTION_SEGMENT_COUNT] is read-only. 에러가 발생한다.

Altibase 분산 데이터베이스 시스템에서 주의 사항

이 주의 사항은 **DB Link를 이용한 Altibase 분산 데이터베이스 시스템 환경에 해당**한다.

Altibase 분산 데이터베이스 시스템에서 언두 테이블스페이스 백업 시 주의 사항

Altibase 분산 데이터베이스 시스템에서 분산 트랜잭션을 복구해야 할 경우, 장애 발생 전 시점의 언두 테이블스페이스에 접근하려면 세그먼트 헤더 정보 파일(txSegEntry.hdr)이 필요합니다. 그러므로 언두 테이블스페이스를 백업할 때 세그먼트 헤더 정보 파일도 함께 백업해야 한다.

백업/복구 종류	
데이터베이스 단위 백업 (ALTER DATABASE BACKUP DATBASE)	Altibase 서버가 자동으로 세그먼트 헤더 정보 파일을 백업한다.
테이블스페이스 단위 백업 (Tablespace-Level Backup)	데이터베이스 관리자가 직접 세그먼트 헤더 정보 파일을 백업해야 한다.
데이터베이스 복구	매체 복구 과정에서 데이터베이스 관리자가 세그먼트 헤더 정보 파일의 백업 파일을 직접 복원해야 한다.
증분 백업 및 복구	Altibase 서버가 자동으로 세그먼트 헤더 정보 파일을 백업한다.

Altibase 분산 데이터베이스 시스템에서 TRANSACTION_SEGMENT_COUNT 변경 시 주의 사항

분산 트랜잭션이 남아있는 경우 TRANSACTION_SEGMENT_COUNT 설정값을 작게 변경하면 Altibase 서버가 구동되지 않을 수 있으니 TRANSACTION_SEGMENT_COUNT 프로퍼티 변경 전에 XA prepare 트랜잭션이 있는지 반드시 확인 후 진행해야 한다. XA prepare 트랜잭션은 V$TRANSACTION.SESSION_ID 값으로 확인할 수 있다.

Adapter for JDBC, Adapter for Oracle에서 LOB 데이터 타입 지원

Adapter for JDBC, Adapter for Oracle에서 지원하는 데이터 타입에 LOB이 추가되었다.

Adapter for Oracle

Adapter for Oracle 에서 LOB 데이터 타입을 지원하려면 ADAPTER_LOB_TYPE_SUPPORT 프로퍼티 값을 1로 변경해야 한다. 관련 프로퍼티 및 제약 사항은 Adapter for Oracle 매뉴얼을 참고한다.

- Adapter for Oracle User's Manual - 2.설치와 설정 - 프로퍼티 - ADAPTER_LOB_TYPE_SUPPORT
- Adapter for Oracle User's Manual - 2.설치와 설정 - 프로퍼티 - ORACLE_ARRAY_DML_MAX_SIZE
- Adapter for Oracle User's Manual - 2.설치와 설정 - 프로퍼티 - ORACLE_ERROR_RETRY_COUNT
- Adapter for Oracle User's Manual - 2.설치와 설정 - 프로퍼티 - ORACLE_SKIP_ERROR
- Adapter for Oracle User's Manual - 3.사용법 - oraAdapter 제약조건 - LOB 데이터 타입 제약 사항

Adapter for JDBC

Adapter for JDBC에서 LOB 데이터 타입을 지원하려면 ADAPTER_LOB_TYPE_SUPPORT 프로퍼티 값을 1로 변경해야 한다. 관련 프로퍼티 및 제약 사항은 Adapter for JDBC 매뉴얼을 참고한다.

- Adapter for JDBC User's Manual - 2.설치와 설정 - 프로퍼티 - ADAPTER_LOB_TYPE_SUPPORT
- Adapter for JDBC User's Manual - 2.설치와 설정 - 프로퍼티 - OTHER_DATABASE_BATCH_DML_MAX_SIZE
- Adapter for JDBC User's Manual - 2.설치와 설정 - 프로퍼티 - OTHER_DATABASE_ERROR_RETRY_COUNT
- Adapter for JDBC User's Manual - 2.설치와 설정 - 프로퍼티 - OTHER_DATABASE_SKIP_ERROR
- Adapter for JDBC User's Manual - 3.사용법 - jdbcAdapter 제약조건 - LOB 데이터 타입 제약 사항

Altibase JDBC 드라이버 연결 속성 추가

연결 속성 getcolumns_return_jdbctype 추가

DatabaseMetaData.getColumns 메서드의 반환 결과 중 DATA_TYPE의 값을 정의하는 연결 속성 getcolumns_return_jdbctype이 추가되었다. true는 JDBC API에서 정의한 java.sql.Types의 SQL 데이터 형식으로 반환하고 false는 V$DATATYPE에 정의된 데이터 타입 형식으로 반환한다. 기본값은 false로, V$DATATYPE에 정의된 데이터 타입 형식으로 반환한다. JDBC API에서 정의한 java.sql.Types의 SQL 데이터 형식으로 반환하기를 원하면 true로 변경하고 애플리케이션을 재컴파일해야 한다.

관련 내용은 Altibase 7.2 JDBC User's Manual에서도 확인할 수 있다.

연결 속성 batch_setbytes_use_lob 추가

BLOB 타입 컬럼을 대상으로 PreparedStatement.setBytes()를 executeBatch()로 실행 시 BLOB 타입으로 처리하도록 변경하고 연결 속성 batch_setbytes_use_lob이 추가되었다. 이 때, Auto-commit 모드에서 이전과 다르게 LobLocator cannot span the transaction 에러가 발생할 수 있다. 이 경우 batch_setbytes_use_lob 연결 속성을 false로 변경하거나 Non Auto-commit 모드로 변경해야 한다.

관련 내용은 Altibase 7.2 JDBC User's Manual에서도 확인할 수 있다.

LISTAGG 함수의 반환 데이터 크기를 설정하는 Altibase 서버 프로퍼티 LISTAGG_PRECISION 추가

LISTAGG 함수의 반환 데이터 크기를 설정하는 Altibase 서버 프로퍼티 LISTAGG_PRECISION이 추가되었다. 기본값은 4000바이트이며 1부터 VARCHAR 최대 크기 32000바이트까지 값을 설정할 수 있다. LISTAGG 함수가 반환하는 데이터의 크기가 4000바이트를 초과하여 ERR-21061 : result of string concatenation is too long. 에러가 발생하면 이 프로퍼티 값을 변경하여 조치할 수 있다.

DB Link - 지역 서버와 원격 서버 데이터베이스 간 CHAR/VARCHAR 타입의 데이터 길이 변환 방식 개선

지역 서버와 원격 서버 데이터베이스 간 CHAR 또는 VARCHAR 타입의 데이터 길이 변환 방식를 개선하였다. 본 개선 사항을 적용하려면 추가된 AltiLinker 프로퍼티인 TARGETS 프로퍼티의 하위 프로퍼티 NLS_BYTE_PER_CHAR를 dblink.conf에 추가하고 0이 아닌 값을 사용해야한다.

WITH 절을 사용한 SQL의 수행 성능 개선

WITH절을 반복적으로 사용한 긴 문장의 SQL 수행 성능 개선

WITH 절을 반복적으로 사용한 긴 문장의 SQL에 COMPACT WITH 기법을 적용하여 SQL 성능을 개선하였다.

> COMPACT WITH 기법
> WITH 절로 생성한 뷰(Recursive with 제외)를 n번 사용 시 내부적으로 statement가 n+1개 생성되는 문제를 개선하기 위해 한 개의 statement를 공유하는 기법

WITH 구문에 ORDER BY 절이 사용된 경우 불필요한 ORDER BY 제거하여 성능 개선

WITH 구문에 ORDER BY 절이 사용된 경우 불필요한 ORDER BY 제거(Order By Elimination, OBYE) 기법을 적용하여 SQL 성능을 개선하였다.