[솔루션 기능 테스트]
BTL Data Integrator 기능 테스트 절차서

한국소프트웨어진흥원
공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007. 11. 20	0.1	초기 작성	손승일

목 차

<그림 차례>

1. 문서 개요

본 문서는 핵심 업무시스템에서 운영되는 공개SW 지원 솔루션의 부족 현상을 극복하고, 다양한 공개SW 지원 솔루션 확보 가속화를 위해 발굴된 ETL솔루션인 BTL Data Integrator(이하 BTL DI)의 기능성을 검증하기 위한 테스트 수행 절차 및 결과를 기술하기 위해 작성되었으며, Oracle에서 CUBRID로 데이터 마이그레이션을 수행하고자 하는 업체의 참고자료로 활용하기 위해 제작되었다.

가. 문서의 목적

다음과 같은 세부적인 목적을 달성하기 위하여 작성되었다.

- BTL DI 설치 절차 및 결과 기술.
- BTL DI를 이용하여 Oracle to CUBRID 데이터 마이그레이션 검증 절차 및 결과 기술.
- 기타 리눅스 OS(Asianux, Redhat EL) 간의 정합성 테스트 수행의 절차서로 사용.
- 진행 중 문제 발생 사항과 각각의 진행사항 기술.
- Oracle에서 CUBRID로 전환 도입을 검토하는 업체의 참고자료로 제공.
- 공개SW 지원 솔루션 확보 확대.

나. 본 문서의 사용방법

다음과 같은 방법으로 사용할 수 있다.

- GINUX 1.0 기반하에 BTL DI의 설치 절차 및 결과를 확인한다.
- GINUX 1.0 기반하에 BTL DI중요 기능 확인 절차 및 결과를 확인한다.
- 기타 리눅스 OS(Asianux, Redhat EL)간의 정합성 테스트 절차서로 사용한다.
- Oracle to CUBRID 데이터 마이그레이션 검토 시 참고 자료로 사용한다.
- BTL DI의 repository는 CUBRID가 사용되었다. 타 DBMS 사용 시는 설치과정이 약간 달라질 수 있다.

2. 테스트 절차 내역

가. BTL DI 설치

단계	항목/시험/결과	
1	시험항목	BTL DI 설치
	시험절차	1. repository용 DBMS 설치(CUBRID) 2. BTL DI 구동용 os계정 생성 3. repository용 DBMS 계정 생성 4. JDK 1.4 이상 설치 5. BTL DI 서버 설치 6. BTL DI 클라이언트 설치 7. BTL DI 서버 구동 8. 포트확인 9. 로그인
	시험결과	1. repository용 DBMS 설치 - BTL DI가 사용하는 repository용 DBMS를 설치한다. root 계정으로 로그인하여 아래의 명령으로 cubrid User를 생성한다. # useradd cubrid cubrid User의 Password를 생성한다. # passwd cubrid cubrid User로 로그인 하여 다운로드 받은 설치 파일을 실행하여 설치한다. # sh CUBRID-7.3.0.1085-x86-Linux-glibc234.sh Install CUBRID to /home/cubrid/CUBRID... In case a different version of the CUBRID product is being used in other machines, please note that the CUBRID 7.3 servers are only compatible with the CUBRID 7.3 clients and vice versa. Do you want to continue? (y/n) [Default: y] : y The CUBRID java stored procedure can not be used in this server, because java environment was not installed or could not be used. Do you want to continue? (y/n) [Default: n] :y CUBRID has been successfully installed. demodb has been successfully created. subway database has been successfully created.

1

```
If you want to use CUBRID, run the following commands
  % . /home/cubrid/.cubrid.sh
  % cubrid_service start
```

환경 설정을 위해 ./home/cubrid/.cubrid.sh을 실행하고 cubrid_service start 명령으로 서비스를 구동한다.

```
# ./home/cubrid/.cubrid.sh
# cubrid_service start

cms start ... OK
#
```

2. BTL DI 구동용 os계정 생성
 - 새로운 OS계정을 생성한다.
 - 생성된 계정의 환경변수 LANG이 한글을 지원하는 케릭터 셋인지 확인한다.

```
# useradd di3
```

3. repository용 DBMS 계정 생성
 - repository용으로 사용될 DBMS에 계정을 생성
  큐브리드매니저의 사용자 추가 기능을 통해 계정을 생성한다.

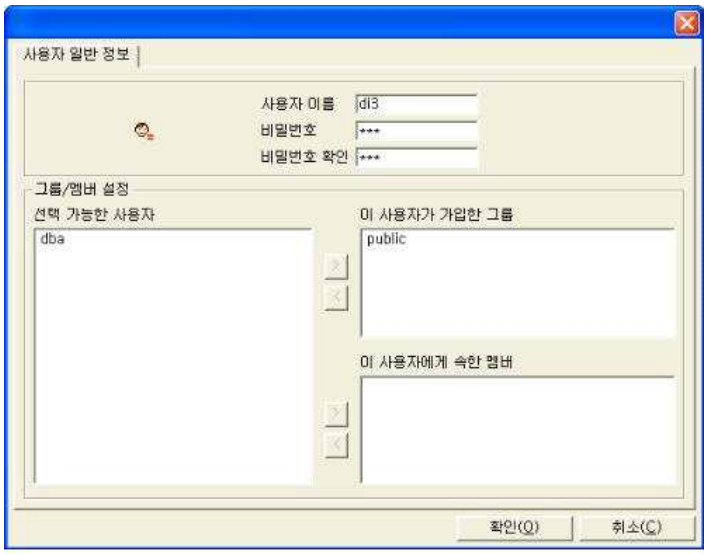


그림 1 DBMS 계정 생성

1		4. JDK 1.4 버전 설치 #sh j2sdk-1_4_2_16-linux-i586.bin #vi .bashrc export JAVA_HOME=/work/di3/j2sdk1.4.2_16 export PATH=$JAVA_HOME/bin:$PATH 5. BTL DI 서버 설치 - ftp프로그램을 이용하여 /work/di3로 서버프로그램을 업로드한다.(제공한 파일 목록중 server, shared 디렉토리를 업로드한다.) - server/bin디렉토리로 이동하여 install.sh 실행. config.sh을 비롯한 기타 실행파일이 생성된다. [di3@lopez bin]$ pwd /work/di3/server/bin [di3@lopez bin]$ ./install.sh /home/di3/di에 symbolic link를 생성합니다. 설치가 완료되었습니다. config.sh 파일을 설치 환경에 맞게 수정하시기 바랍니다. [di3@lopez bin]$ - config.sh 파일 설정 오라클을 리포지토리로 사용할 경우는 특별한 설정을 해야 하지만, 큐브리드가 리포지토리이므로 USE_ORACLE_REPOSITORY=0으로 설정한다. DI_HOME을 /work/di3 으로 설정한다. JAVA_HOME을 java가 설치된 디렉토리로 설정한다. #!/bin/sh ## # Oracle Repository 설정 # Oracle을 Repository로 사용하는 경우 USE_ORACLE_REPOSITORY를 1로 설정합니다. # Oracle을 Repository로 사용하지 않는 경우 지정하지 않아도 됩니다. # 64bit 호스트에서 32bit JVM을 사용하는 경우 USE_ORACLE_LIB32를 1로 설정합니다. # 환경변수의 정확한 값은 DBA에게 문의하시기 바랍니다. ##

1		``` ################### USE_ORACLE_REPOSITORY=0 USE_ORACLE_LIB32=1 ORACLE_HOME=/oracle NLS_LANG=American_America.KO16KSC5601 #NLS_LANG=American_America.KO16MSWIN949 #NLS_LANG=Korean_Korea.KO16MSWIN949 ## ################### # JRE 홈 디렉토리 설정 # ${JAVA_HOME}/bin/java가 존재하여야 합니다. # 예) /usr/local/java/j2sdk1.4.2_06 ## ################### JAVA_HOME=/usr/java/j2sdk1.4.2_15 JAVA=${JAVA_HOME}/bin/java ## ################### # DI 홈 디렉토리 설정 # DI 홈 디렉토리 아래에 server/ 와 shared/가 존재하여야 합니다. # 예) /home/user/di3 ## ################### DI_HOME=/work/di3 ``` - repository.xml 파일 설정. server/conf 디렉토리로 이동하여 vi repository.xml 을 실행한다. uri에서 서버의 IP, Port, database name을 수정한다. user, password부분에 생성했던 큐브리드 계정 Id와 Password를 소문자로 입력한다. ``` [di3@lopez bin]$ cd .. [di3@lopez server]$ cd conf [di3@lopez conf]$ vi repository.xml <?xml version="1.0" encoding="UTF-8"?> <repository active="cubrid"> ```

```
        <ha enable="false"/>
                <item columnDefaultNullable="true" dataSource="false"
truncate="false" dbms="CUBRID 7.1"
driver="cubrid.jdbc.driver.CUBRIDDriver" name="cubrid"
uri="jdbc:CUBRID:127.0.0.1:33000:demodb:::" validatingDml="SELECT 1
FROM DB_ROOT">
                        <param name="user" value="di3"/>
                        <param name="password" value="di3"/>
                </item>
                <supportedDbms>
                </supportedDbms>
</repository>
```

6. BTL DI 클라이언트 설치

- C:₩di3 디렉토리를 생성하고 제공받은 파일중 client와 shared 디렉토리를 카피하여 옮긴다.
- C:₩di3₩client₩bin₩di3 파일의 속성 다이얼로그를 열어서 대상, 시작위치 부분을 현재 디렉토리에 맞도록 수정한다.

1 di클라이언트디렉토리 위치와 java설치 위치 부분 수정

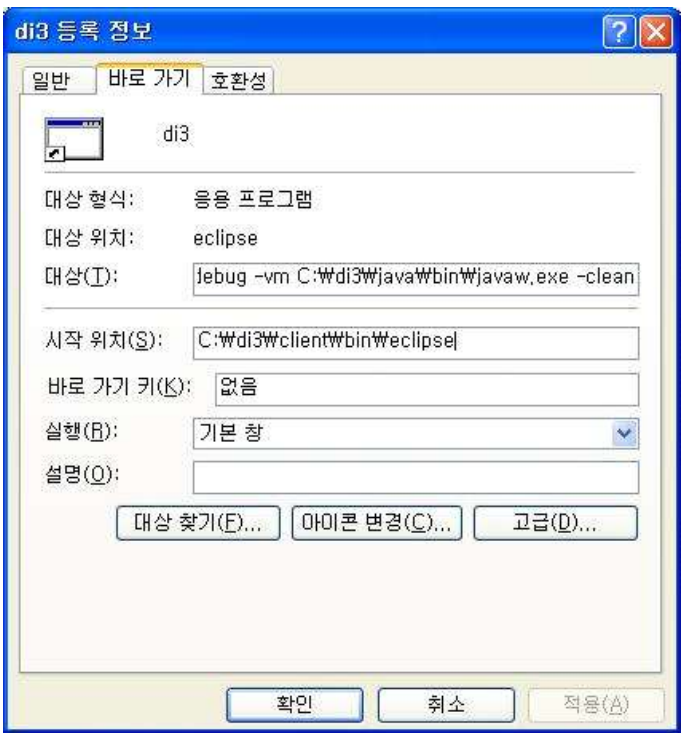


그림 2 BTL DI 클라이언트 설치

1		7. BTL DI 서버 구동 - di_server 커맨드 입력 - 정상 구동 확인. 마지막에 스케쥴러가 실행되면 정상구동이 완료된 상태이다. [di3@lopez di]$ ./di_server [di3@lopez di]$ Linux 2.4.20-8 i386 java.specification.version = 1.4 로깅 시스템을 초기화 합니다... 완료. 01:28:08 INFO : 시그널 핸들러를 설치합니다... 01:28:08 INFO : 완료. 01:28:08 INFO : 서버 환경을 설정합니다. 01:28:08 INFO : 완료. 01:28:08 INFO : 디렉토리를 구성합니다. 01:28:08 INFO : 완료. 01:28:08 INFO : 라이센스 정보를 검토합니다. 01:28:08 WARN : 라이센스가 등록되지 않아 사용에 제약이 따릅니다. 01:28:08 INFO : 완료. 01:28:08 INFO : Repository를 초기화 합니다. 01:28:22 INFO : 완료. 01:28:22 INFO : Core 쓰레드를 초기화 합니다. 01:28:22 INFO : 완료. 01:28:22 INFO : RMI 서버를 초기화 합니다. 01:28:22 INFO : 클라이언트에 알려줄 서버 주소가 지정되지 않아 0.0.0.0을 사용합니다. 01:28:22 INFO : 완료. 01:28:22 INFO : 서버의 초기화를 성공적으로 완료하였습니다. 01:28:22 INFO : 모니터를 시작합니다. 01:28:22 INFO : 스케쥴러를 시작합니다. 8. BTL DI 클라이언트 구동 - 클라이언트 실행 아이콘을 더블클릭하여 BTL DI 클라이언트 실행

1		 그림 3 BTL DI 클라이언트 구동 9. 로그인 -로그인 확인  그림 4 로그인

나. BTL DI 기능 테스트

<table>
<tr><th>단계</th><th colspan="2">항목/시험/결과</th></tr>
<tr><td rowspan="3">1</td><td>시험항목</td><td>BTL DI 구동</td></tr>
<tr><td>시험절차</td><td>1. BTL DI 서버 구동
2. BTL DI 클라이언트 구동
3. 로그인
4. 라이센스 키 입력</td></tr>
<tr><td>시험결과</td><td>1. BTL DI 서버 구동
- di_server 커맨드 입력
#./di_server

- 정상 구동 확인
#./di_server_check
UID PID PPID C STIME TTY TIME CMD
di3 4782 1 5 Nov21 ? 01:07:18
/work/di3/j2sdk1.4.2_16/bin/java -Xms256M -Xmx1024M
-XX:PermSize=32M -XX:MaxPermSize

2. BTL DI 클라이언트 구동
- 클라이언트 실행 파일 C:₩di3₩client₩bin₩di3을 더블클릭하여 BTL DI 클라이언트 실행
 

그림 5 BTL DI 클라이언트 구동</td></tr>
</table>

<table>
<tr><td rowspan="2">1</td><td></td><td>- 로그인
 

그림 6 로그인
- 라이센스 등록. 상단메뉴에서 도움말 -> 프로그램정보 -> 라이센스 정보 및 등록 을 선택한다. 제공받은 고객명, 라이센스 키를 입력한다.</td></tr>
<tr><td>비 고</td><td></td></tr>
<tr><td rowspan="3">2</td><td>시험항목</td><td>프로젝트 매핑</td></tr>
<tr><td>시험절차</td><td>1. 프로젝트 생성
2. 데이타베이스 연결 만들기
3. 테이블정의 가져오기
4. 잡 생성
5. 잡 디자인</td></tr>
<tr><td>시험결과</td><td>1. 프로젝트 생성
- 좌상단 리포지토리탐색기에 폴더를 우클릭하여 새로만들기->프로젝트만들기 메뉴를 선택한다.</td></tr>
</table>

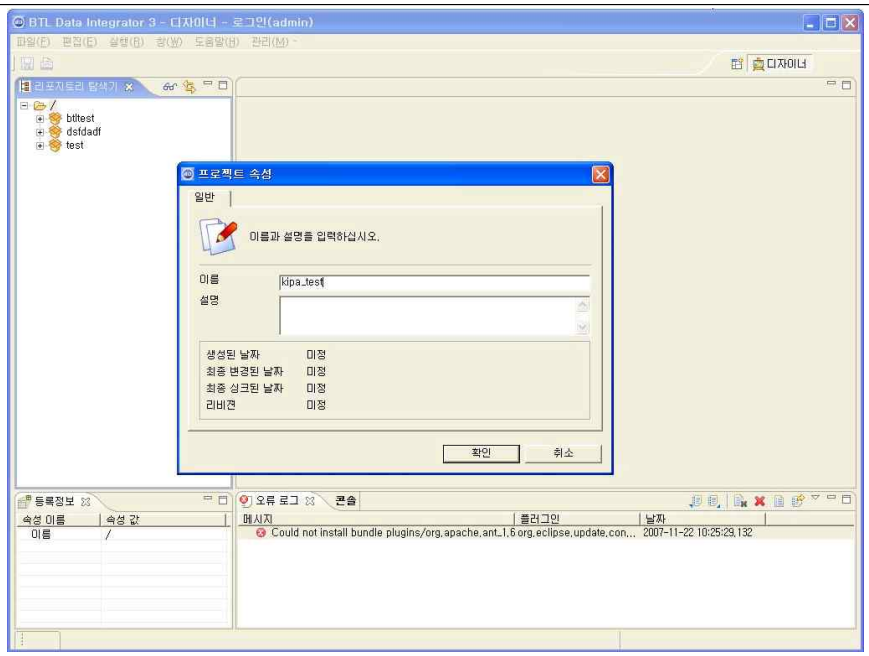


그림 7 프로젝트 생성

2

2. 데이타베이스 연결 만들기

- 생성된 프로젝트를 클릭-> 모델 우클릭하여 새로만들기 -> 데이타베이스연결 만들기 메뉴를 선택한다.

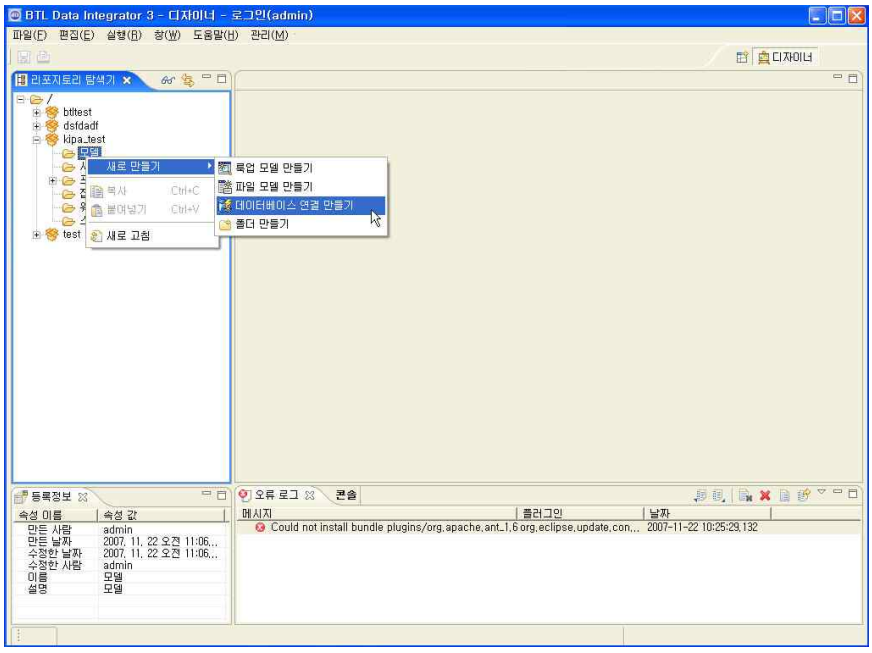
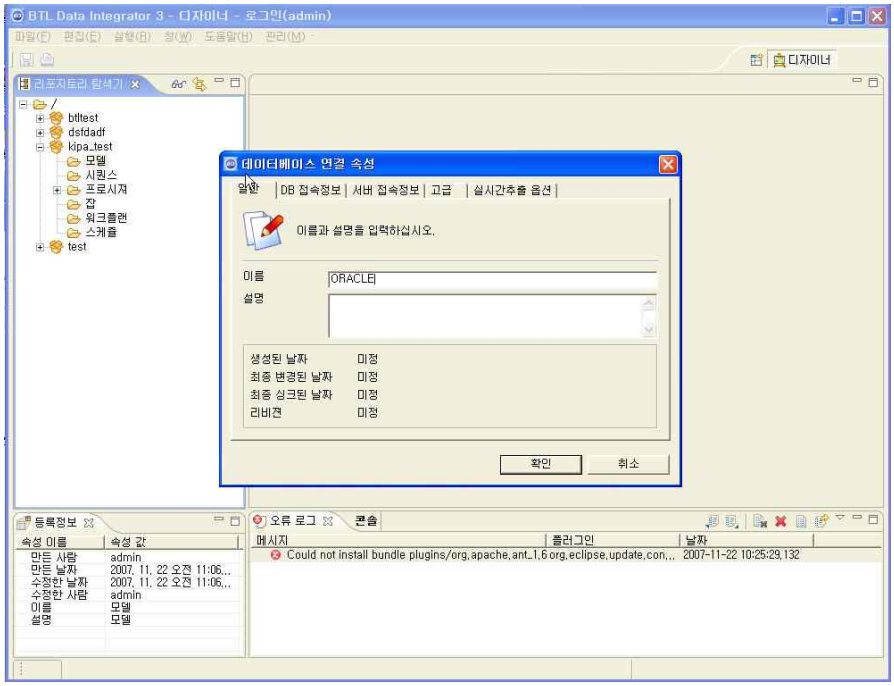
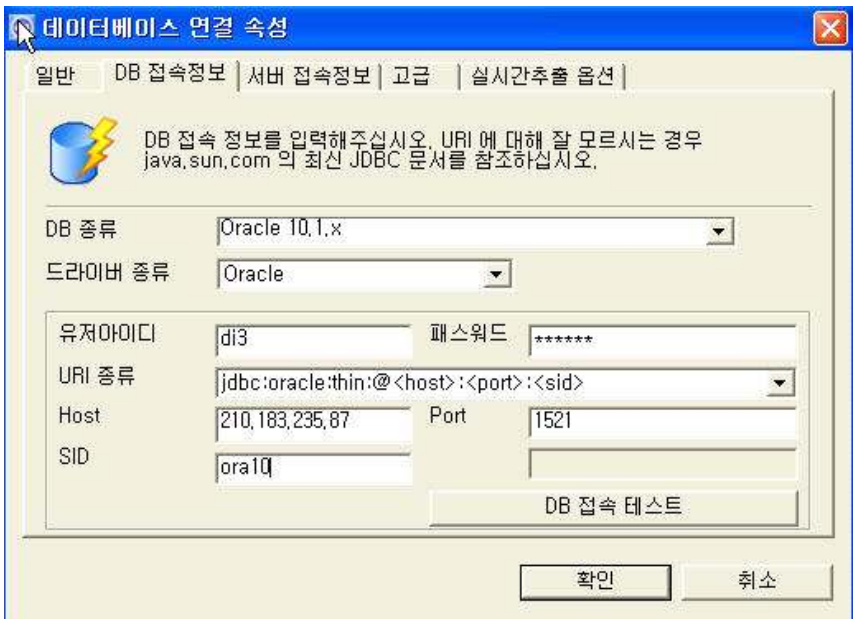
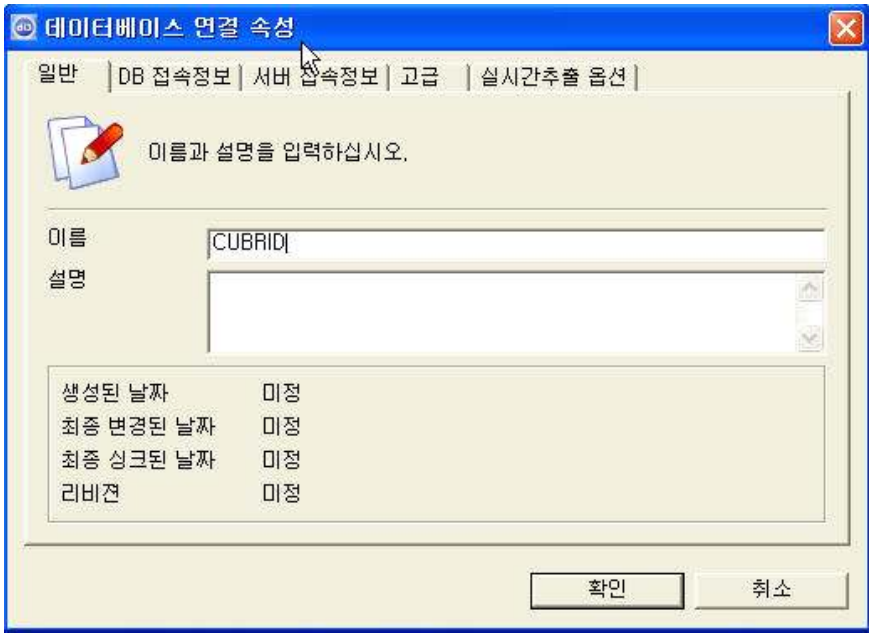
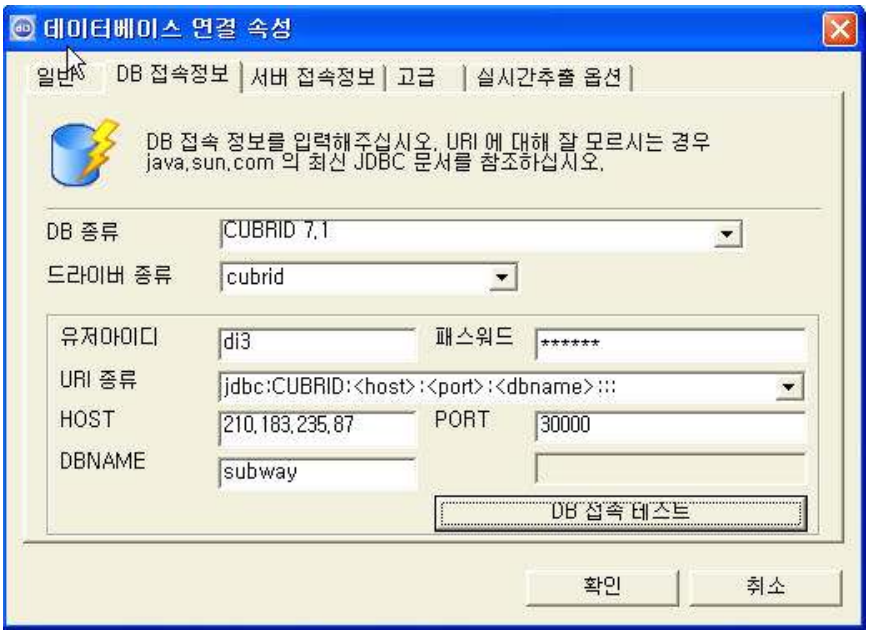
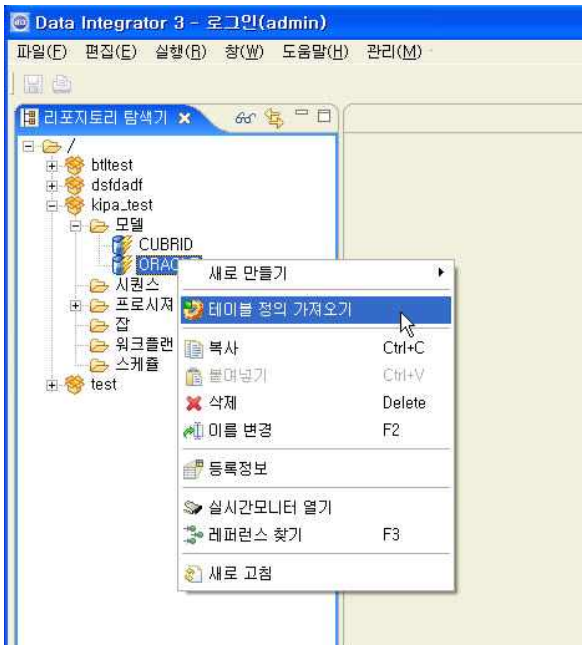
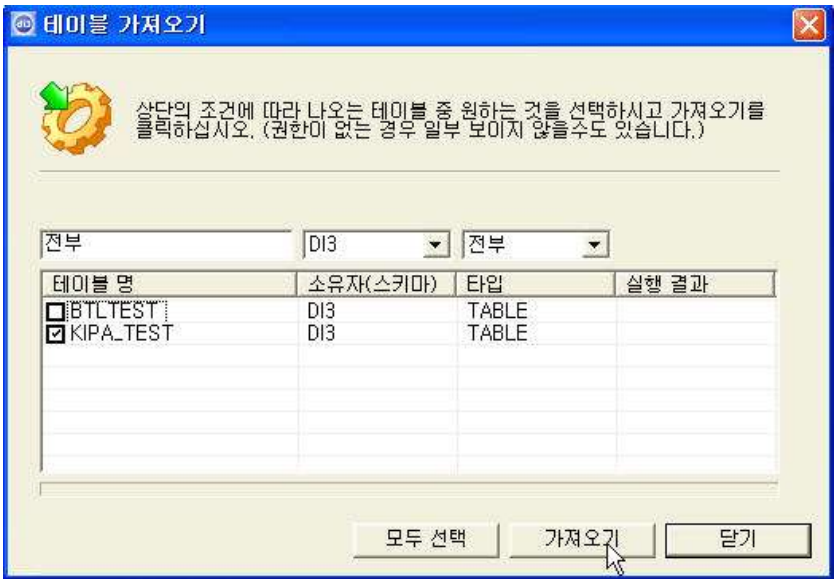
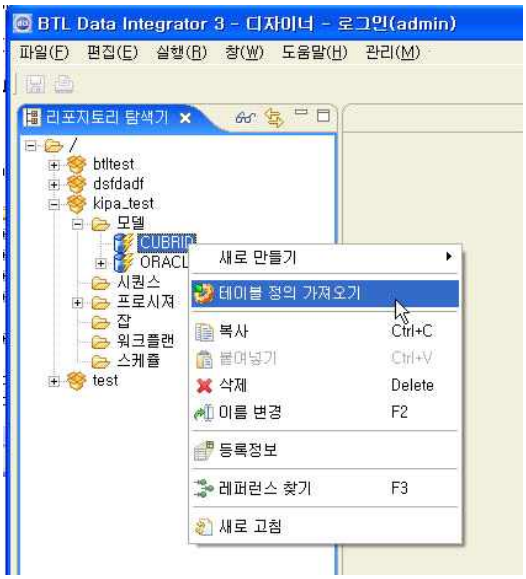
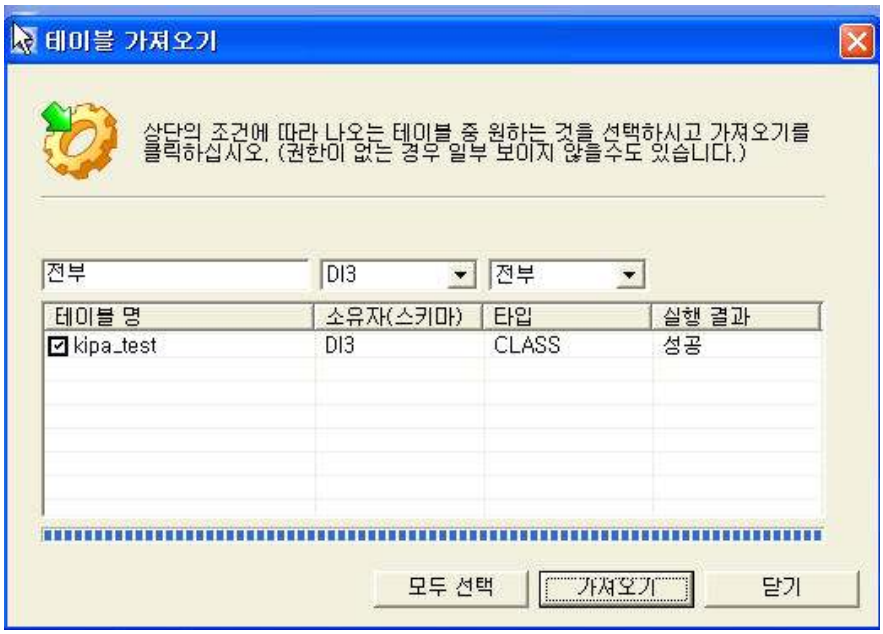


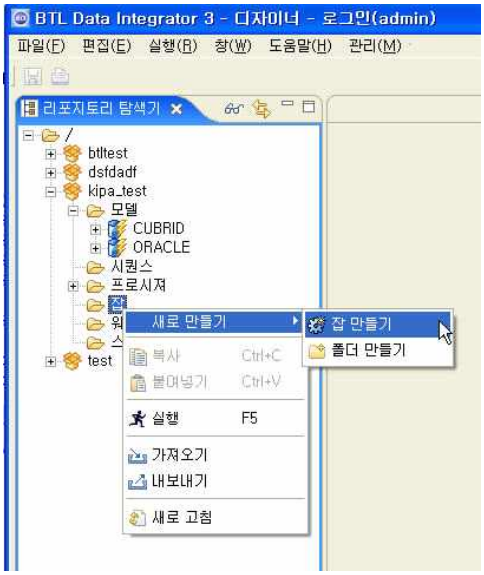
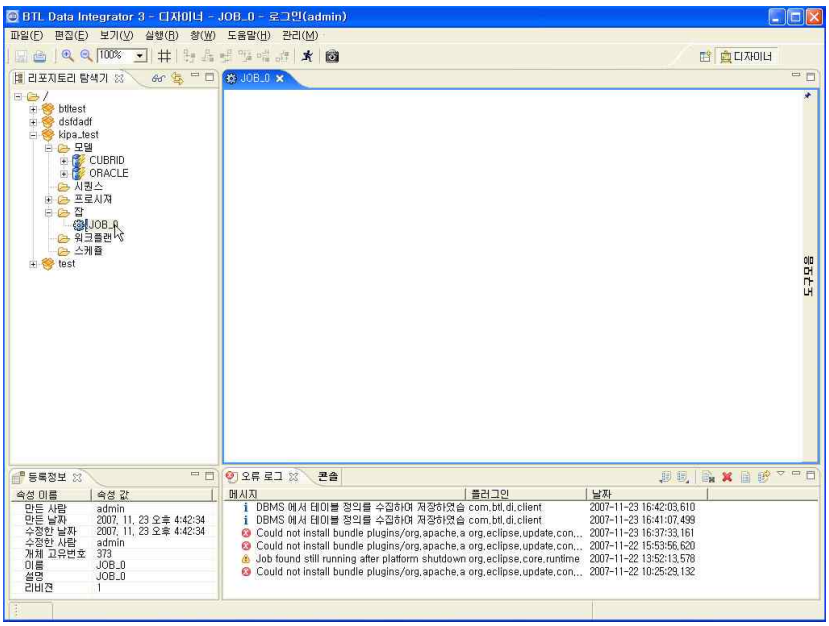
그림 8 데이타베이스 연결 만들기

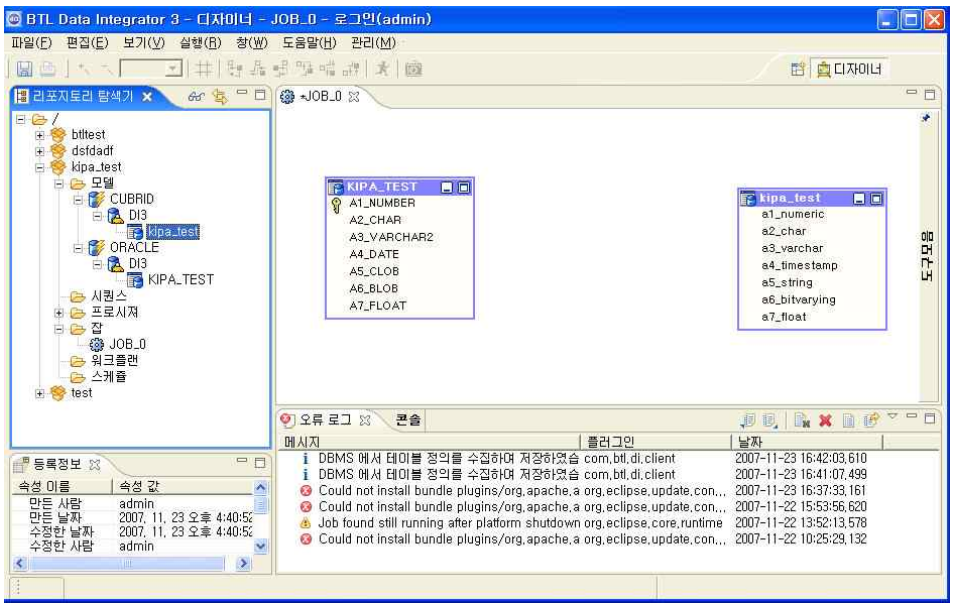
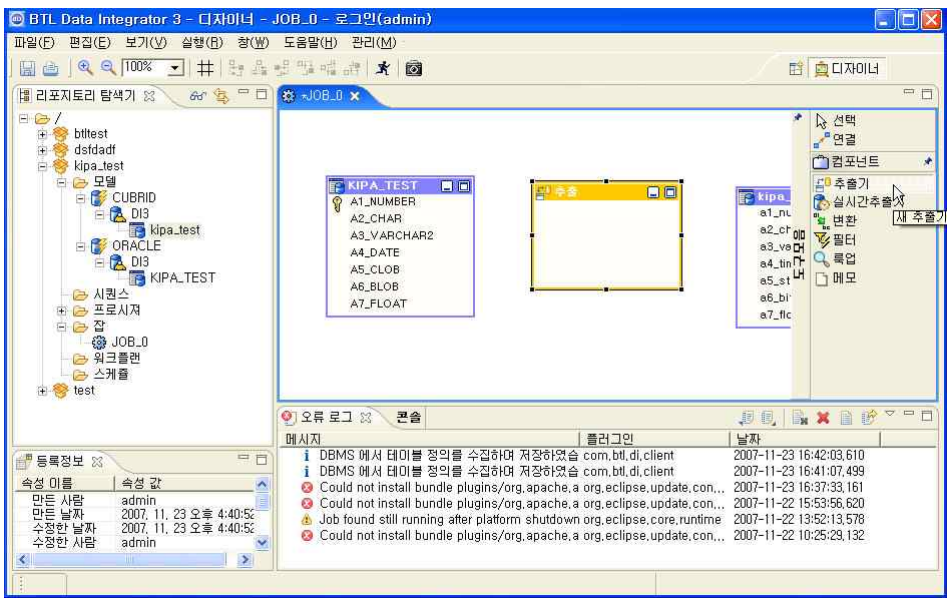
2		- 데이터베이스 이름 입력  그림 9 데이터베이스 이름 입력 - DBMS정보 입력(Oracle)  그림 10 DBMS정보 입력(Oracle)

2		- 위와 같은 절차를 거쳐서 CUBRID 데이타베이스연결을 생성한다.  그림 11 데이터베이스 이름 입력  그림 12 그림 10 DBMS정보 입력(CUBRID)

2		3. 테이블정의 가져오기 - 생성된 Oracle 데이터베이스연결을 우클릭하여 테이블정의 가져오기메뉴를 선택한다.  그림 13 테이블정의 가져오기 - 테스트용으로 생성한 테이블을 선택하여 가져오기 버튼을 클릭한다.  그림 14 테이블 가져오기

2		- 생성된 CUBRID 데이터베이스연결을 우클릭하여 테이블정의 가져오기메뉴를 선택한다.  그림 15 테이블 정의 가져오기 - 테스트용으로 생성한 테이블을 선택하여 가져오기 버튼을 클릭한다.  그림 16 테이블 가져오기

2		4. 잡 생성 - 잡 디렉토리를 우클릭하여 새로만들기-> 잡만들기 메뉴를 선택한다.  그림 17 잡 생성 5. 잡 디자인 - 생성된 잡을 더블클릭하여 잡에디터를 생성한다.  그림 18 잡 디자인

2		- 리포지토리탐색기에서 잡 에디터로 Oracle테이블과 CUBRID class를 드래그앤드랍한다.  그림 19 클래스 드래그앤드랍 - 잡에디터 우측의 도구메뉴를 열어서 추출을 클릭한 후 잡에디터의 원하는 위치에 다시 클릭하여 추출을 생성한다.  그림 20 추출 생성

		- 잡에디터 ORACLE class 컬럼을 선택하고 Ctrl+Alt 누른 상태에서 class 헤드를 클릭한 후 추출의 헤드를 클릭하여 연결 설정한다.

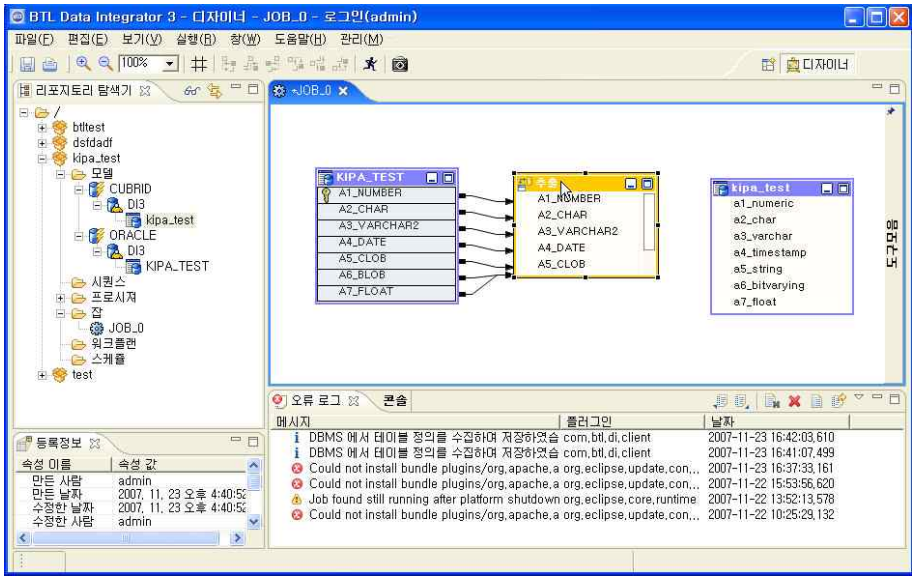


그림 21 연결 설정

2		- 잡에디터 추출의 컬럼을 선택하고 Ctrl+Alt 누른 상태에서 헤드를 클릭한 후 CUBRID의 class 헤드를 클릭하여 연결 설정한다.

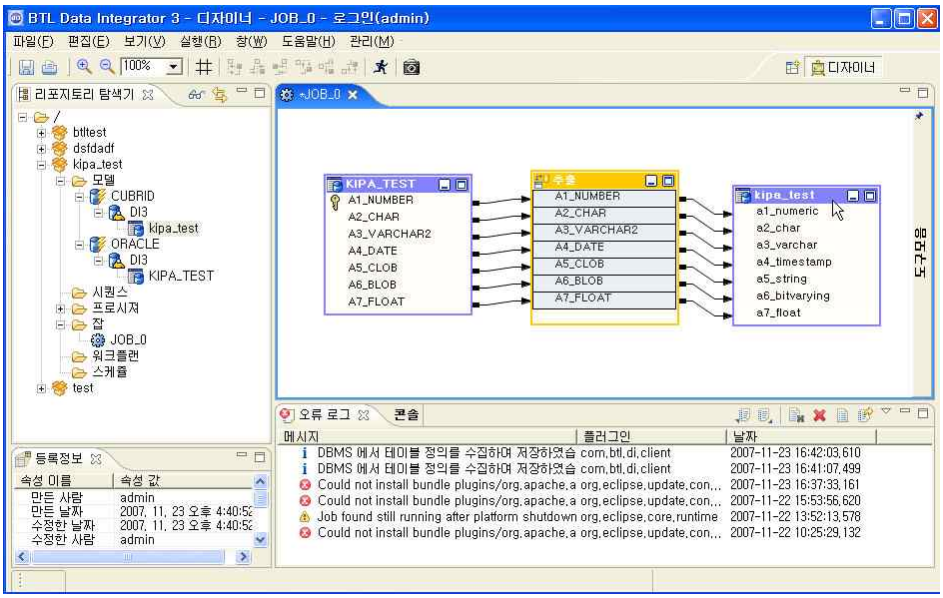
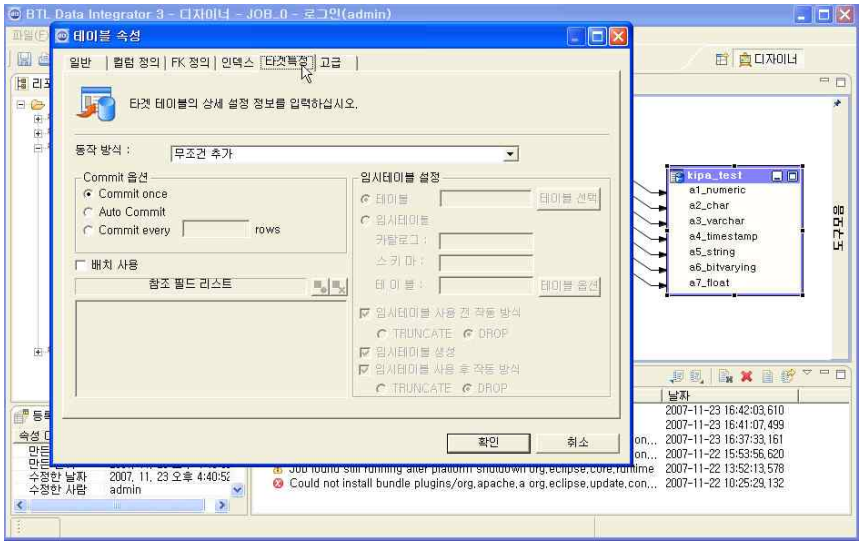
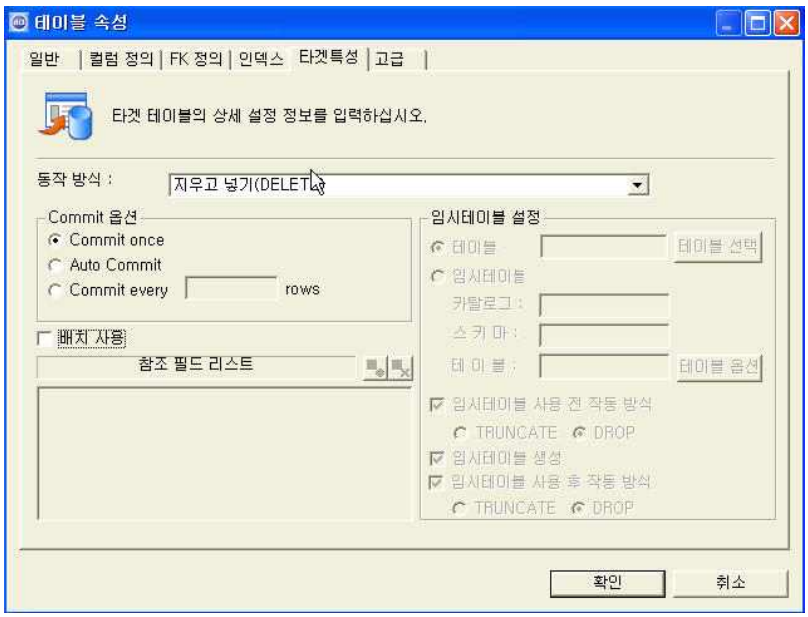
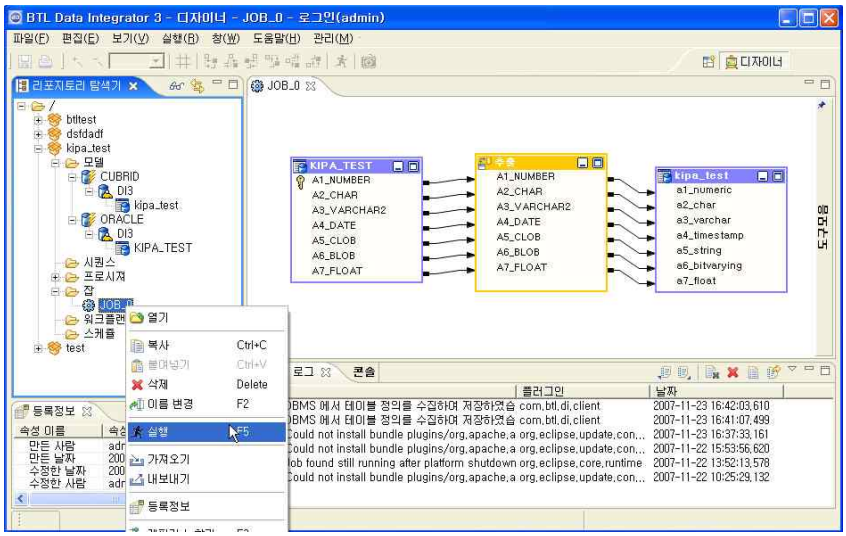
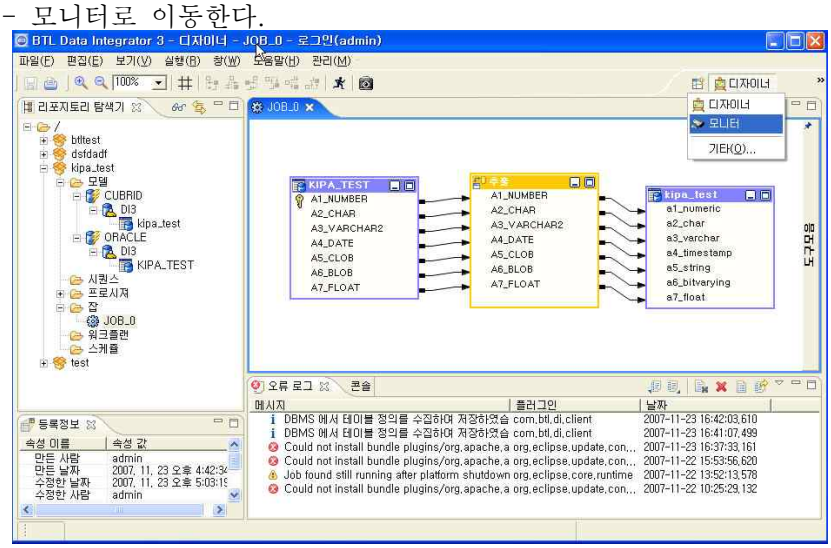


그림 22 추출에서 CUBRID로 연결 설정

2		- 잡에디터의 CUBID class를 더블클릭하여 다이얼로그를 오픈한 후 타겟특성탭으로 이동한다.  그림 23 타겟 특성탭으로 이동 - 동작방식에 지우고 넣기를 선택하고, 확인을 클릭하여 다이얼로그를 저장한다.  그림 24 동작방식 설정 - Ctrl + s 를 눌러서 잡에디터의 내용을 저장한다.
	비 고	

3	시험항목	잡 실행하기
	시험절차	1. 잡 실행 2. 모니터보기
	시험결과	1. 잡 실행 - 잡에디터가 활성화된 상태에서 F5키를 눌러 실행시킨다. - 또는 리포지토리탐색기에서 우클릭하여 실행메뉴를 클릭한다.  그림 25 잡 실행 2. 모니터보기 - 모니터로 이동한다.  그림 26 모니터로 이동

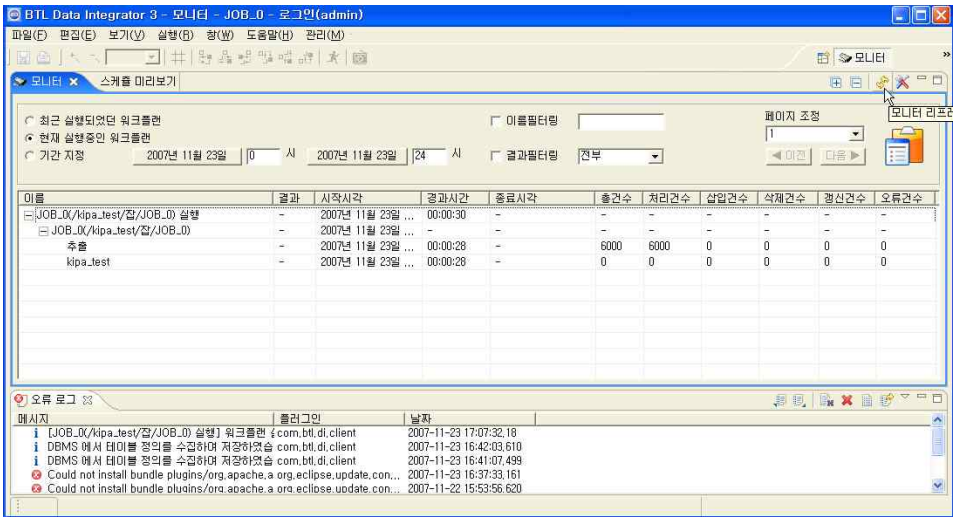
<table>
<tr><td rowspan="2">3</td><td></td><td>- 모니터뷰에서 현재 진행 상황을 확인한다.
 

그림 27 현재 진행 상황 확인</td></tr>
<tr><td>비 고</td><td></td></tr>
<tr><td rowspan="2">4</td><td>시험항목</td><td>워크플랜 생성 및 실행</td></tr>
<tr><td>시험절차</td><td>1. 워크플랜생성
2. 워크플랜에 SQL실행기 추가
3. SQL실행기에 쿼리 입력
4. 워크플랜에 커맨드라인 도구 추가
5. 커맨드라인 설정
6. 예외처리기 추가
7. 예외처리기 설정
8. 콤포넌트 매핑
9. 실행 및 결과 확인</td></tr>
<tr><td></td><td></td><td>1. 워크플랜생성
- 리포지토리탐색기의 워크플랜 디렉토리를 우클릭하여 새로만들기->워크플랜 만들기 메뉴를 선택한다.</td></tr>
</table>

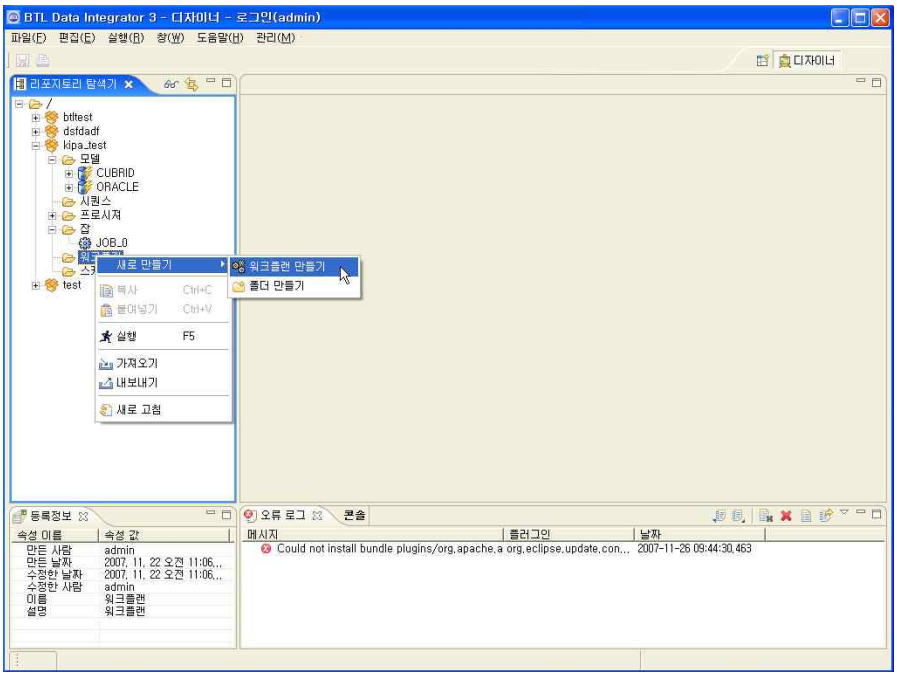


그림 28 워크플랜 생성

4

- 생성된 워크플랜을 더블클릭하여 워크플랜 에디터를 생성한다.

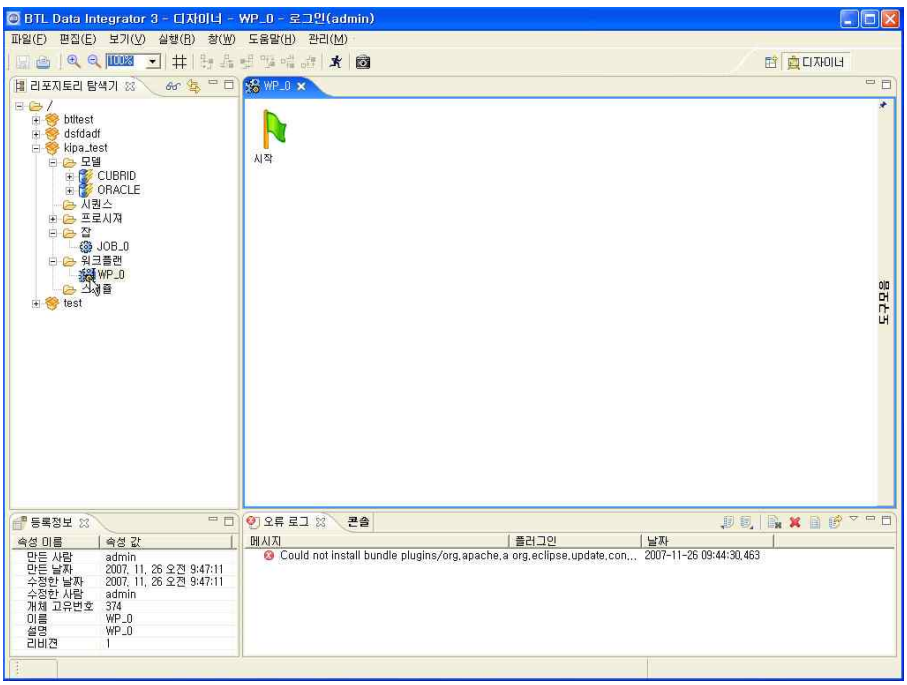
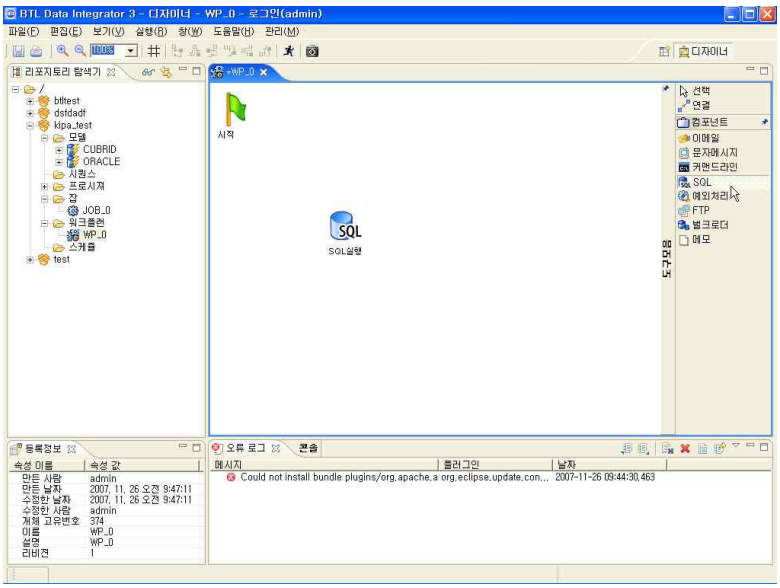
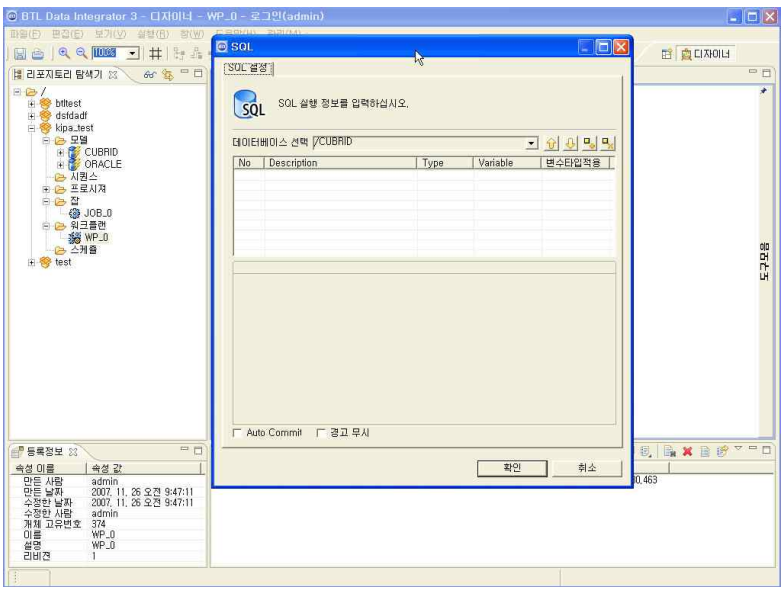
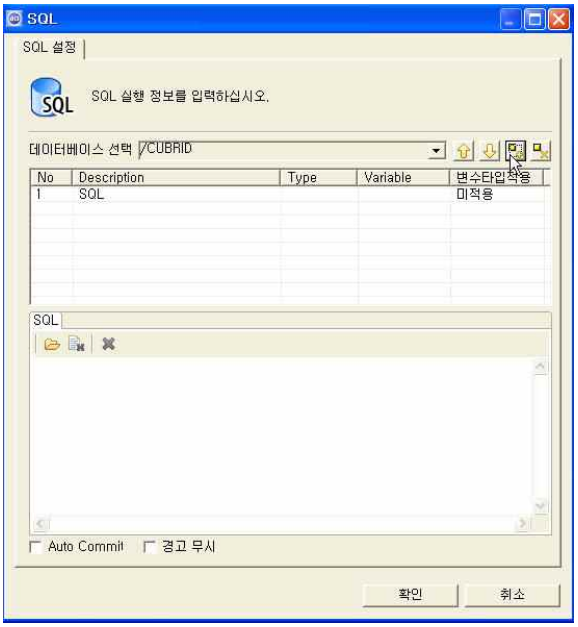
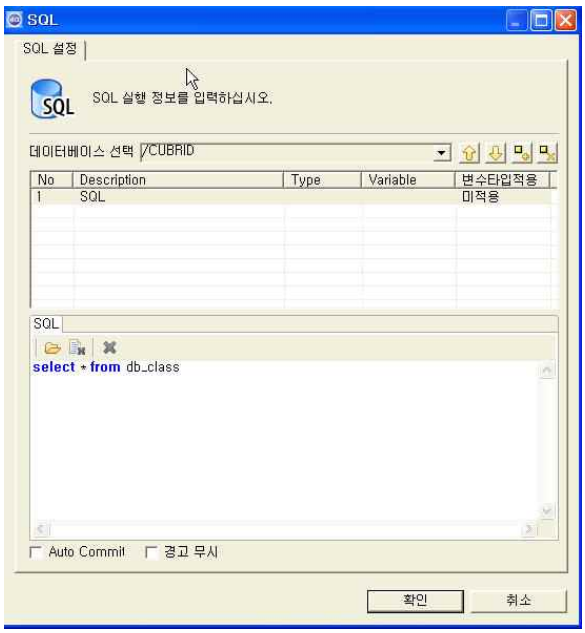
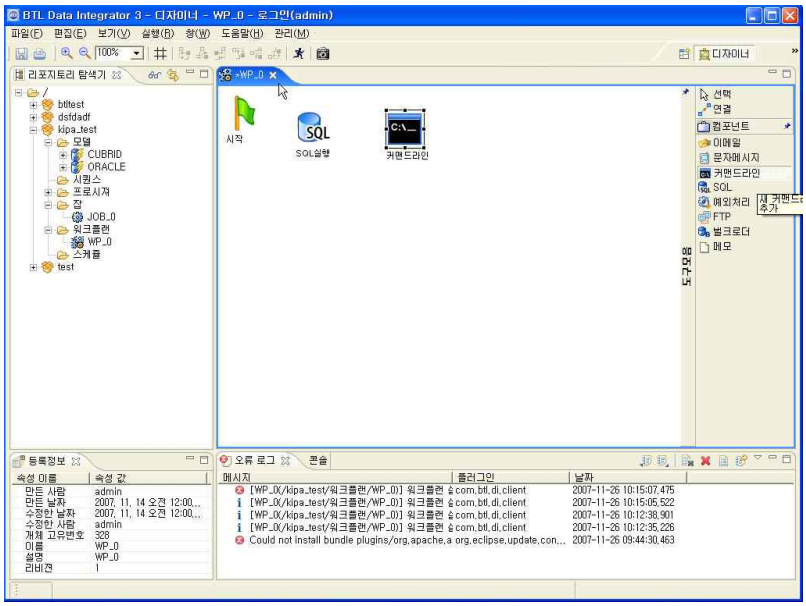
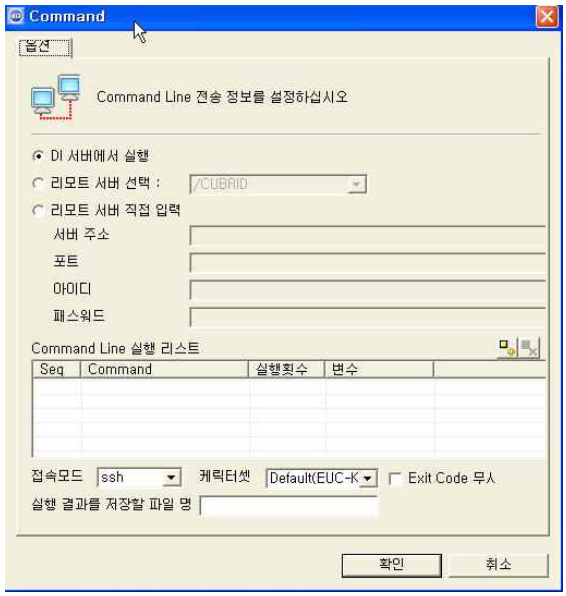
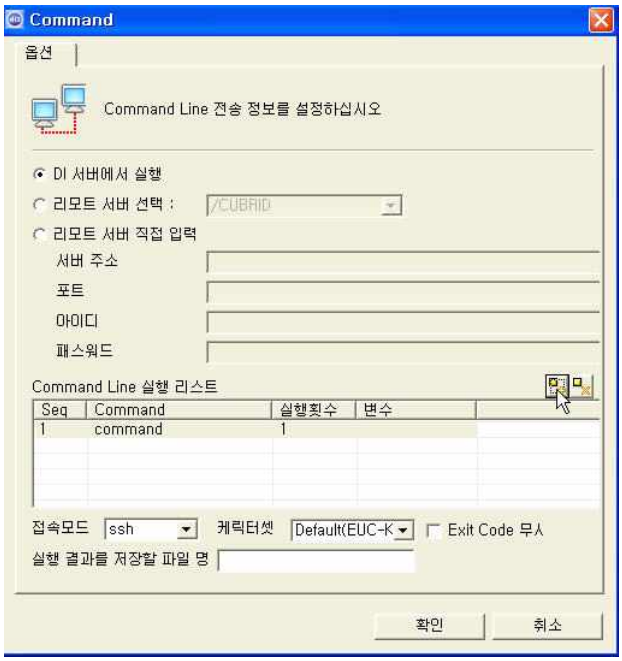
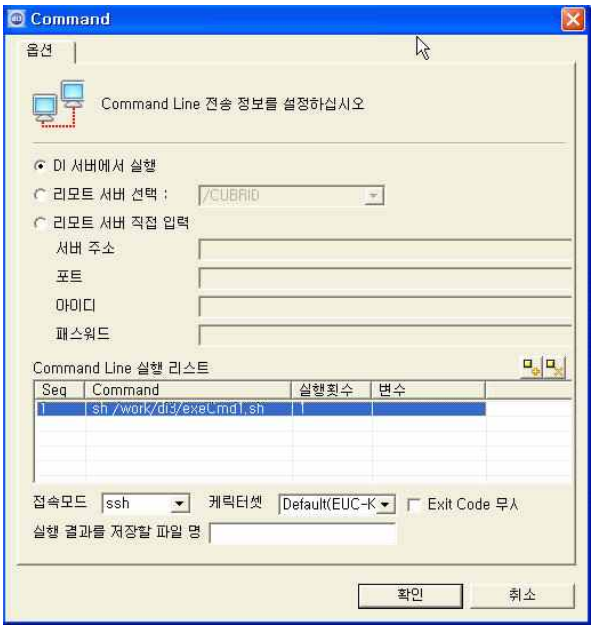


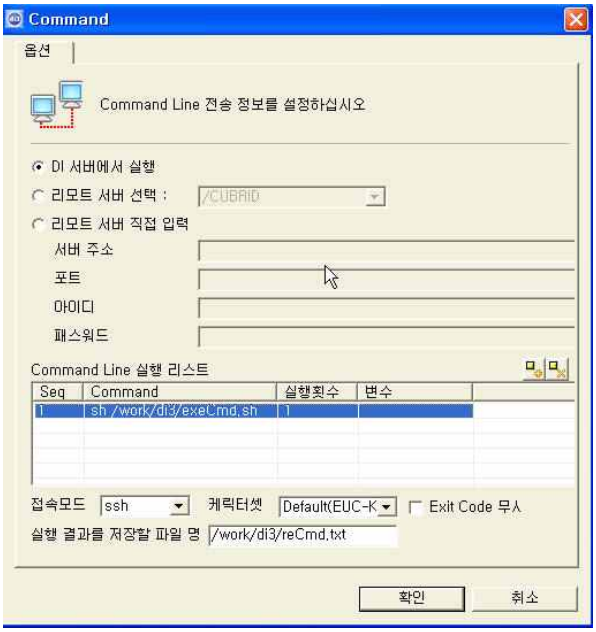
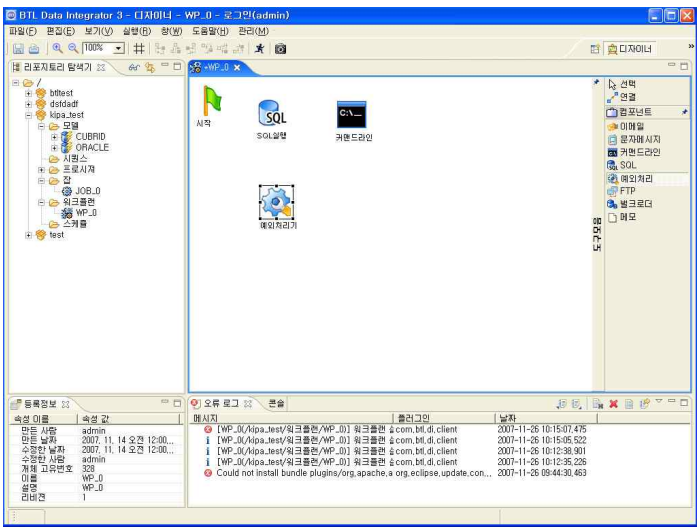
그림 29 워크플랜 에디터 생성

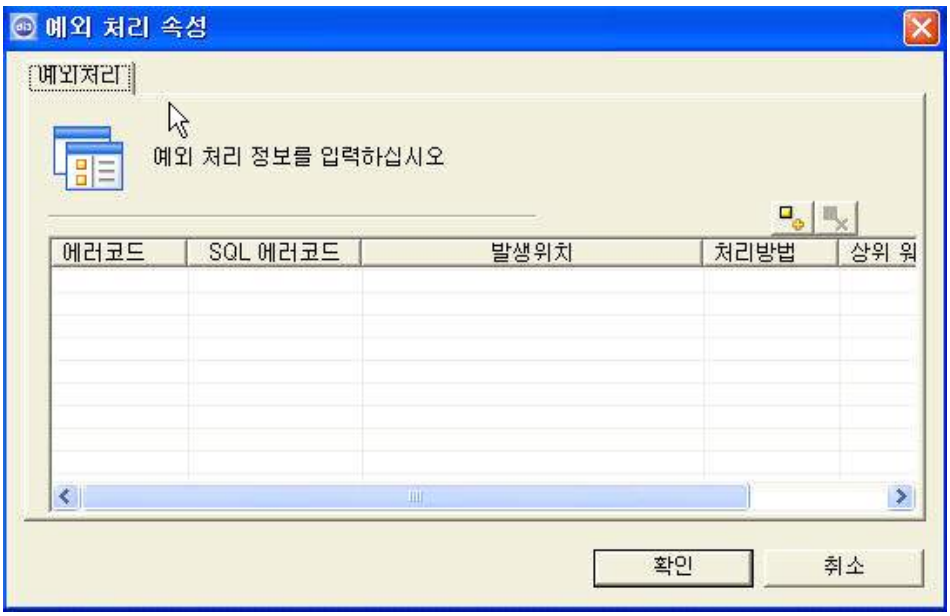
4		2. 워크플랜에 SQL실행기 추가 - 우측의 도구모음에서 SQL도구를 클릭한 후 워크플랜에디터의 원하는 위치에 다시 클릭한다.  그림 30 SQL 실행기 추가 3. SQL실행기에 쿼리 입력 - SQL을 더블클릭하여 상세설정 다이얼로그를 연다.  그림 31 SQL 실행기에 쿼리 입력

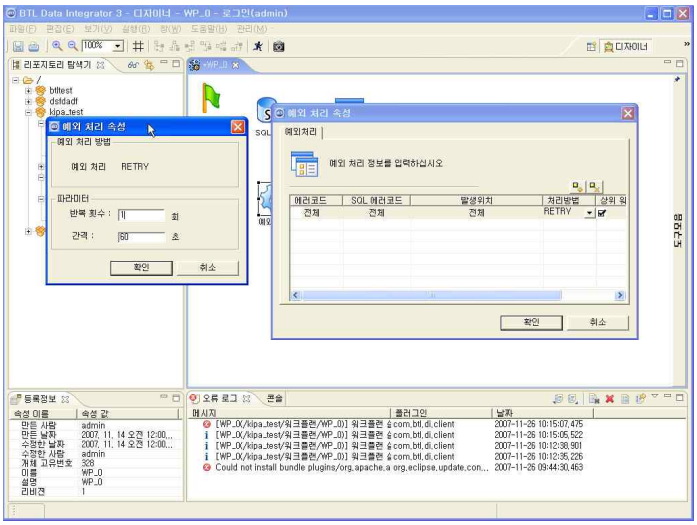
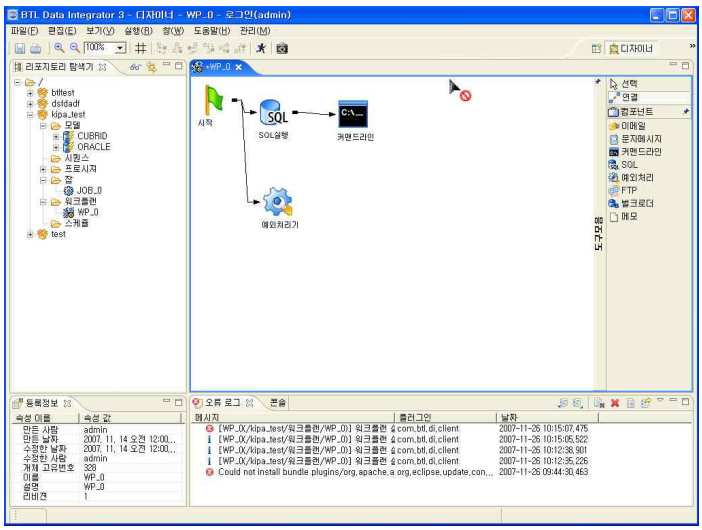
		- + 모양의 버튼을 클릭하여 새로운SQL탭을 생성한다.  그림 32 새로운 SQL탭 생성
4		- 실행하고자 하는 SQL문장을 입력한 후 확인을 눌러 저장한다. 만일 여러개의 SQL탭을 만들었다면 실행시 위에서부터 순차로 실행된다.  그림 33 SQL문장 저장

4		4. 워크플랜에 커맨드라인 도구 추가 - 우측의 도구모음에서 커맨드라인 도구를 클릭한 후 워크플랜에디터의 원하는 위치에 다시 클릭한다.  그림 34 커맨드라인 도구 추가 5. 커맨드라인 설정 - 커맨드라인을 더블클릭하여 상세설정 다이얼로그를 연다.  그림 35 커맨드라인 설정

4		- DI 서버에서 실행을 선택한 후 + 버튼을 눌러 커맨드 명령어를 추가한다.  그림 36 커맨드 명령어 추가 - 실행하고자 하는 스크립트파일의 이름을 절대경로를 포함하여 입력한다.  그림 37 실행 스크립트 파일 입력

4		- 접속모드는 리모트서버 직접입력에서 필요한 옵션이므로 무시한다. - 실행결과를 저장할 파일명에 절대경로를 포함한 파일이름을 입력한다.  그림 38 결과 저장 파일 경로 입력 6. 예외처리기 추가 - 우측의 도구모음에서 예외처리기 도구를 클릭한 후 워크플랜에디터의 원하는 위치에 다시 클릭한다.  그림 39 예외처리기 추가

4		7. 예외처리기 설정 - 예외처리기를 더블클릭하여 상세설정 다이얼로그를 연다.  그림 40 예외처리기 설정 - + 버튼을 눌러서 새로운 예외케이스를 생성한다.  그림 41 예외케이스 생성

		- 처리방법을 클릭하여 RETRY를 선택한다. - 생성된 예외처리속성 다이얼로그에서 반복횟수는 1회로 수정한다. 확인을 눌러 저장한다.  그림 42 반복횟수 설정
4		8. 콤포넌트 매핑 - 도구모음의 연결을 클릭한 후 선을 그을 시작점을 클릭하고 다시 도착점을 클릭하면 선이 연결된다. 단축키로는 Ctrl+Alt + 클릭을 이용할 수 있다. Ctrl +Alt를 누른 상태에서 시작점클릭 -> 도착점클릭으로 선을 연결할 수 있다. 연결선의 최초시작점은 시작 콤포넌트이어야만 한다.  그림 43 콤포넌트 매핑

4

9. 실행 및 결과 확인

- 워크플랜에디터에 모두 선을 연결했다면 F5를 눌러서 실행한다.

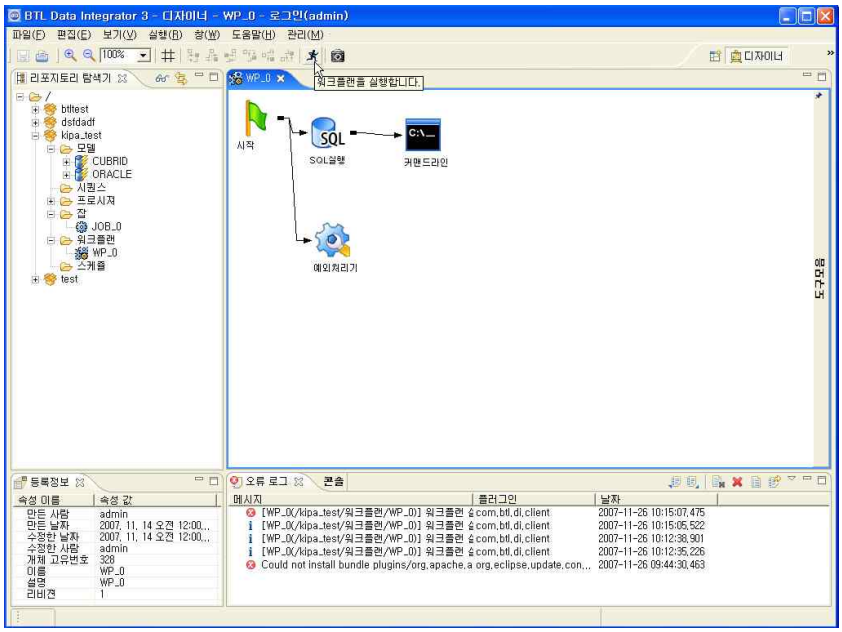


그림 44 워크플랜 실행

- 실행이 완료되면 워크플랜 실행결과 다이얼로그가 열린다.

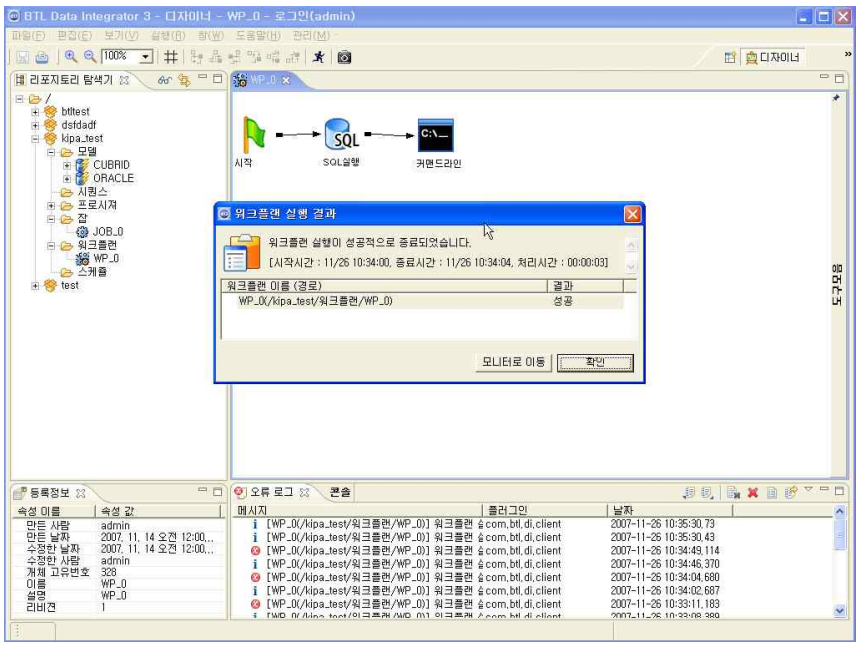
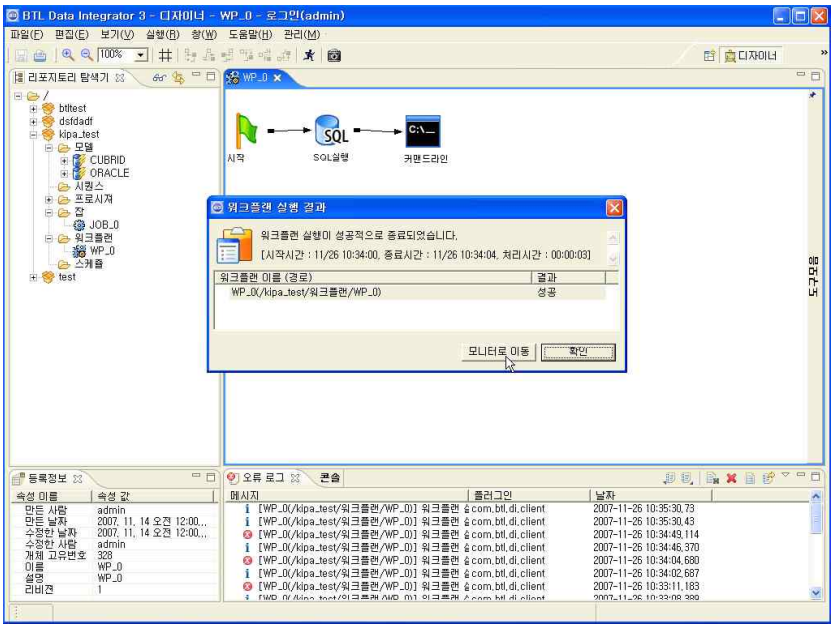
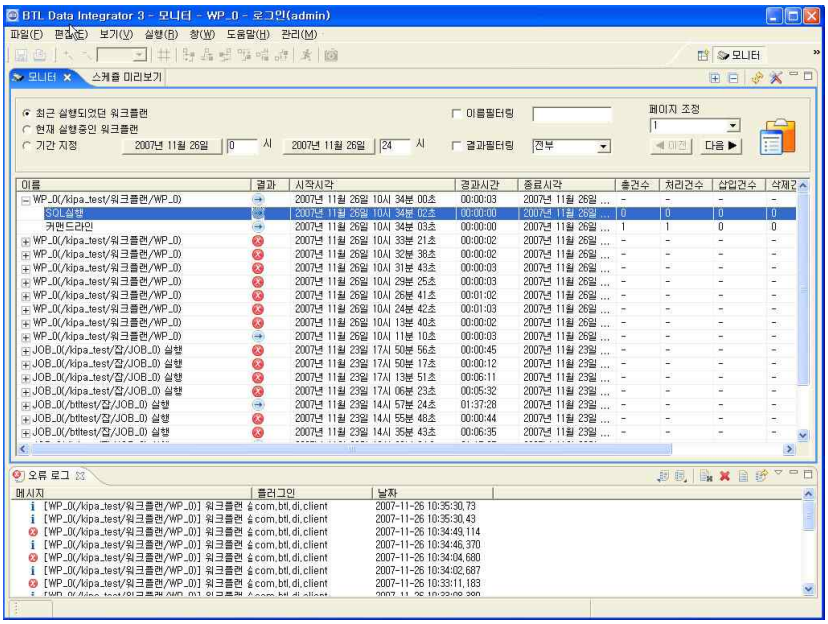


그림 45 실행결과 다이얼로그

4		- 워크플랜 실행결과 다이얼로그의 워크플랜항목을 더블클릭하거나 모니터로이동 버튼을 클릭하여 모니터화면으로 이동한다.  그림 46 모니터로 이동 - 실행결과를 확인한다.  그림 47 실행결과 확인

4

- 워크플랜을 우클릭한 후 리포지토리에서찾기 메뉴를 선택하여 디자이너 화면으로 이동한다. 모니터화면 우상단의 디자이너버튼을 눌러서 이동할 수도 있다.

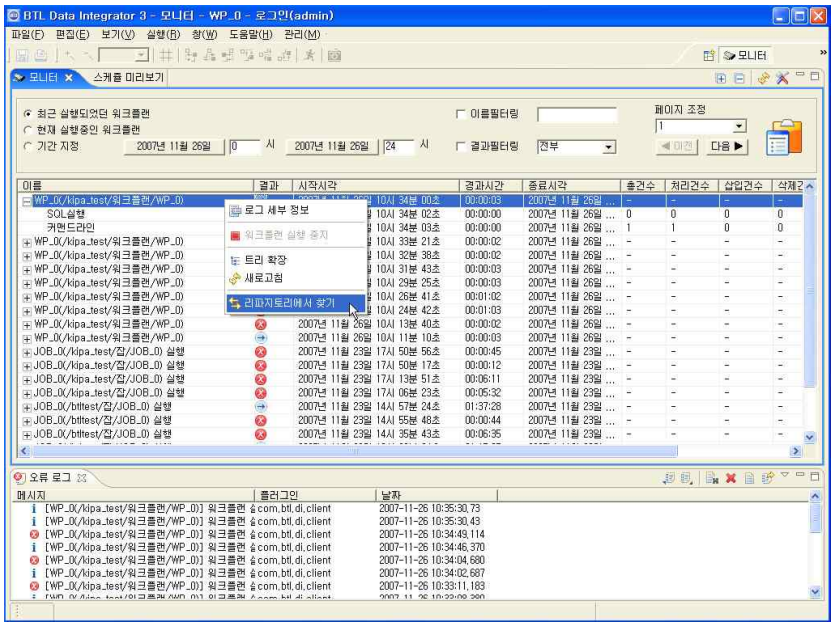


그림 48 디자이너 화면으로 이동

- 커맨드라인을 더블클릭하여 상세설정 다이얼로그를 연다. 실행할 스크립트명을 없는 파일로 지정한다. 고의로 에러를 발생시키는 것이다.

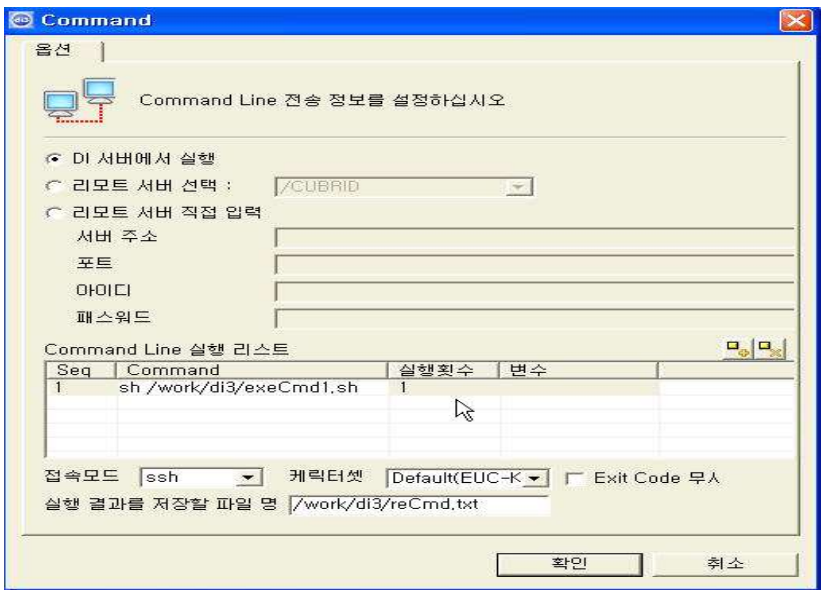
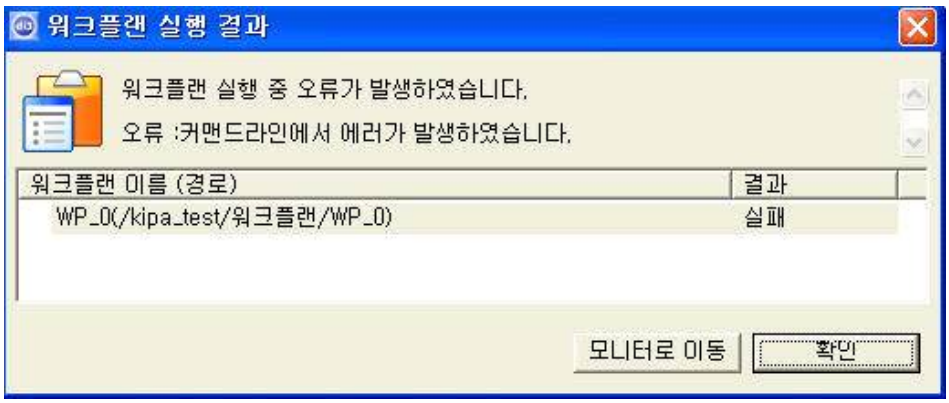
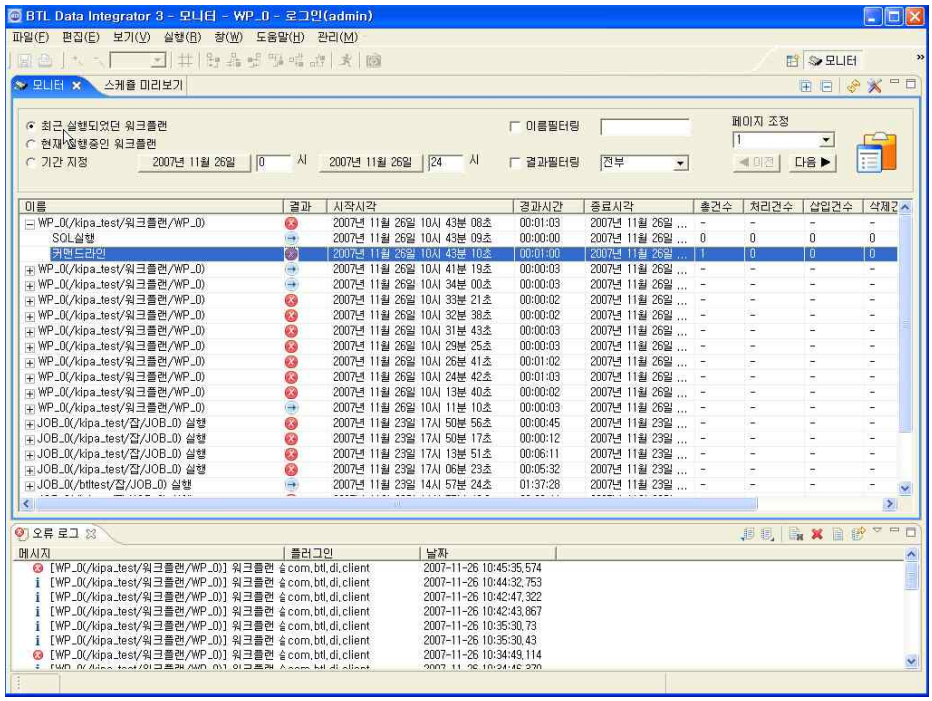
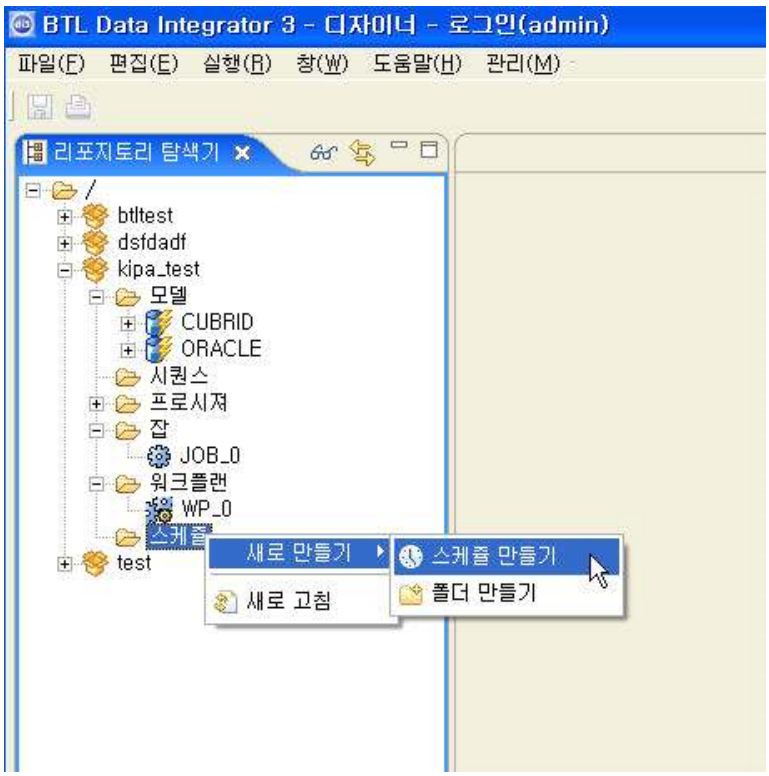
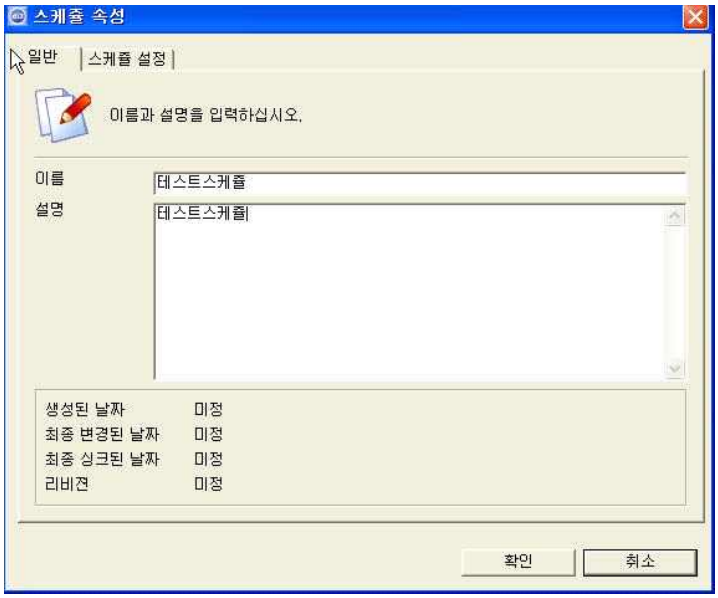
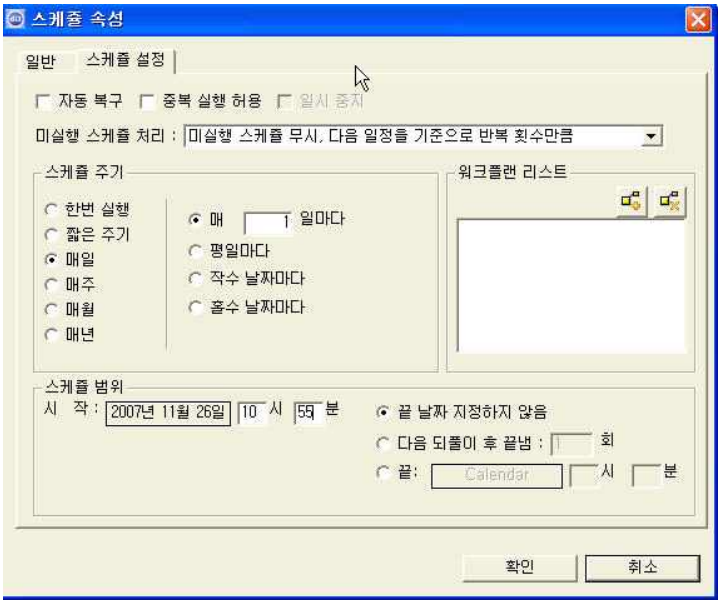
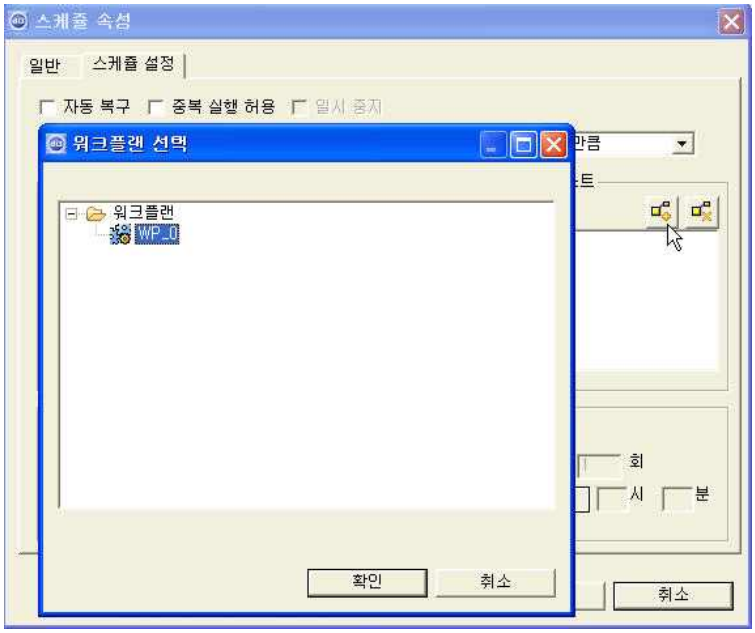
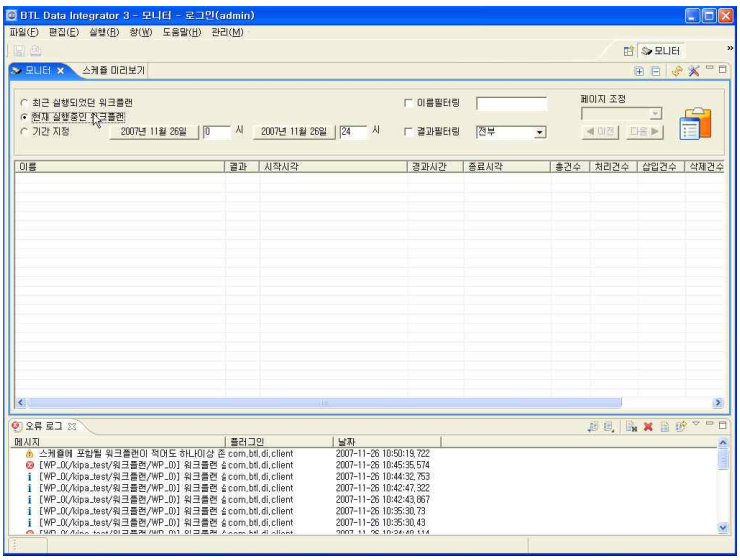


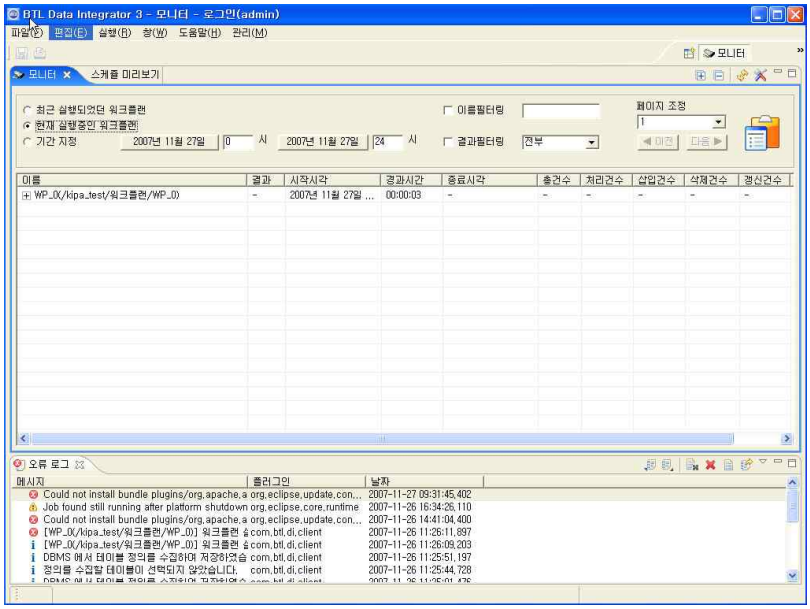
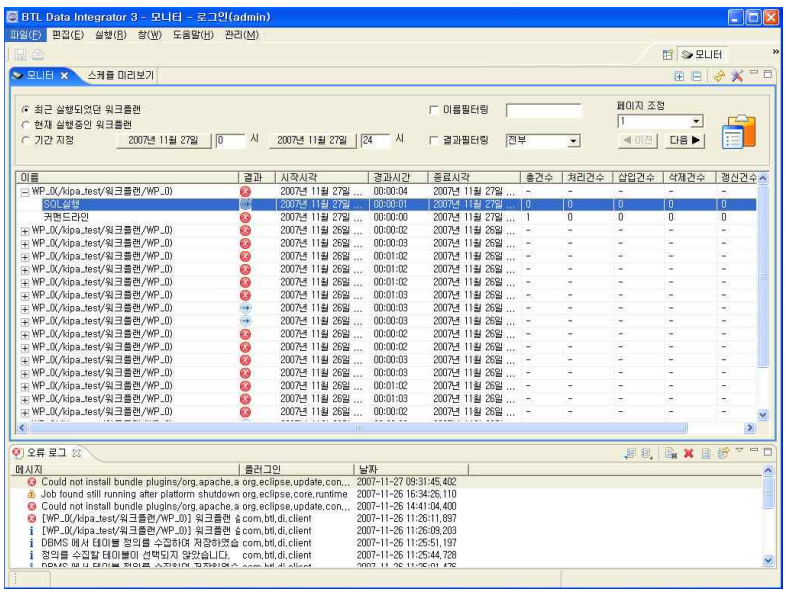
그림 49 고의로 에러 발생

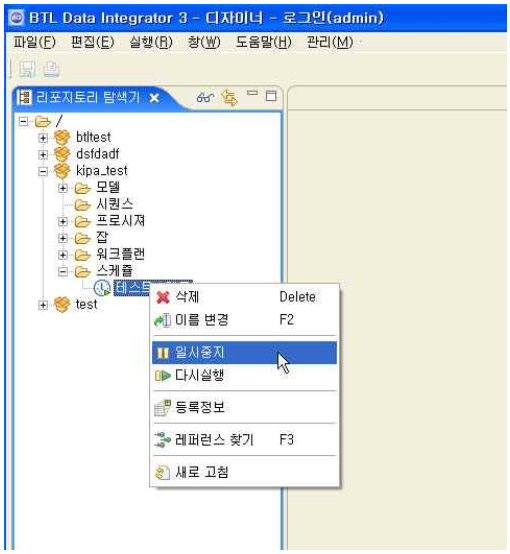
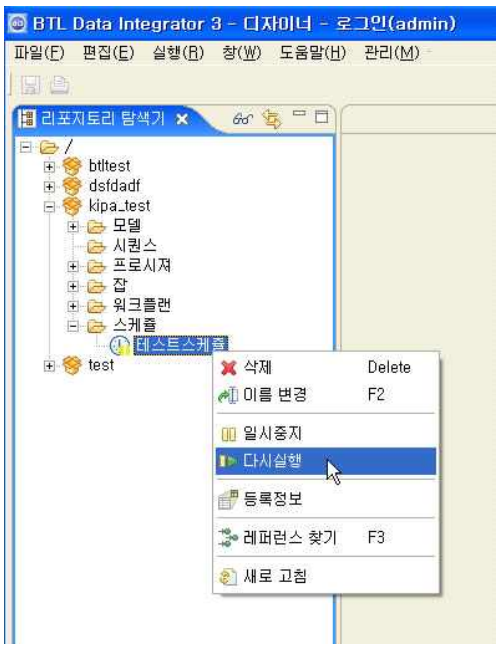
4		- 다시 한번 실행하여 실행결과를 확인한다.  그림 50 실행결과 확인 - 모니터상에서 작업 실패를 확인한다.  그림 51 작업 실패 확인
	비 고	

5	시험항목	스케쥴 등록
	시험절차	1. 스케쥴 생성 2. 스케쥴의 옵션 설정 3. 실행할 워크플랜 등록 4. 결과 확인 5. 스케쥴 일시중지
	시험결과	1. 스케쥴 생성 - 리포지토리탐색기의 스케쥴디렉토리를 우클릭하여 새로만들기->스케쥴만들기 메뉴를 선택한다.  그림 52 스케쥴 생성

5		2. 스케쥴의 옵션 설정 - 스케쥴속성 다이얼로그가 열린다. - 일반탭에서 이름과 설명을 입력한다.  그림 53 스케쥴 옵션 설정 - 스케쥴설정탭에서 스케쥴주기에 매일 -> 매1일마다 로 설정한다. 스케쥴범위에서 원하는 시간을 입력한다.  그림 54 스케쥴 범위 설정

5		3. 실행할 워크플랜 등록 - 워크플랜리스트의 추가버튼을 클릭하여 워크플랜을 등록한다. 여러개를 선택하면 선택된 워크플랜들이 모두 실행된다.  그림 55 실행할 워크플랜 등록 4. 결과확인 - 입력한 시각이 되면 모니터로 이동하여 현재실행중인워크플랜 라디오버튼을 클릭한다.  그림 56 결과확인

5		- 워크플랜이 시작되는지 확인한다. 우상단의 모니터리프레쉬 버튼을 클릭하여 경과상태를 확인한다.  그림 57 경과상태 확인 - 모니터에서 항목이 사라졌다면 최근 실행되었던 워크플랜으로 이동하여 결과물을 확인한다.  그림 58 결과물 확인

5		- 디자이너로 이동한 후 생성했던 스케쥴을 우클릭하여 일시정지 메뉴를 선택한다. 일시정지를 하면 매일로 지정했던 내용이 동작하지 않게 된다.  그림 59 스케쥴 일시정지 - 스케쥴아이콘의 모양이 일시정지 모양으로 바뀐다. 다시 우클릭하여 다시실행메뉴를 클릭하면 스케쥴이 지정했던 내용으로 다시 동작한다.  그림 60 스케쥴 다시실행
	비 고	