IBM® WebSphere® Commerce

Catalog Manager 사용자 안내서

버전 5.4

IBM® WebSphere® Commerce

Catalog Manager 사용자 안내서

버전 5.4

주!

이 책과 이 책이 지원하는 제품의 사용에 앞서, 129 페이지의 『주의사항』에 있는 일반 정보를 읽으십시오.

초판, 첫 번째 개정판(2002년 4월)

이 개정판은 다음 제품에 적용됩니다.

- IBM WebSphere Commerce, 버전 5.4(프로그램 5724-A18)

새 개정판에 별도로 명시하지 않는 한, 위 제품의 모든 후속 릴리스와 수정에 적용됩니다. 제품 레벨에 맞는 올바른 버전을 사용하고 있는지 확인하십시오.

한국 IBM 담당자나 IBM 지사를 통해 문의하십시오. 아래의 주소에는 서적이 비치되어 있지 않습니다.

IBM은 여러분의 의견을 환영합니다. 다음 중 한 가지 방법으로 의견을 보내실 수 있습니다.

1. 아래의 전자 우편 ID로 전자 우편을 보내십시오.

   ibmkspoe@kr.ibm.com

   응답을 원하는 경우 전체 네트워크 주소를 기입해 주십시오.

2. 우편으로 보내실 경우에는 다음 주소로 우송해 주십시오.

   135-270
   서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
   한국 아이.비.엠 주식회사
   고객만족센터

독자가 IBM에 정보를 보낼 경우, 그 정보가 타당하면 IBM은 적절한 방식으로 이를 사용하거나 배포할 수 있으며, 제공한 독자는 이에 대해 책임을 지거나 사용에 제한을 받지 않습니다.

목차

시작하기 전에

*IBM WebSphere Commerce Catalog Manager 사용자 안내서*는 WebSphere Commerce Catalog Manager에 대한 정보를 제공합니다. 특히, 다음 주제에 대한 자세한 내용을 제공합니다.

- Catalog Manager 도구 및 유틸리티를 사용하여 데이터 변환, 로드 및 추출
- Catalog Manager 웹 편집기를 사용하여 카탈로그 데이터 작업
- Catalog Manager 명령어

이 책에 사용된 규칙

이 책은 다음과 같은 규칙을 사용합니다.

굵은체	필드, 버튼 또는 메뉴 선택 사항의 이름과 같은 GUI(Graphical User Interface) 제어 또는 명령을 표시합니다.
모노체	파일 이름 및 디렉토리 경로와 같이 보이는 대로 정확하게 입력해야 하는 텍스트의 예를 표시합니다.
기울임꼴	사용자가 값을 대체해야 하는 변수 및 강조에 사용됩니다.
Windows	Window용 WebSphere Commerce 고유의 정보를 나타냅니다.
NT	Windows NT용 WebSphere Commerce 고유의 정보를 나타냅니다.
2000	Windows 2000용 WebSphere Commerce 고유의 정보를 나타냅니다.
AIX	AIX용 WebSphere Commerce 고유의 정보를 나타냅니다.
Solaris	Solaris Operating Environment software용 WebSphere Commerce 고유의 정보를 나타냅니다.
Linux	Linux용 WebSphere Commerce 고유의 정보를 나타냅니다.
400	IBM @server iSeries(이전에 AS/400으로 불림)용 WebSphere Commerce 고유의 정보를 나타냅니다.
DB2	DB2 Universal Database 고유의 정보를 나타냅니다.
Oracle	Oracle 데이터베이스 고유의 정보를 나타냅니다.
Business	WebSphere Commerce Business Edition 고유의 정보를 나타냅니다.
PRO	WebSphere Commerce Professional Edition 고유의 정보를 나타냅니다.

이 책의 사용자

WebSphere Commerce Catalog Manager의 기능을 이해하려는 회사 사용자와 사이트 개발자, 운영자 및 사용 방법을 배우려는 분들이 이 책을 읽어야 합니다.

특히 WebSphere Commerce Catalog Manager의 다양한 기능을 이해하고자 하는 WebSphere Commerce 상점 개발자 및 사이트 운영자는 이 책을 읽어야 합니다.

프로그램 확장을 수행 중인 WebSphere Commerce 상점 개발자 및 사이트 운영자는 다음 영역에 대한 지식이 있어야 합니다.

- 데이터베이스 기술
- Enterprise JavaBeans 구성요소 아키텍처
- HTML
- Java
- JavaServer Pages 기술
- VisualAge for Java, Enterprise Edition, 버전 3.5 이상
- XML

카탈로그, 상품 및 항목 요구사항과 상점 개발자의 통신 방법을 이해하기 위해 읽기 카테고리 및 상품 관리자는 이 안내서를 읽어야 합니다. 카테고리 및 상품 관리자는 상점의 카탈로그 갱신을 위한 웹 편집기 사용 방법 또한 이해해야 합니다.

추가 정보 위치

이 사용자 안내서는 WebSphere Commerce 웹 사이트에서 Adobe PDF로 사용 가능합니다. 이 문서의 최신 버전 및 WebSphere Commerce 관련 정보는 WebSphere Commerce 기술 라이브러리 웹 사이트를 확인하십시오.

- Business

  http://www.ibm.com/software/webservers/commerce/wc_be/lit-tech-general.html

- PRO

  http://www.ibm.com/software/webservers/commerce/wc_pe/lit-tech-general.html

제 1 부 Catalog Manager 개요

WebSphere Commerce Catalog Manager는 특정 카탈로그 관리상의 문제점을 해결하는 데 필요한 다양한 기능을 갖춘 일반 툴킷을 제공합니다. 이는 WebSphere Commerce 스키마에 작성되는 사용자 정의를 핸들할 만큼 충분히 유연합니다.

Catalog Manager는 복수 소스의 정보를 통합된 WebSphere Commerce 시스템으로 모으고 정보 관리의 표준 수단인 XML 파일을 사용하여 표준 카탈로그 및 상품 정의 포맷에 모든 다양한 데이터를 다시 맵핑할 수 있게 합니다.

Catalog Manager는 다음을 수행할 수 있는 방법을 제공합니다.

- 복수 입력 소스에서 WebSphere Commerce로 ASCII 및 XML 파일 양식으로 데이터 반입
- ASCII에서 XML 포맷으로 데이터 변환 및 다시 수행
- 하나의 XML 포맷에서 다른 포맷으로 데이터 다시 맵핑
- 복수 입력 스트림에서 하나로 결합된 데이터베이스로 데이터 결합
- 웹 브라우저 인터페이스를 통한 데이터 작성, 편집 및 삭제

Catalog Manager는 다음을 포함합니다.

- Catalog Manager 로더 패키지

  이 패키지는 주로 WebSphere Commerce 데이터베이스로 데이터를 준비 및 로드하는 명령 유틸리티로 구성됩니다. 로더 패키지를 사용하여 WebSphere Commerce 데이터베이스에서 많은 양의 데이터를 로드하고 데이터를 갱신할 수 있습니다.

  로더 패키지로 다음 또한 수행할 수 있습니다.

  – XML 문서처럼 데이터베이스에서 데이터 추출

  – 대체 XML 포맷으로 XML 데이터베이스 변환

  – 문자 분리 변수 포맷과 XML 데이터 포맷간의 데이터 변환

- Catalog Manager 관리 도구

  Catalog Manager는 해당 기능의 관리를 보조하는 사용자 인터페이스가 있는 다음 두 도구도 포함합니다.

  – 텍스트 변환 도구

  – XSL 편집기

- Catalog Manager 웹 편집기

  웹 편집기는 웹 브라우저를 통한 카탈로그 데이터의 작성, 삭제 및 변경을 가능하게 합니다.

제 2 부 데이터 변환, 로드 및 추출

제 1 장 소개

Catalog Manager 로더 패키지는 주로 WebSphere Commerce 데이터베이스로 데이터를 준비하고 로드하는 유틸리티로 구성됩니다. 로더 패키지를 사용하여 WebSphere Commerce 데이터베이스에서 많은 양의 데이터를 로드하고 데이터를 갱신할 수 있습니다. 로더 패키지로 다음 또한 수행할 수 있습니다.

- XML 문서처럼 데이터베이스에서 데이터 추출
- 대체 XML 포맷으로 XML 데이터베이스 변환
- 문자 분리 변수 포맷과 XML 데이터 포맷간의 데이터 변환

Catalog Manager는 해당 기능의 관리를 보조하는 사용자 인터페이스가 있는 다음 두 도구도 포함합니다.

- 텍스트 변환 도구
- XSL 편집기

유틸리티

WebSphere Commerce 로더 패키지는 WebSphere Commerce 데이터베이스로 데이터를 준비하고 로드하는 유틸리티를 포함합니다. 이 유틸리티를 사용하여 WebSphere Commerce 데이터베이스에서 많거나 적은 양의 데이터를 로드하고 갱신할 수 있습니다.

로드 중인 프로세스는 데이터를 WebSphere Commerce 데이터베이스로 이동하기 위해 필요한 단계로 구성됩니다.

1. DTD 작성기를 사용하여 DTD 생성
2. ID 분석기를 사용하여 입력 파일에서 식별자 분석
3. 로더를 사용하여 데이터 로드

로더 패키지는 XML 문서처럼 데이터베이스로부터 데이터를 추출하고 대체 XML 포맷으로 XML 데이터를 변환하는 유틸리티를 포함합니다.

로더 패키지는 다음 유틸리티로 구성됩니다.

- **텍스트 변환기**

  텍스트 변환기는 문자 분리 변수 포맷과 XML 데이터 포맷간의 데이터를 변환합니다.

  추가 정보는 9 페이지의 제 2 장 『텍스트 변환』을 참조하십시오.

- **XML 변환기**

XML 변환기는 XML 문서의 데이터를 변경, 수집 및 다시 맵핑해서 필요시 다른 사용자나 시스템이 사용할 XML 포맷을 대체합니다.

추가 정보는 17 페이지의 제 3 장『XML 데이터 변환』을 참조하십시오.

- **DTD 작성기**

  DTD는 XML 데이터 문서 내에서 사용할 수 있는 구조적 요소 및 마크업 정의를 지정합니다. 예를 들어 DTD는 문서에서 사용될 요소를 나열하고, 개별 요소가 있는 속성을 지정할 수 있습니다.

  DTD 작성기는 데이터베이스 스키마 기반의 로더용 DTD를 작성하는 유틸리티입니다. 이 DTD는 로더가 데이터를 반입하는 테이블 및 열을 설명합니다. DTD 작성기는 스키마 및 Catalog Manager 웹 편집기와 함께 사용 가능한 상세 XML 문서 또한 작성할 수 있습니다.

  DTD 작성기는 데이터에 적절한 대상 데이터베이스 기반의 DTD를 생성합니다. 이 DTD는 로드 프로세스에 사용됩니다. DTD 작성기를 한 번만 실행해야 합니다.

  추가 정보는 23 페이지의 제 4 장『DTD 및 스키마 생성』을 참조하십시오.

- **ID 분석기**

  ID 분석기는 XML 요소에 필요한 식별자를 생성하는 유틸리티입니다. XML 컨텐츠가 이미 식별자를 제공한 경우, ID 분석기를 실행할 필요가 없습니다.

  ID 분석기는 관련 식별자로 일련의 XML 요소를 갱신합니다. 로더 패키지 XML 파일이 바로 대상 데이터베이스 스키마에 맵핑하기 때문에 이 단계는 필수적입니다. 따라서 식별자가 있어야 합니다.

  ID 분석기는 오류 발생시 예외 문서를 생성하는 오류 보고자를 포함합니다.

  추가 정보는 29 페이지의 제 5 장『식별자 분석』을 참조하십시오.

- **로더**

  데이터베이스에 데이터를 로드하기 위한 입력으로 로더는 올바르고 체계화된 XML을 사용합니다. XML 문서의 요소는 데이터베이스의 테이블 이름에 맵핑하고 요소 속성은 열에 맵핑합니다. 로더는 시스템에 데이터를 로드하는 가장 일반적인 방법입니다.

  로더는 테이블에 열 레벨 갱신을 허용합니다. 데이터베이스에서 데이터 삭제 또한 허용합니다.

  다음은 로더에 입력한 체계화되고 올바른 XML의 예입니다.

```
<ADDRBOOK
  ADDRBOOK_ID="11801"
  MEMBER_ID="100"
  DISPLAYNAME="Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
```

위 예에서 ADDRBOOK은 테이블 이름이며 갱신될 열은 ADDRBOOK 요소의 속성으로 표시됩니다.

로더 유틸리티는 다음 특징을 포함합니다.

- **오류 보고자**

  로더는 오류 발생시 예외 문서를 생성하는 오류 보고자를 포함합니다.

- **상품 어드바이저 공간 검색 동기화**

  WebSphere Commerce 카탈로그에서 정보를 추출하고 검색에 적합한 양식으로 정보를 표시해서 상품 어드바이저 공간 검색을 작성합니다. 카탈로그가 갱신되면, 공간 검색과 카탈로그는 비동기화됩니다. 카탈로그로 공간 검색을 동기화하는 과정에서 지연을 피하려면, 로더의 상품 어드바이저 공간 검색 동기화 기능을 사용으로 설정합니다.

  상품 어드바이저 공간 검색 동기화를 사용으로 설정할 경우, 로더는 다음 테이블에 표시된 대로 스케줄러 작업 테이블 SCHCONFIG와 스케줄러 작업 인스턴스 SCHSTATUS에 명령 정보를 추가해서 적절한 공간 검색 명령을 계획합니다.

테이블	모드	공간 검색 명령	조건
CATENTRY	갱신	UpdateSearchSpaces	공개 플래그 갱신
	삭제	RemoveProductFromAllSearchSpaces	상품 삭제
CATENTDESC	추가	UpdateSearchSpaces	설명/언어 추가
	갱신	UpdateSearchSpaces	공개 플래그 갱신
			설명 데이터 갱신
	삭제	UpdateSearchSpaces	설명/언어 제거
LISTPRICE	추가	UpdateSearchSpaces	원가 추가
	갱신	UpdateSearchSpaces	원가 갱신
	삭제	UpdateSearchSpaces	원가 삭제
ATTRVALUE	추가	UpdateSearchSpaces	사용자 정의 값 추가
	갱신	UpdateSearchSpaces	사용자 정의 값 갱신
	삭제	UpdateSearchSpaces	사용자 정의 값 삭제
CATENTATTR	추가	UpdateSearchSpaces	속성 추가
	갱신	UpdateSearchSpaces	속성 갱신
	삭제	UpdateSearchSpaces	속성 삭제
CATGPENREL	추가	AddProductsToSearchSpace	카테고리에 상품 추가
	삭제	RemoveProductsFromSearchSpace	카테고리에서 상품 제거

추가 정보는 45 페이지의 제 6 장 『데이터 로드』를 참조하십시오.

- **추출기**

  추출기는 데이터베이스에서 XML 문서로 데이터의 선택된 서브세트를 추출하는 데이터베이스에 반하는 조회를 사용합니다. 데이터베이스에서 추출된 데이터는 추출 필터 XML 문서를 사용하여 지정됩니다.

추출기의 기능은 로더의 기능과 상반됩니다. 추출기를 사용하여 WebSphere Commerce 데이터베이스에서 데이터의 선택적인 서브세트를 XML 파일 양식으로 추출할 수 있습니다. 예를 들어 다가올 휴일 관련 상품에 대한 데이터를 추출하거나 다른 시스템을 사용하기 위해 통합 데이터베이스에서 정보를 추출할 수 있습니다.

추가 정보는 65 페이지의 제 7 장 『데이터 추출』을 참조하십시오.

로더 패키지는 로거(logger) 기능도 제공합니다. 로더 패키지의 개별 유틸리티는 프로그램 추적 정보를 제공하는 메시지 뿐 아니라 성공, 실패 및 오류를 표시하는 메시지를 작성합니다.

관리 도구

Catalog Manager는 기능 관리를 보조하는 다음 도구를 포함합니다.

텍스트 변환 도구

텍스트 변환 도구는 운영자가 문자 분리 변수 포맷과 XML 데이터 포맷간의 데이터 변환을 처리하는 데 필요한 정보를 준비하도록 도와줍니다.

XSL 편집기

XML 변환기는 XSL(Extensible Stylesheet Language)을 사용해서 하나의 XML 파일에 다른 XML 파일을 변환하는 규칙을 정의합니다. XSL 편집기의 맵핑 기능은 소스 DTD(Document Type Definition)의 요소를 대상 DTD의 요소로 연관시키는 비주얼 인터페이스를 운영자에게 제공합니다.

제 2 장 텍스트 변환

Catalog Manager는 ASCII 파일 출력을 다른 도구(예: 스프레드시트 프로그램)에서 WebSphere Commerce 데이터베이스에 입력 가능한 XML 데이터 포맷으로 전환할 수 있게 합니다.

텍스트 변환 도구는 문자 분리 변수 포맷과 XML 데이터 포맷간의 데이터 변환을 처리하는 데 필요한 정보를 준비합니다. 다음 보기가 제공됩니다.

1. 텍스트 스키마 편집 보기는 변환에 사용되도록 XML 스키마 파일을 작성하고 수정하게 합니다.
2. 변환 명령 편집 보기는 변환 처리를 실행할 때 사용하는 실제 명령을 작성하고 수정하게 합니다.
3. 변환 명령 처리 보기는 변환 과정을 실행하게 합니다.

텍스트 변환 도구 실행

텍스트 변환 도구를 실행하려면 WebSphere Commerce 디렉토리에 제공된 적절한 스크립트나 명령을 사용합니다.

- NT *drive*:\WebSphere\CommerceServer\bin\TextTrans.cmd
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\bin\TextTrans.cmd
- AIX /usr/WebSphere/CommerceServer/bin/texttrans.sh
- Solaris Linux /opt/WebSphere/CommerceServer/bin/texttrans.sh

400 iSeries 환경에서 Windows NT 또는 Windows 2000 시스템에 텍스트 변환 도구를 실행하려면 먼저 필요한 파일을 복사해야 합니다. 다음은 사용 방법에 관한 예입니다.

1. Windows 시스템에 새 디렉토리를 작성하십시오(예: *drive*:\TextTrans).
2. *drive*:\TextTrans 아래에 다음 새 서브디렉토리를 작성하십시오.

   ```
   \bin
   \lib\loader
   \wcsadmin
   ```

3. 다음 도표에 따라 iSeries 시스템에서 아래 디렉토리로 파일을 복사하십시오.

   ```
   From                                  To
     /QIBM/ProdData/WebCommerce            drive:\TextTrans
       /bin                                  \bin
       /TextTrans.cmd          -------->     \TextTrans.cmd
       /lib/loader                           \lib\loader
         /wcmxmlp.jar          -------->       \wcmxmlp.jar
         /wcmxslt.jar          -------->       \wcmxslt.jar
       /wcsadmin                             \wcsadmin
         /acsxml.ico           -------->       \acsxml.ico
         /swing.jar            -------->       \swing.jar
         /TextTransformerUI.zip -------->      \TextTransformerUI.zip
         /TextTransform.cnt    -------->       \TextTransform.cnt
         /TextTransform.hlp    -------->       \TextTransform.hlp
         /TextTransform.reg    -------->       \TextTransform.reg
   ```

4. `set lib=%WCS_HOME%\lib\loader`

   를 수행하기 전에

   `set WCS_HOME=`*drive*`:\TextTrans`

   를 삽입하여 Windows 시스템에서 TextTrans.cmd 파일을 수정하십시오.

5. Windows 시스템에서 TextTrans.cmd를 실행하여 텍스트 변환 도구를 실행하십시오.

텍스트 스키마 편집 보기 사용

다음은 문자 분리 변수 포맷과 XML 포맷 간의 데이터 변환을 위한 XML 스키마 작성 및 수정과 관련된 프로시저입니다.

스키마 파일 작성

새 스키마 파일을 작성하려면 다음을 수행하십시오.

1. 텍스트 변환 도구를 실행하십시오.
2. **파일 > 새로 만들기**를 선택하거나 도구 모음에서 새 아이콘을 누르십시오.
3. 작성할 XML 스키마 파일의 경로를 선택하고 이름을 입력하십시오.

   주: 기본 파일 이름은 ″Default.xml″입니다.
4. **CSV 포맷**을 선택하십시오.

   주: WebSphere Commerce Suite 포맷을 선택하지 마십시오. 이 옵션은 WebSphere Commerce의 이전 서버에서만 사용합니다.
5. 완료되면 **확인**을 누르십시오.

새 요소와 속성을 작성하려면 아래 표시된 단계를 수행하십시오.

스키마 파일 열기

스키마 파일을 열려면 다음을 수행하십시오.

1. 텍스트 변환 도구를 실행하십시오.
2. **파일 > 열기**를 선택하거나 도구 모음에서 새 아이콘을 누르십시오.
3. 스키마 파일을 선택해 여십시오.
4. 완료되면 **확인**을 누르십시오.

요소와 속성을 수정하려면 아래 설명된 단계를 수행하십시오.

스키마 파일 저장

스키마 파일을 저장하려면 다음을 수행하십시오.

1. 텍스트 변환 도구를 실행하십시오.
2. 스키마에 작성한 모든 변경을 저장하려면 **파일 > 저장**을 선택하거나 도구 모음의 저장 아이콘을 누르십시오.
3. 새 이름으로 스키마의 사본을 저장하려면 다음을 수행하십시오.
   a. **파일 > 다른 이름으로 저장**을 선택하십시오.
   b. 작성할 XML 파일의 경로를 선택하고 이름을 입력하십시오.
   c. 완료되면 **확인**을 누르십시오.

스키마 파일 편집

스키마 파일을 편집하려면, 위에서 설명한 대로 스키마 파일을 열고 다음 절차를 수행하십시오.

요소 추가

요소를 추가하려면 다음을 수행하십시오.

1. 요소 목록 필드에서 새 요소 이름을 입력하십시오.
2. 행 추가 아이콘을 누르십시오.

요소 제거

요소를 제거하려면 다음을 수행하십시오.

1. 요소 이름을 선택하십시오.
2. 행 제거 아이콘을 누르십시오.

요소 바꾸기

요소를 바꾸려면 다음을 수행하십시오.

1. 요소 목록 필드에서 새 요소 이름을 입력하십시오.
2. 바꿀 요소의 이름을 선택하십시오.
3. 행 바꾸기 아이콘을 누르십시오.

한 행 위로 요소 이동

한 행 위로 요소를 이동하려면 다음을 수행하십시오.

1. 위로 이동될 요소의 이름을 선택하십시오.
2. 위로 행 이동 아이콘을 누르십시오.

한 행 아래로 요소 이동

한 행 아래로 요소를 이동하려면 다음을 수행하십시오.

1. 아래로 이동될 요소의 이름을 선택하십시오.
2. 아래로 행 이동 아이콘을 누르십시오.

속성 추가

속성을 추가하려면 다음을 수행하십시오.

1. 속성 목록 필드에 새 속성 이름을 입력하십시오.
2. 행 추가 아이콘을 누르십시오.

속성 제거

속성을 제거하려면 다음을 수행하십시오.

1. 속성 이름을 선택하십시오.
2. 행 제거 아이콘을 누르십시오.

속성 바꾸기

속성을 바꾸려면 다음을 수행하십시오.

1. 속성 목록 필드에 새 속성 이름을 입력하십시오.
2. 바꾸기할 속성의 이름을 선택하십시오.
3. 행 바꾸기 아이콘을 누르십시오.

한 행 위로 속성 이동

한 행 위로 속성을 이동하려면 다음을 수행하십시오.

1. 위로 이동될 속성의 이름을 선택하십시오.
2. 위로 행 이동 아이콘을 누르십시오.

한 행 아래로 속성 이동

한 행 아래로 속성을 이동하려면 다음을 수행하십시오.

1. 아래로 이동될 속성의 이름을 선택하십시오.
2. 아래로 행 이동 아이콘을 누르십시오.

스키마 파일 구조 변경

텍스트 스키마 편집 보기 아래 분할창의 파일 구조 보기는 문자 분리 변수 파일의 레이아웃을 설명합니다. 다음 필드는 예상 파일 구조에 필요합니다.

필드 분리자

속성값을 분리하는 분리문자를 지정하십시오. 기본값은 쉼표(",")입니다.

레코드 분리자

데이터 레코드를 분리하는 분리문자를 지정하십시오. 기본값은 "

" 입니다(이는 \r\n에 대한 엔티티 참조와 동일합니다).

문자열 분리자

문자열 시작 및 종료 경계를 표시하는 분리문자를 지정하십시오. 기본값은 인용구(" , ")입니다.

머리글 포함

부울값은 텍스트 데이터 파일에 머리글 행이 있으면 "true"로 머리글 행이 없으면 "false"로 지정됩니다. 머리글이 있는 경우, 머리글이 XML 태그 이름으로 사용되기 때문에 이는 태그 이름에 대한 XML 규칙에 적합해야 합니다. 기본값은 "false"입니다.

머리글 행 번호

텍스트 데이터 파일에 존재하는 머리글 행 번호를 지정하십시오. 기본값은 0("0") 입니다.

XML에서 문자 분리 변수 포맷으로 데이터를 변환하도록 스키마 파일 준비

텍스트 스키마 편집 보기를 사용하여 작성한 XML 스키마 파일을 사용하여 XML 포맷에서 문자 분리 변수 포맷으로 데이터를 변환하려면 변환을 처리하기 전에 텍스트 편집기를 사용하여 스키마 파일의 데이터 유형 스펙을 ″CSV 포맷″에서 ″XML 포맷″으로 변경해야 합니다.

변환 명령 편집 보기 사용

변환 명령 편집 보기를 사용하여 새 명령 파일 작성, 기존 명령 파일 열기 또는 명령 파일에 변경사항을 저장할 수 있습니다. 기본 명령 파일은 ″Manifest.txt″입니다.

새 명령 작성, 명령 제거, 편집된 정보로 명령 바꾸기 및 명령 순서 변경을 작성할 수 있습니다.

주: 명령 파일은 명령 테이블이 갱신될 때마다 자동으로 저장됩니다.

명령 작성

새 명령을 작성하려면 다음을 수행하십시오.

1. 문자 분리 변수 포맷 파일(.csv 확장자) 또는 XML 포맷 파일(.xml 확장자)로 소스 파일을 지정하십시오.
2. XML 스키마 파일이 변환에 사용되도록 지정하십시오.
3. XML 포맷 파일(.xml 확장자) 또는 문자 분리 변수 포맷 파일(.csv 확장자)로 변환 처리 중 작성 또는 수정될 출력 파일(새 파일이 저장될 위치)의 이름을 지정하십시오.
4. 명령 모드를 지정하십시오.

   출력 파일을 작성할 경우 **작성**을, 출력 데이터를 기존 데이터 파일에 추가할 경우 **추가**를 선택하십시오.
5. 행 추가 아이콘을 누르십시오.

명령 제거

명령을 제거하려면 다음을 수행하십시오.

1. 명령을 선택하십시오.
2. 행 제거 아이콘을 누르십시오.

명령 바꾸기 또는 편집

명령을 편집하거나 바꾸려면 다음을 수행하십시오.

1. 명령 편집 아이콘을 누르십시오.

   행 데이터가 적절한 입력 필드에 작성됩니다.
2. 해당 입력 필드에 텍스트를 변경하십시오.

3. 행을 갱신하려면 행 바꾸기 아이콘을 누르십시오.

한 행 위로 명령 이동

한 행 위로 명령을 이동하려면 다음을 수행하십시오.

1. 명령을 선택하십시오.
2. 위로 행 이동 아이콘을 누르십시오.

주: 변환 처리 순서가 변경됩니다.

한 행 아래로 명령 이동

한 행 아래로 명령을 이동하려면 다음을 수행하십시오.

1. 명령을 선택하십시오.
2. 아래로 행 이동 아이콘을 누르십시오.

주: 변환 처리 순서가 변경됩니다.

명령 지우기

명령을 지우려면 다음을 수행하십시오.

1. 명령을 선택하십시오.
2. 입력 필드 지우기 아이콘을 누르십시오.

   소스 파일 필드, 스키마 파일 필드 및 출력 파일 필드에서 텍스트를 지우십시오.

변환 처리 보기 사용

텍스트 변환 처리를 실행하려면 다음을 수행하십시오.

1. 파일 필드에서 매개변수 파일의 이름을 입력하거나 찾아보기합니다.
2. **처리**를 누르십시오.

처리 버튼 아래의 출력 영역은 변환 처리 상태에 대한 정보를 표시합니다. 텍스트 영역 아래의 저장 버튼을 눌러 출력 정보를 저장하거나 지우기 버튼을 눌러 모든 상태 정보를 지울 수 있습니다.

제 3 장 XML 데이터 변환

XSL은 다음을 제공합니다.

1. XML 문서에 포맷팅 지정을 위한 언어
2. XML 파일을 다른 일반 구조 파일에서 변환하는 방법을 설명하는 언어

   XSL의 변환 성능은 XML 파일을 다른 XML 스키마나 DTD에 적합한 다른 XML 파일로 변환하는 데 사용될 수 있습니다.

XML 파일을 대체 XML 포맷으로 변환하려면 XSL 변환 규칙 파일을 사용하여 변환 규칙을 지정해야 합니다.

다음은 자국어로 일본어를 사용하는 XSL 변환 규칙 파일로 `MemberSubsystem.xsl`을 사용하여 `MemberSubsystemExtracted.xml`에서 데이터를 변환하는 예입니다.

- NT 2000 AIX Solaris Linux

```
java com.ibm.wca.XMLTransformer.XMLTransformer -infile MemberSubsystemExtracted.xml
-transform MemberSubsystem.xsl -outfile TransMbrStr.xml -param 'language="-10"'
```

- 400

```
QWEBCOMM/TRNWCSXML INFILE(MemberSubsystemExtracted.xml)
TRANSFORM(MemberSubsystem.xsl) INSTROOT(/QIBM/UserData/WebCommerce/instances/my_inst)
OUTFILE(TransMbrStr.xml) PARAM('language=-10')
```

XML 변환기는 XSL을 사용해서 XML 파일을 다른 XML파일로 변환하는 규칙을 정의합니다. XSL 편집기에서 맵핑 기능은 소스 DTD의 요소를 대상 DTD의 요소로 연관시키는 비주얼 인터페이스를 제공합니다. 주어진 두 개의 DTD로, 첫 번째(소스) DTD에 적합한 XML 파일이 두 번째(대상) DTD에 적합한 파일로 변환되는 방법을 판별하는 XSL 규칙을 개발할 수 있습니다.

XSL 편집기 실행

XSL 편집기를 실행하려면 WebSphere Commerce 디렉토리에 제공된 적절한 스크립트나 명령을 사용하십시오.

- NT *drive*:\WebSphere\CommerceServer\bin\XSLEditor.cmd
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\bin\XSLEditor.cmd
- AIX /usr/WebSphere/CommerceServer/bin/xsleditor.sh
- Solaris Linux /opt/WebSphere/CommerceServer/bin/xsleditor.sh

400 iSeries 환경에서 Windows NT 또는 Windows 2000 시스템에 XML 편집기를 실행하려면 먼저 필요한 파일을 복사해야 합니다. 다음은 사용 방법에 관한 예입니다.

1. Windows 시스템에 새 디렉토리를 작성하십시오(예: *drive*:\XMLTrans).
2. *drive*:\TextTrans 아래에 다음 새 서브디렉토리를 작성하십시오.

```
\bin
\lib\loader
\wcsadmin
```

3. 다음 도표에 따라 iSeries 시스템에서 아래 디렉토리로 파일을 복사하십시오.

```
From                                    To
  /QIBM/ProdData/WebCommerce              drive:\XMLTrans
    /bin                                    \bin
    /XSLEditor.cmd          -------->       \XSLEditor.cmd
    /lib/loader                             \lib\loader
      /wcmxmlp.jar          -------->         \wcmxmlp.jar
      /wcmxslt.jar          -------->         \wcmxslt.jar
    /wcsadmin                               \wcsadmin
      /config.dtd           -------->         \acsxml.ico
      /swing.jar            -------->         \swing.jar
      /XML_transform.ico    -------->         \XML_transform.ico
      /XMLTransformerUI.zip -------->         \XMLTransformerUI.zip
      /XMLTransformWP.xml   -------->         \XMLTransformWP.xml
      /XMLTransformWP.dtd   -------->         \XMLTransformWP.dtd
      /XMLRuleConfig.xml    -------->         \XMLRuleConfig.xml
      /XMLTransform.cnt     -------->         \XMLTransform.cnt
      /XMLTransform.hlp     -------->         \XMLTransform.hlp
      /XMLTransform.reg     -------->         \XMLTransform.reg
```

4. `set lib=%WCS_HOME%\lib\loader`

   를 수행하기 전에

   `set WCS_HOME=`*drive*`:\XMLTrans`

   를 삽입하여 Windows 시스템에서 XSLEditor.cmd 파일을 수정하십시오.

5. Windows 시스템에서 XSLEditor.cmd를 실행하여 XML 변환 도구를 실행하십시오.

맵핑 규칙 빌딩 영역에서 작업

XSL 편집기를 실행하면 맵핑 규칙 빌딩 영역 창이 표시됩니다. 이 창을 사용해서 맵핑 규칙 빌딩 영역을 관리하십시오.

맵핑 규칙 작성 빌딩 영역

맵핑 규칙 빌딩 영역을 작성하려면 다음을 수행하십시오.

1. XSL 편집기를 실행하십시오.
2. 드롭 다운 메뉴에서 **[새로 만들기]**를 선택하십시오.
3. 이름 필드에 새 맵핑 규칙 빌딩 영역의 이름을 입력하십시오.
4. 설명 필드에 새 맵핑 규칙 빌딩 영역의 간단한 설명을 입력하십시오.
5. 소스 스키마 필드에 기존 파일의 이름을 입력하거나 소스 스키마로 사용되는 기존 파일에서 탐색하십시오.
6. 대상 스키마 필드에 기존 파일의 이름을 입력하거나 대상 스키마로 사용되는 기존 파일에서 탐색하십시오.
7. XSL 규칙 파일 필드에 작성될 새 규칙 파일의 이름을 입력하십시오.

   여기에 전체 경로를 지정할 수 있습니다. 경로를 지정하지 않을 경우, 파일은 현재 진행 중인 작업 디렉토리에 작성됩니다.
8. **열기**를 눌러 새 맵핑 규칙 빌딩 영역을 작성하고 여십시오.

맵핑 규칙 빌딩 영역 수정

맵핑 규칙 빌딩 영역을 수정하려면 다음을 수행하십시오.

1. XSL 편집기를 실행하십시오.
2. 드롭 다운 메뉴에서 수정하려는 맵핑 규칙 빌딩 영역을 선택하십시오.
3. **열기**를 눌러 맵핑 규칙 빌딩 영역을 여십시오.
4. 변경하려는 필드를 갱신하십시오.
5. **저장**을 눌러 변경사항을 저장하십시오.

맵핑 규칙 빌딩 영역 삭제

맵핑 규칙을 삭제하려면 다음을 수행하십시오.

1. XSL 편집기를 실행하십시오.
2. 드롭 다운 메뉴에서 삭제하려는 맵핑 규칙 빌딩 영역을 선택하십시오.
3. 제거 버튼을 눌러 항목을 제거하십시오.

주: 맵핑 규칙 빌딩 영역을 제거해도 디스크에서 실제 파일을 삭제하지 않습니다.

XSL 편집기 사용

XSL 편집기를 사용하여 맵핑 규칙 빌딩 영역을 열면, 맵핑 규칙 빌딩 영역이 XSL 편집기 기본 창에 표시됩니다.

XSL 편집기 기본 창에서 왼쪽 분할창은 "소스 스키마"로 레이블이 붙은 소스 DTD의 계층 구조 보기를 표시합니다. 오른쪽 분할창은 "대상 스키마"로 레이블이 붙은 대상 DTD의 계층 구조 보기를 표시합니다.

맵핑 규칙 작성

요소 대 요소 맵핑

소스 계층에서 요소를 선택하고 끌어오기하여 대상 계층의 요소에 놓으십시오. XSL 규칙이 생성되면 창의 맨 아래에 위치한 맵핑 규칙 보기에 표시됩니다.

다음은 생성된 XSL 규칙의 예입니다.

```
<xsl:template match="merchant">
  <xsl:element name="MERCHANT">
  </xsl:element>
</xsl:template>
```

주: 기존에 존재하지 않지만 필요한 관계가 자동으로 생성됩니다.

속성 대 속성 맵핑

소스 계층에서 속성을 선택하고 끌어오기하여 대상 계층의 속성에 놓으십시오. XSL 규칙이 생성되면 창의 맨 아래에 위치한 맵핑 규칙 보기에 표시됩니다.

다음은 생성된 XSL 규칙의 예입니다.

```
<xsl:attribute name="MEADDR1">
  <xsl:apply-templates select="@mecmail1"/>
</xsl:attribute>
```

주: 기존에 존재하지 않지만 필요한 관계가 자동으로 생성됩니다.

사용자 정의 맵핑 표현식 작성

사용자 정의 맵핑 표현식을 작성하려면 먼저 대상 계층에서 요소 또는 속성을 선택하십시오. 그리고 나서 마우스 오른쪽 버튼을 눌러 사용자 정의 표현식 작성 메뉴를 선택하십시오. 사용자 정의 표현식 작성 창이 두 개의 드롭 다운 메뉴에서 사용 가능한 템플리트 및 규칙 표현식의 목록을 표시합니다. 다음을 수행하여 사용자 정의 표현식을 완료하십시오.

1. 사용자 정의 표현식이 추가될 템플리트를 선택하십시오.

2. 작성될 규칙 표현식을 선택하십시오(예: **Constant Expression**).
3. 테이블에 나열된 각각의 매개변수 값에 대한 열 값을 입력하고 값을 확약하려면 **Enter**를 누르십시오.
4. **확인**을 눌러 작성 단계를 완료하거나 **취소**를 눌러 규칙 작성을 취소하십시오.

생성된 XSL 규칙은 규칙 구성 파일(XSLRuleConfig.xml)에 정의된 사용자 정의 표현식에 기초합니다. 필요한 경우 규칙 구성 파일을 수정할 수 있으며, 새 규칙을 추가할 수 있습니다. 규칙 표현식 목록에서 규칙을 사용 가능하게 하려면, 규칙이 ″true″가 되도록 Visibility 속성을 설정하십시오.

맵핑 규칙 삭제

맵핑 규칙을 삭제하려면 다음을 수행하십시오.

1. 맵핑 규칙 테이블에서 규칙을 선택하십시오.
2. 마우스 오른쪽 버튼을 눌러 **삭제**를 선택하십시오.

   규칙과 모든 규칙의 종속자는 삭제됩니다.

   주: 갱신된 맵핑 규칙 및 생성된 XSL 규칙은 자동적으로 지속됩니다.

XML 변환 처리

XML 변환을 처리하려면 다음을 수행하십시오.

1. 변환 처리 창을 불러오려면 **도구 > 변환**을 선택하십시오.
2. 필수 필드를 완료하십시오.
   a. 입력 XML 파일 필드에 소스 XML 데이터 파일의 이름과 경로를 입력하거나 찾아보기하십시오.
   b. XSL 규칙 파일 필드에 변환에 사용될 맵핑 규칙 파일의 이름과 경로를 입력하거나 찾아보기하십시오.

      맵핑 규칙 빌딩 영역이 열린 경우, 이 필드에는 맵핑 규칙 빌딩 영역에서 현재 열린 맵핑 규칙 파일의 경로가 사전 기입됩니다.
   c. 출력 XML 파일 필드에 변환 처리 중 작성된 새 XML 데이터 파일의 이름과 경로를 입력하거나 찾아보기하십시오.
3. **시작**을 눌러 XML 변환 처리를 시작하거나 **종료**를 눌러 창을 종료하십시오.

맵핑 규칙 테이블 사용자 정의

맵핑 규칙 테이블을 사용자 정의하려면 다음을 수행하십시오.

1. 테이블 열을 숨기려면 테이블의 셀을 마우스 오른쪽 버튼으로 누른 다음 **열 숨기기**를 선택하십시오.
2. 테이블에 숨겨진 열을 표시하려면 다음을 수행하십시오.
   a. 테이블에서 셀을 마우스 오른쪽 버튼으로 누르십시오.

b. 숨겨진 열의 목록을 불러오려면 **열 표시**를 선택하십시오.

c. 목록에서 열을 선택하십시오.

주: 복수 열을 선택하려면 **Shift**를 누른 상태로 열 이름을 누르십시오.

d. **확인**을 눌러 선택된 열을 표시하거나 **취소**를 눌러 취소하십시오.

3. 테이블의 모든 숨겨진 열을 표시하려면 테이블에서 셀을 마우스 오른쪽 버튼으로 누르고 **모든 열 표시**를 선택하십시오.

   모든 열은 기본 순서로 표시됩니다.

완료된 XSL 규칙/값 표현식 표시

값 표현식이나 XSL 규칙 열에서 셀을 눌러 선택된 행에 완료된 규칙 컨텐츠를 표시하는 창을 불러옵니다.

제 4 장 DTD 및 스키마 생성

DTD 작성기는 로더 패키지를 사용하여 DTD와 스키마를 작성할 수 있습니다. DTD 작성기는 데이터베이스 테이블 이름에 있는 입력 파일을 사용하고 DTD 생성 명령을 호출하는 방법에 따라, DTD를 생성하거나 데이터베이스를 기술하는 상세 XML 파일에 있는 스키마 및 DTD를 생성합니다.

DTD 작성기 설정

로더 DTD는 WebSphere Commerce 데이터베이스 스키마에 직접 맵핑합니다. 개별 테이블은 요소이고 개별 열은 속성입니다.

예: 데이터베이스 스키마에 로더 DTD 맵핑

CATENTRY 테이블에 대한 DDL 명령문	**DTD**
CREATE TABLE "CATENTRY" ("CATENTRY_ID" BIGINT NOT NULL, "MEMBER_ID" BIGINT NOT NULL, "CATENTTYPE_ID" CHAR(16) NOT NULL, "MARKFORDELETE" INTEGER NOT NULL, "PARTNUMBER" VARCHAR(64) NOT NULL, "MFPARTNUMBER" VARCHAR(64) , "MFNAME" VARCHAR(64) , "URL" VARCHAR(254) , "FIELD1" INTEGER , "FIELD2" INTEGER , "FIELD3" DECIMAL(20,5) , "FIELD4" VARCHAR(254) , "FIELD5" VARCHAR(254) , "LASTUPDATE" TIMESTAMP , "OID" VARCHAR(64) , "ONSPECIAL" INTEGER , "ONAUCTION" INTEGER , "BUYABLE" INTEGER , "BASEITEM_ID" INTEGER , "CLASSIFGRP_ID" INTEGER , "ITEMSPC_ID" INTEGER , "STATE" INTEGER);	<!ELEMENT CATENTRY EMPTY> <!ATTLIST CATENTRY CATENTRY_ID CDATA #REQUIRED MEMBER_ID CDATA #REQUIRED CATENTTYPE_ID CDATA #REQUIRED MARKFORDELETE CDATA #REQUIRED PARTNUMBER CDATA #REQUIRED MFPARTNUMBER CDATA #IMPLIED MFNAME CDATA #IMPLIED URL CDATA #IMPLIED FIELD1 CDATA #IMPLIED FIELD2 CDATA #IMPLIED FIELD3 CDATA #IMPLIED FIELD4 CDATA #IMPLIED FIELD5 CDATA #IMPLIED LASTUPDATE CDATA #IMPLIED OID CDATA #IMPLIED ONSPECIAL CDATA #IMPLIED ONAUCTION CDATA #IMPLIED BUYABLE CDATA #IMPLIED BASEITEM_ID CDATA #IMPLIED CLASSIFGRP_ID CDATA #IMPLIED ITEMSPC_ID CDATA #IMPLIED STATE CDATA "1" >

DTD 작성기 기능을 설정하려면 다음을 수행하십시오.

1. 새 DTD 작성기 사용자 정의 특성 파일을 작성하십시오.
   - NT 2000 Solaris Linux

     DB2ConnectionCustomizer.properties는 DTDGenerator.zip 아카이브에 위치 지정됩니다. 이 파일을 추출하려면 확장자 .properties를 유지한 채 이름을

바꾸고 calsspath의 디렉토리에 위치 지정하십시오.
중요: 기존 DB2ConnectionCustomizer.properties 파일을 제거하거나 수정하지 마십시오.

- 400

  ISeries_GENWCSDTD_Customizer.properties는 /QIBM/ProdData/WebCommerce/properties 디렉토리에 위치합니다. 이 파일을 /*instroot*/xml 디렉토리에 복사하고 확장자 .properties를 유지한 채 새 파일의 이름을 바꿔서 새 파일에 필요한 변경을 작성하십시오.
  중요: 원래의 ISeries_GENWCSDTD_Customizer.properties 파일을 제거하거나 수정하지 마십시오.

2. 새 파일에 database-driver 값을 수정하십시오. 예를 들면, 다음과 같습니다.

```
DBVendorName = DB2
DBDriverName = COM.ibm.db2.jdbc.app.DB2Driver
DBURL = jdbc:db2:
```

여기서

- DBVendorName은 데이터베이스의 유형을 선택하는 데 사용됩니다.
  옵션은 다음과 같습니다.
  - iSeries용 DB2 Universal Database(DB2/iSeries)
  - 기타 운영체제용 DB2(DB2)
  - Oracle 데이터베이스(Oracle)
- DBDriverName은 JDBC 드라이버를 선택하는 데 사용됩니다.
  옵션은 다음과 같습니다.
  - iSeries용 DB2 Universal Database(com.ibm.db2.jdbc.app.DB2Driver)
  - 기타 운영체제용 DB2(COM.ibm.db2.jdbc.app.DB2Driver)
  - Oracle 데이터베이스(oracle.jdbc.driver.OracleDriver)
- DBURL은 데이터베이스에 액세스할 URL을 지정하는 데 사용됩니다.
  옵션은 다음과 같습니다.
  - iSeries(jdbc:db2://)용 DB2 Universal Database
  - 기타 운영체제(jdbc:db2:)용 DB2
  - Oracle 데이터베이스(jdbc:oracle:oci8:@)

3. DTD 생성 명령의 사용자 정의 매개변수 값으로 새 파일의 이름을 지정하십시오.

DTD 생성

TableNames.txt 입력 파일은 각 행에 다음 데이터베이스 테이블 이름을 포함합니다.

```
MEMBER
ADDRBOOK
ADDRESS
```

다음은 DTD 작성기를 호출할 수 있는 방법의 예입니다.

- NT 2000 AIX Solaris Linux

```
java com.ibm.wca.DTDGenerator.GenerateDTD -dbname MALL -dbuser db2inst1
-dbpwd db2ibm -outfile wc.dtd -infile TableNames.txt
```

- 400

```
QWEBCOMM/GENWCSDTD DATABASE(DATABASE_NAME) SCHEMA(MALL)
INSTROOT(/QIBM/UserData/WebCommerce/instances/mser)
PASSWD(mypassword) OUTFILE(wc.dtd) INFILE(TableNames.txt)
```

출력 파일 wc.dtd는 다음을 포함합니다.

```
<!ELEMENT MALL (( MEMBER | ADDRBOOK | ADDRESS)*)>
<!ELEMENT MEMBER EMPTY>
<!ATTLIST MEMBER
   MEMBER_ID            CDATA          #REQUIRED
   TYPE                 CDATA          #REQUIRED
   STATE                CDATA          #IMPLIED
>
<!ELEMENT ADDRBOOK EMPTY>
<!ATTLIST ADDRBOOK
   ADDRBOOK_ID          CDATA          #REQUIRED
   MEMBER_ID            CDATA          #REQUIRED
   TYPE                 CDATA          #IMPLIED
   DISPLAYNAME          CDATA          #REQUIRED
   DESCRIPTION          CDATA          #IMPLIED
>
<!ELEMENT ADDRESS EMPTY>
<!ATTLIST ADDRESS
   ADDRESS_ID           CDATA          #REQUIRED
   ADDRESSTYPE          CDATA          #IMPLIED
   MEMBER_ID            CDATA          #REQUIRED
   ADDRBOOK_ID          CDATA          #REQUIRED
   ORGUNITNAME          CDATA          #IMPLIED
   FIELD3               CDATA          #IMPLIED
   BILLINGCODE          CDATA          #IMPLIED
   BILLINGCODETYPE      CDATA          #IMPLIED
   STATUS               CDATA          #IMPLIED
   ORGNAME              CDATA          #IMPLIED
   ISPRIMARY            CDATA          #IMPLIED
   LASTNAME             CDATA          #IMPLIED
   PERSONTITLE          CDATA          #IMPLIED
   FIRSTNAME            CDATA          #IMPLIED
   MIDDLENAME           CDATA          #IMPLIED
   BUSINESSTITLE        CDATA          #IMPLIED
   PHONE1               CDATA          #IMPLIED
```

```
   FAX1                 CDATA          #IMPLIED
   PHONE2               CDATA          #IMPLIED
   ADDRESS1             CDATA          #IMPLIED
   FAX2                 CDATA          #IMPLIED
   NICKNAME             CDATA          #REQUIRED
   ADDRESS2             CDATA          #IMPLIED
   ADDRESS3             CDATA          #IMPLIED
   CITY                 CDATA          #IMPLIED
   STATE                CDATA          #IMPLIED
   COUNTRY              CDATA          #IMPLIED
   ZIPCODE              CDATA          #IMPLIED
   EMAIL1               CDATA          #IMPLIED
   EMAIL2               CDATA          #IMPLIED
   PHONE1TYPE           CDATA          #IMPLIED
   PHONE2TYPE           CDATA          #IMPLIED
   PUBLISHPHONE1        CDATA          #IMPLIED
   PUBLISHPHONE2        CDATA          #IMPLIED
   BESTCALLINGTIME      CDATA          #IMPLIED
   PACKAGESUPPRESSION   CDATA          #IMPLIED
   LASTCREATE           CDATA          #IMPLIED
   OFFICEADDRESS        CDATA          #IMPLIED
   SELFADDRESS          CDATA          "0"
   FIELD1               CDATA          #IMPLIED
   FIELD2               CDATA          #IMPLIED
   TAXGEOCODE           CDATA          #IMPLIED
   SHIPPINGGEOCODE      CDATA          #IMPLIED
>
```

스키마 및 상세 XML 파일 생성

DTD 작성기를 호출하는 예는 다음과 같습니다.

- NT 2000

```
java com.ibm.wca.DTDGenerator.GenerateDTD -dbname SAMPLE
-dbuser johndoe -dbpwd password -xmlTableDesc c:\sample\sample.xml
-outfile tables.dtd -tablenames "employee,staff"
```

- AIX Solaris Linux

```
java com.ibm.wca.DTDGenerator.GenerateDTD -dbname SAMPLE
-dbuser johndoe -dbpwd password -xmlTableDesc usr/sample/sample.xml
-outfile tables.dtd -tablenames "employee,staff"
```

- 400

```
QWEBCOMM/GENWCSDTD DATABASE(MYDB) SCHEMA(SAMPLE)
INSTROOT(/QIBM/UserData/WebCommerce/instances/mser) PASSWD(mypassword)
OUTFILE(tables.dtd) TABNAMES('employee,staff') XMLTABDESC(/sample/sample.xml)
```

sample 디렉토리에 스키마 파일이 작성되고 파일 이름은 WCAWebForm.xsd입니다. sample.xml 출력 파일은 다음을 포함합니다.

```
<?xml version="1.0" encoding="UTF-8"?>
  <formList xmlns="WCAWebForm.xsd" dbname="SAMPLE" dtdname="tables.dtd">
    <form name="EMPLOYEE">
      <uniqueIndex name="U2" columns="FIRSTNME,LASTNAME"/>
      <uniqueIndex name="U3" columns="MIDINIT,LASTNAME"/>
```

```
    <field name="EMPNO" type="string" maxlength="6" minOccurs='1'
        uniqueKey="true" showColumnInList="true" />
    <field name="FIRSTNME" type="string" maxlength="32" minOccurs='1'
        showColumnInList="true" />
    <field name="MIDINIT" type="string" maxlength="1" minOccurs='1'
        showColumnInList="true" />
    <field name="LASTNAME" type="string" maxlength="15" minOccurs='1'
        showColumnInList="true" />
    <field name="WORKDEPT" type="string" maxlength="3" showColumnInList="true" />
    <field name="PHONENO" type="string" maxlength="4"/>
    <field name="HIREDATE" type="date" maxlength="10"/>
    <field name="JOB" type="string" maxlength="8"/>
    <field name="EDLEVEL" type="integer" maxlength="5" minOccurs='1' />
    <field name="SEX" type="string" maxlength="1"/>
    <field name="BIRTHDATE" type="date" maxlength="10"/>
    <field name="SALARY" type="decimal" maxlength="9"/>
    <field name="BONUS" type="decimal" maxlength="9"/>
    <field name="COMM" type="decimal" maxlength="9"/>
  </form>
  <form name="STAFF">
    <field name="ID" type="integer" maxlength="5" minOccurs='1'
        uniqueKey="true" showColumnInList="true" />
    <field name="NAME" type="string" maxlength="9" showColumnInList="true" />
    <field name="DEPT" type="integer" maxlength="5" showColumnInList="true" />
    <field name="JOB" type="string" maxlength="5" showColumnInList="true" />
    <field name="YEARS" type="integer" maxlength="5" showColumnInList="true" />
    <field name="SALARY" type="decimal" maxlength="7"/>
    <field name="COMM" type="decimal" maxlength="7"/>
  </form>
</formList>
```

제 5 장 식별자 분석

대상 데이터베이스에 로드될 XML 데이터는 XML 요소에 필요한 식별자를 포함해야 합니다. XML 문서의 카탈로그 엔티티에 대해 식별자를 생성 및 위치 지정하려면, ID 분석 명령을 호출하십시오.

ID 분석기는 기본 테이블에 대한 식별자를 분석만 합니다. 기본 테이블은 KEYS 또는 SUBKEYS 테이블에 나열된 것입니다. KEYS 또는 SUBKEYS 외의 테이블에 대해 식별자를 분석할 필요가 있는 경우, ID 분석기를 실행하기 전에 SUBKEYS 테이블에 테이블을 추가하십시오.

다음은 ID 분석기 사용이 필요할 수 있는 상황의 예입니다.
- XML 포맷에서 새 컨텐츠를 로드 중, 데이터에 대한 식별자가 필요합니다.
- 컨텐츠 갱신 중, 데이터베이스의 오브젝트에 이미 식별자가 존재합니다.

ID 분석기는 실제 식별자 또는 다음 기법을 사용하여 분석될 수 있는 식별자를 제공할 수 있습니다.
- 내부 별명 분석
- 특성 파일 스펙
- 고유 색인 분석

ID 분석기 설정

ID 분석기의 시간소인, 저장영역 및 데이터베이스 드라이버 핸들 방법을 설정하려면 다음을 수행하십시오.

1. 새 ID 분석기 사용자 정의 특성 파일을 작성하십시오.
   - NT 2000 AIX Solaris Linux

     DB2ConnectionCustomizer.properties는 IdResGen.zip 아카이브에 위치 지정됩니다. 이 파일을 추출하려면 확장자 .properties를 유지한 채 이름을 바꾸고 calsspath의 디렉토리에 위치 지정하십시오.
     중요: 기존 DB2ConnectionCustomizer.properties 파일을 제거하거나 수정하지 마십시오.

   - 400

     ISeries_RESWCSID_Customizer.properties는 /QIBM/ProdData/WebCommerce/properties 디렉토리에 위치합니다. 이 파일을 /*instroot*/xml 디렉토리에 복사하고 확장자 .properties를 유지한 채 새 파일의 이름을 바꿔

서 새 파일에 필요한 변경을 작성하십시오.

중요: 원래의 ISeries_RESWCSID_Customizer.properties 파일을 제거하거나 수정하지 마십시오.

2. 새 파일에 지정된 특성값을 수정하십시오.
3. ID 분석 명령의 사용자 정의 매개변수 값으로 새 파일의 이름을 지정하십시오.

ID 분석기의 시간소인 핸들 방법 설정

다음 기본 입력 시간소인 마스크는 ID 분석기 사용자 정의 특성 파일에 제공됩니다.

```
InputTimeStampFormat.1 = yyyy-DD hh:mm:ss.SSSSSS
InputTimeStampFormat.2 = yyyy-MM-dd hh:mm:ss.SSSSSS
InputTimeStampFormat.3 = yyyy-DD-hh.mm.ss.SSSSSS
InputTimeStampFormat.4 = yyyy-MM-dd-HH.mm.ss.SSSSSS
InputTimeStampFormat.5 = yyyy-MM-dd-hh.mm.ss.SSSSSS
InputTimeStampFormat.6 = yyyy-MM-dd HH:mm:ss.SSSSSS
InputTimeStampFormat.7 = yyyy-DD HH:mm:ss.SSSSSS
```

시간소인 마스크를 수정하거나 또는 ID 분석기 사용자 정의 특성 파일에 원하는 만큼 추가할 수 있습니다. 입력 시간소인을 추가할 경우, 현재 순서 다음 번호를 사용해야 합니다(예를 들어 위 목록에 추가 중인 경우, 다음 입력 시간소인 마스크는 InputTimeStampFormat.8입니다).

또한 ID 분석기 사용자 정의 특성 파일에 다음 특성값을 수정해서 시간소인 포맷, 마이크로초 마스크 및 데이터베이스 고유의 포맷을 사용자 정의할 수 있습니다.

```
TargetTimeStampFormat = yyyy-MM-dd HH:mm:ss.SSSSSS
MicroSecondMask = SSSSSS
DatabaseSpecificFormat = YYYY-MM-DD HH24:MI:SS
```

ID 분석기의 저장영역 핸들 방법 설정

다음은 영속 해시맵에 관련된 특성에 대한 기본값을 지정하는 ID 분석기 사용자 정의 특성 파일 내의 섹션입니다.

```
////////////////////////////////////////////////////////////////////////
/// 0 = Normal hashmap with no backend storage
/// 1 = JDBM
////////////////////////////////////////////////////////////////////////

PersistentStorageType = 0

////////////////////////////////////////////////////////////////////////
/// If PersistentStorageType != 0, set MemoryStorageSize to the maximum
/// size of the hashmap in memory data and after that the hashmap will
/// stream the data to a persistent storage as specified
/// If -1, then it uses the normal hashmap with no backend storage
////////////////////////////////////////////////////////////////////////

MemoryStorageSize = 1
```

ID 분석기 사용자 정의 특성 파일의 PersistentStorageType 값을 설정해서 ID 분석기의 영속 저장영역 핸들 방법을 지정할 수 있습니다.

- PersistentStorageType = 0으로 설정할 경우, ID 분석기는 "정상" 방식(기호 hashmap이 메모리에 존재)에서 작동합니다.
- PersistentStorageType = 1로 설정할 경우, 기호 및 키를 지속시키기 위해 JDBM을 사용합니다.

ID 분석기 사용자 정의 특성 파일의 MemoryStorageSize 값을 설정해서 메모리에 저장된 레코드의 수를 지정할 수 있습니다.

- MemoryStorageSize에 대한 "1" 값은 메모리에 보존된 유일한 하나의 레코드를 표시합니다.
- MemoryStorageSize에 대한 "-1" 값은 메모리에 보존된 모든 레코드를 표시하는 특정 의미를 가집니다.

이 경우 ID 분석기는 "정상" 작동 이전 상태로 변경됩니다.

ID 분석기의 데이터베이스 드라이버 핸들 방법 설정

ID 분석기 사용자 정의 특성 파일의 다음 행은 데이터베이스 드라이버에 대한 기본값을 지정합니다.

```
DBVendorName = DB2
DBDriverName = COM.ibm.db2.jdbc.app.DB2Driver
DBURL = jdbc:db2:
```

여기서

- DBVendorName은 데이터베이스의 유형을 선택하는 데 사용됩니다.

  옵션은 다음과 같습니다.

  – iSeries(DB2/iSeries)용 DB2 Universal Database
  – 기타 운영체제(DB2)용 DB2
  – Oracle 데이터베이스(oracle)
- DBDriverName은 JDBC 드라이버를 선택하는 데 사용됩니다.

  옵션은 다음과 같습니다.

  – iSeries(com.ibm.db2.jdbc.app.DB2Driver)용 DB2 Universal Database
  – 기타 운영체제(COM.ibm.db2.jdbc.app.DB2Driver)용 DB2
  – Oracle 데이터베이스(oracle.jdbc.driver.OracleDriver)
- DBURL은 데이터베이스에 액세스할 URL을 지정하는 데 사용됩니다.

  옵션은 다음과 같습니다.

  – iSeries(jdbc:db2://)용 DB2 Universal Database
  – 기타 운영체제(jdbc:db2:)용 DB2

– Oracle 데이터베이스(jdbc:oracle:oci8:@)

데이터 처리 방법 결정

ID 분석 명령은 입력 파일 처리를 위해 load, update 또는 mixed 메소드를 선택하게 합니다.

- 파일의 **모든** 레코드가 데이터베이스에 ***존재하지 않을*** 경우 load 메소드를 사용하여 입력 파일을 처리하십시오.
- 파일의 **모든** 레코드가 데이터베이스에 ***존재할*** 경우 update 메소드를 사용하여 입력 파일을 처리하십시오.
- 파일의 ***단지 일부*** 레코드가 데이터베이스에 ***존재할*** 경우 mixed 메소드를 사용하여 입력 파일을 처리하십시오.

load 메소드 선택

ID 분석기의 load 메소드는 데이터베이스에 로드되는 레코드에 대한 새 식별자를 생성하는 데 사용됩니다. 이 메소드로 새 식별자가 레코드에 작성됩니다. 다음 예는 새 데이터 식별자를 생성하는 데 사용됩니다.

- NT 2000

```
idresgen -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method load -customizer customizer -schemaname mall
```

load 메소드는 기본값입니다.

- AIX Solaris Linux

```
./idresgen.sh -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method load -customizer customizer -schemaname mall
```

load 메소드는 기본값입니다.

- 400

```
QWEBCOMM/RESWCSID DATABASE(DATABASE_NAME) SCHEMA(MALL)
INSTROOT(/QIBM/UserData/WebCommerce/instances/mser)
PASSWD(mypassword) INFILE(input.xml) OUTFILE(output.xml)
METHOD(*LOAD)
```

update 메소드 선택

ID 분석기에 update 메소드를 지정할 경우, 입력 파일의 레코드가 기존 데이터베이스에 이미 있어야 합니다. ID 분석기는 식별자를 데이터베이스에 위치 지정합니다. 데이터베이스에 레코드가 존재하지 않을 경우, ID 분석기는 이 레코드에 대한 식별자를 분석할 수 없으며 오류가 발생했다고 표시합니다. 다음 예는 데이터베이스에 이미 있는 데이터에 대한 식별자를 위치 지정하는 데 사용됩니다.

- NT 2000

```
idresgen -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method update -customizer customizer -schemaname mall
```

- AIX Solaris Linux

```
./idresgen.sh -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method update -customizer customizer -schemaname mall
```

- 400

```
QWEBCOMM/RESWCSID DATABASE(DATABASE_NAME) SCHEMA(MALL)
INSTROOT(/QIBM/UserData/WebCommerce/instances/mser) PASSWD(mypassword)
INFILE(input.xml) OUTFILE(output.xml) METHOD(*UPD)
```

mixed 메소드 선택

입력 데이터 파일이 일부 새 레코드와 데이터베이스에 이미 존재하는 레코드를 포함할 경우, ID 분석기가 mixed 메소드를 사용하여 실행되어야 합니다. 데이터베이스에 레코드가 있지 않을 경우에만 ID 분석기가 이 메소드로 레코드에 대한 새 식별자를 작성합니다. 그렇지 않으면 기존 식별자를 데이터베이스에서 얻습니다. 다음 예는 새 데이터에 대한 식별자를 생성하고 데이터베이스에 이미 있는 데이터에 식별자를 위치 지정하는 데 사용됩니다.

- NT 2000

```
idresgen -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method mixed -customizer customizer -schemaname mall
```

- AIX Solaris Linux

```
./idresgen.sh -dbname db -dbuser user -dbpwd pwd -infile input.xml
-outfile output.xml -method mixed -customizer customizer -schemaname mall
```

- 400

```
QWEBCOMM/RESWCSID DATABASE(DATABASE_NAME) SCHEMA(MALL)
INSTROOT(/QIBM/UserData/WebCommerce/instances/mser) PASSWD(mypassword)
INFILE(input.xml) OUTFILE(output.xml) METHOD(*MIX)
```

ID 분석 기법 사용

ID 분석기는 Java 특성 파일을 사용하여 기본 테이블의 어느 열이 기본 테이블의 식별자를 필요로 하는 테이블을 찾는 데 사용될 수 있는지를 판별합니다. KEYS나 SUBKEYS 테이블에 나열된 경우, 테이블이 기본입니다.

ID 분석기로 내부 별명 분석을 사용하면 별명이 XML 파일의 1차 키 속성(식별자)에 위치 지정됩니다. 그러면 별명은 XML 파일 전체에서 해당 요소를 참조하는 데 사용될 수 있습니다. 이 처리는 프로그램이 XML 파일을 빌드하는 데 필요한 고유 색인을 판별할 필요성을 제거합니다.

또한 ID 분석기는 요구사항을 충족하는 고유 색인이 있는지 여부를 판별하기 위해 데이터베이스 스키마를 분석할 수 있습니다. 분석되고 있는 테이블의 특성 파일에 항목이

없거나 특성 파일이 없을 때에만, ID 분석기는 고유 색인을 찾습니다. 이 조건이 참일 경우, 고유 색인 확인이 수행됩니다. 고유 색인이 존재하나 테이블에 대한 1차 키를 포함하지 않을 경우, 이는 올바른 것으로 간주됩니다.

ID 분석기로 특성 파일 지정

ID 분석기는 대체 Java 특성 파일을 사용하여 기본 항목의 어느 열이 기본 행의 식별자를 필요로 하는 테이블을 찾는 데 사용되어야 하는지를 설명하게 합니다.

기본 특성 파일은 `IdResolveKeys.properties`이지만 원할 경우 ID 분석 명령을 호출할 때, 이 파일을 수정하거나 개인 파일을 지정할 수 있습니다.

- NT 2000 AIX Solaris Linux

  `IdResolveKeys.properties`는 다음 디렉토리에 위치 지정됩니다.

  - NT *drive*:\WebSphere\CommerceServer\properties
  - 2000 *drive*:\Program Files\WebSphere\CommerceServer\properties
  - AIX /usr/WebSphere/CommerceServer/properties
  - Solaris Linux /opt/WebSphere/CommerceServer/properties

  ID 분석기를 실행할 때 현재 디렉토리에 이 파일을 위치시키지 않으면, classpath 환경 변수에 정의된 디렉토리에 위치시킬 수 있습니다. 또한 이 파일에 대한 전체 경로를 지정할 수도 있습니다.

- 400

  `IdResolveKeys.properties`를 변경하려면, `/QIBM/ProdData/WebCommerce/properties` 디렉토리에서 복사해서 `/instroot/xml` 디렉토리로 저장한 후, 새 파일에 필요한 변경을 작성하십시오.

  주: 위 디렉토리는 RESWCSID 명령이 사용하는 classpath에 있습니다.

식별자 생성을 위한 특성 파일 사용

다음 예에서 ADDRBOOK과 ADDRESS 레코드 각각에 대해 ADDRBOOK_ID와 ADDRESS_ID 식별자를 분석하는 것이 필요합니다. MEMBER 레코드에 대한 식별자는 이미 알려져 있습니다. 개별 레코드는 WebSphere Commerce 데이터베이스에 대한 올바른 식별자를 필요로 합니다. 또한 ADDRESS 레코드의 ADDRBOOK_ID는 해당 foreign-key 제한자를 만족시키기 위해 기본 테이블에서 식별자를 필요로 합니다.

```
<MEMBER
  MEMBER_ID="100"
  TYPE="U"
    STATE="1"
/>
<MEMBER
  MEMBER_ID="101"
```

```
  TYPE="U"
    STATE="1"
/>
<ADDRBOOK
  MEMBER_ID="100"
  DISPLAYNAME="Friends"                DISPLAYNAME 열의 실제 값
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRBOOK_ID="@Friends"               찾아보기로 DISPLAYNAME 값을 사용해서
                                       ADDRBOOK을 참조
  MEMBER_ID="101"
  NICKNAME="Bob"
  ADDRESS1="15 Brave Developers St."
  CITY="Toronto"
  ZIPCODE="A0A0A0"
  COUNTRY="Canada"
  STATUS="P"
/>
```

foreign-key 열에 대한 식별자를 분석할 때, 관계 행이 기본 행의 어느 열을 사용할 지를 판별하려면 특성 파일이 필요합니다. 다음 프로시저는 구문 분석 중인 위 파일이 정확하게 처리되는지 여부를 확인합니다.

IdResolveKeys.properties에서 다음을 지정하십시오.

```
NAMEDELIMITER=@
SELECTDELIMITER=:

ADDRBOOK=@DISPLAYNAME:DISPLAYNAME
ADDRESS=@NICKNAME:NICKNAME
```

NAMEDELIMITER 및 SELECTDELIMITER가 전체 특성 파일에 사용되는 분리문자를 설정하면, 이는 지속적으로 사용되어야 합니다.

ADDRBOOK=@DISPLAYNAME:DISPLAYNAME은 주소록 레코드를 수신하면 주소록 행에 대한 식별자가 작성됨을 말합니다. DISPLAYNAME 필드는 입력 레코드에서 추출되고 새 식별자에 연관을 형성하는 데 사용됩니다. DISPLAYNAME 문자열은 주소록 행 DISPLAYNAME에 일치시키는 데 사용되고 foreign key가 필요로 하는 식별자를 분석합니다.

이전 입력의 예를 사용하여(DISPLAYNAME is Friends), 이 레코드에 작성된 식별자가 12951임을 추측할 수 있습니다. DISPLAYNAME은 12951에 대한 look-aside 키로 사용됩니다. 다음 레코드 ADDRESS(ADDRBOOK_ID가 ″@...″의 양식을 가짐)로 처리를 계속합니다(이는 주소록 식별자를 찾는 데 사용되는 다음 분리문자를 표시함). 문자열은 DISPLAYNAME과 일치하고 12951은 리턴하여 ADDRBOOK_ID 속성에 위치 지정됩니다.

```
<MEMBER
  MEMBER_ID="100"
  TYPE="U"
    STATE="1"
/>
<MEMBER
  MEMBER_ID="101"
  TYPE="U"
    STATE="1"
/>
<ADDRBOOK
  ADDRBOOK_ID="12951"                생성된 1차 키
  MEMBER_ID="100"
  DISPLAYNAME="Friends"              변경되지 않은 ADDRBOOK DISPLAYNAME 값
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRESS_ID="13051"                 생성된 1차 키
  ADDRBOOK_ID="12951"                ADDRESS는 정확한 ADDRBOOK 참조
  MEMBER_ID="101"
  NICKNAME="Bob"
  ADDRESS1="15 Brave Developers St."
  CITY="Toronto"
  ZIPCODE="A0A0A0"
  COUNTRY="Canada"
  STATUS="P"
/>
```

복합 키로 특성 파일 사용

두 개 이상의 열로 이루어진 키는 복합 키입니다. 특성 파일에서 필드 이름 다음의 NAMEDELIMITER 및 SELECTDELIMITER 둘 다를 지정해서 복합 키 찾아보기를 정의할 수 있습니다. ADDRBOOK에 대한 찾아보기 기준을 얻으려면 레코드가 표시 이름과 구성원 ID의 복합어여야 합니다. (예를 들어, 특성 파일에서 다음을 지정하십시오.)

```
ADDRBOOK=@DISPLAYNAME@MEMBER_ID:DISPLAYNAME MEMBER_ID
```

그러면 XML 입력 파일 코드 부분이 다음에 옵니다.

```
<ADDRBOOK
  MEMBER_ID="100"
  DISPLAYNAME="Friends"                MEMBER 100의 ADDRBOOK "Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRBOOK
  MEMBER_ID="101"
  DISPLAYNAME="Friends"                MEMBER 101의 ADDRBOOK "Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRBOOK_ID="@Friends@100"           MEMBER 100의 ADDRBOOK "Friends"에 대한
```

1차 키 찾아보기

```
  MEMBER_ID="101"
  NICKNAME="Bob"
  ADDRESS1="15 Brave Developers St."
  CITY="Toronto"
  ZIPCODE="A0A0A0"
  COUNTRY="Canada"
  STATUS="P"
/>
```

분석 후 다음이 산출됩니다.

```
<MEMBER
  MEMBER_ID="100"
  TYPE="U"
    STATE="1"
/>
<MEMBER
  MEMBER_ID="101"
  TYPE="U"
    STATE="1"
/>
<ADDRBOOK
  ADDRBOOK_ID="12951"                    관심 ADDRBOOK
  MEMBER_ID="100"
  DISPLAYNAME="Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRBOOK
  ADDRBOOK_ID="12952"
  MEMBER_ID="101"
  DISPLAYNAME="Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRESS_ID="13051"
  ADDRBOOK_ID="12951"                    ADDRESS는 정확한 ADDRBOOK 참조
  MEMBER_ID="101"
  NICKNAME="Bob"
  ADDRESS1="15 Brave Developers St."
  CITY="Toronto"
  ZIPCODE="A0A0A0"
  COUNTRY="Canada"
  STATUS="P"
/>
```

계단식 1차 키로 특성 파일 사용

1차 키 테이블 STOREENT는 1차 키 STOREENT_ID를 정의합니다. STOREENT를 참조하는 foreign 테이블인 STORE는 기본 테이블 STOREENT의 foreign key인 1차 키 STORE_ID를 정의합니다. 이는 STORE_ID가 STOREENT_ID 값 중의 하나여야 함을 의미합니다. 따라서 foreign 테이블 STORE의 1차 키인 STORE_ID는 이중 역할(1차 및 foreign)을 가집니다.

다른 테이블 CONTRACT는 STORE의 foreign 테이블이며 CONTRACT에 대한 foreign key인 STORE_ID는 STORE의 1차 키 STORE_ID를 참조한다고 가정해봅시다. 그러면 STORE 테이블이 CONTRACT 테이블에 대한 기본 테이블입니다.

STORE 테이블의 STORE_ID는 작성된다기 보다는 STOREENT_ID에서 참조되기 때문에, ID 분석기는 STORE 테이블에 대한 내부 별명 및 ID 값 연관을 작성하지 않습니다. CONTRACT 테이블이 STORE 테이블에서 STORE_ID를 분석하려 시도하면 빈 값을 가져옵니다.

이러한 특별 조건으로 인해, 특성 파일의 항목을 작성해서 내부 별명의 작성을 명시적으로 지정해야 합니다. IdResolveKeys.properties에서 다음을 지정하십시오.

```
"STORE=@STORE_ID:STORE_ID"
```

이는 ID 분석기가 다음을 강제 수행하게 합니다.

- foreign 참조로 STORE_ID를 분석하는 동안 내부 별명 및 ID 값 연관을 작성하십시오.
- CONTRACT 테이블에 대한 STORE_ID를 분석하는 동안 연관을 사용하십시오.

특성 파일에서 STORE=@STORE_ID:STORE_ID 항목을 사용하면 XML 입력 파일의 코드 부분이 다음에 나옵니다.

```
<STOREENT
  IDENTIFIER="Out Fashions"
       MEMBER_ID="-2000"
  STOREENT_ID="@storeent_id_1"
  TYPE="G"
/>
<STORE
  STORE_ID="@storeent_id_1"
  STOREGRP_ID="1"
  STORELEVEL="store_level"
/>
<CONTRACT
  CONTRACT_ID="@contract_id_1"
  STATE="0"
  STORE_ID="@storeent_id_1"
/>
```

분석 후 다음이 산출됩니다.

```
<STOREENT
  IDENTIFIER="Out Fashions"
       MEMBER_ID="-2000"
  STOREENT_ID="10501"
  TYPE="G"
/>
<STORE
  STORE_ID="10501"
  STOREGRP_ID="1"
```

```
  STORELEVEL="store_level"
/>
<CONTRACT
  CONTRACT_ID="@contract_id_1"
  STATE="0"
  STORE_ID="10501"
/>
```

내부 별명 분석 사용

ID 분석기로 내부 별명 분석을 사용하면 별명이 XML 파일의 1차 키 속성(식별자)에 위치 지정됩니다. 그러면 별명은 XML 파일 전체에서 해당 요소를 참조하는 데 사용될 수 있습니다. 이 처리는 프로그램이 XML 파일을 빌드하는 데 필요한 고유 색인을 판별할 필요성을 제거합니다.

내부 별명은 전체 파일에 지속적으로 사용되어야 합니다. 주소록 ID ADDRBOOK_ID가 @addrbook_1의 별명으로 사용될 경우, 파일의 ID를 참조하는 모든 foreign-key는 @addrbook_1을 사용해야 합니다. 별명이 일시적임에 유의하십시오. 이들은 저장되지 않으며, 별명을 다시 소개하지 않은 분리 XML 파일에서 사용될 수 없습니다.

내부 별명 ID 분석 사용의 부분 예

분석 이전:

```
<MEMBER
  MEMBER_ID="100"
  TYPE="U"
    STATE="1"
/>
<ADDRBOOK
  ADDRBOOK_ID="@addrbook_1"            ADDRBOOK의 별명
  MEMBER_ID="100"
  DISPLAYNAME="Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRESS_ID="@address_1"              ADDRESS의 별명
  ADDRBOOK_ID="@addrbook_1"            ADDRBOOK의 별명 참조
  MEMBER_ID="101"
  NICKNAME="Bob"
  ADDRESS1="1 Brave Developer St."
  CITY="Toronto"
  ZIPCODE="A3B0F4"
  COUNTRY="Canada"
  STATUS="P"
/>
```

분석 이후:

```
<MEMBER
  MEMBER_ID="100"
  TYPE="U"
```

```
    STATE="1"
/>
<ADDRBOOK
  ADDRBOOK_ID="11801"                  생성된 1차 키
  MEMBER_ID="100"
  DISPLAYNAME="Friends"
  DESCRIPTION="All my friends"
  TYPE="P"
/>
<ADDRESS
  ADDRESS_ID="11901"                   생성된 1차 키
  ADDRBOOK_ID="11801"                  ADDRBOOK 항목 참조
  MEMBER_ID="100"
  NICKNAME="Bob"
  ADDRESS1="1 Brave Developer St."
  CITY="Toronto"
  ZIPCODE="A3B0F4"
  COUNTRY="Canada"
  STATUS="P"
/>
```

고유 색인 분석 사용

분석되고 있는 테이블에 대한 특성 파일의 항목이 없거나 특성 파일이 없을 때, ID 분석기의 기본 작동인 고유 색인 분석이 사용됩니다. 고유 색인 분석은 식별자 위치 지정 수단으로 테이블에 지정된 모든 고유 색인을 사용합니다. (예를 들어, MEMBER_ID 및 IDENTIFIER는 CATALOG 테이블의 고유 색인입니다.) 따라서 CATALOG 테이블의 1차 키 CATALOG_ID의 분석 지점으로 사용될 수 있습니다.

데이터베이스의 컨텐츠를 갱신하려면, 데이터베이스의 기본 테이블에서 고유 키를 알아야 합니다. 이를 찾으려면 데이터베이스를 조회하십시오. 예를 들어, 고유 키를 검색하는 DB2 명령은 다음과 같습니다.

```
db2 describe indexes for table schema.tablename show detail
```

고유 색인 분석의 부분 예

분석 이전:

```
<MEMBER
  MEMBER_ID="100"
    TYPE="O"
    STATE="1"
/>

<CATALOG
  DESCRIPTION="Winter Catalog"
  IDENTIFIER="WC2001"
  MEMBER_ID="100"
  TPCLEVEL="2"
/>

<CATALOGDSC
```

```
  CATALOG_ID="@WC2001@100"              구성원 "100"의 카탈로그 "WC2001"
                                        다시 참조(주: 순서가 중요합니다.)
  FULLIMAGE="c:\store\img\wc.gif"
  LANGUAGE_ID="-1"
  LONGDESCRIPTION="2001 Winter Catalog"
  SHORTDESCRIPTION="2001 Winter Catalog"
  NAME="InFashion 2001 Winter Catalog"
  THUMBNAIL="c:\store\img\wc_th.gif"
/>
```

분석 이후:

```
<MEMBER
  MEMBER_ID="100"
    TYPE="O"
    STATE="1"
/>
<CATALOG
  CATALOG_ID="10351"                    자동 생성된 1차 키
  DESCRIPTION="Winter Catalog"
  IDENTIFIER="WC2001"
  MEMBER_ID="100"
  TPCLEVEL="2"
/>
<CATALOGDSC
  CATALOG_ID="10351"                    정확한 카탈로그 참조
  FULLIMAGE="c:\store\img\wc.gif"
  LANGUAGE_ID="-1"
  LONGDESCRIPTION="2001 Winter Catalog"
  SHORTDESCRIPTION="2001 Winter Catalog"
  NAME="InFashion 2001 Winter Catalog"
  THUMBNAIL="c:\store\img\wc_th.gif"
/>
```

MEMBER 테이블에 데이터 로드

ID 분석기는 시스템이 생성한 식별자를 가지는 테이블에 대한 분석을 핸들합니다. 이는 KEYS 또는 SUBKEYS 테이블에 등록된 어떠한 테이블과 열도 포함합니다. 이 분석은 두 가지 구성요소를 가집니다.

1. 데이터베이스에 기본 테이블(KEYS 또는 SUBKEYS에 나열된 테이블) 존재 여부 판별

   이 분석은 고유 색인 분석 또는 특성 파일 스펙 중 하나를 사용하는 XML 데이터의 컨텐츠에 기초합니다.

2. 기본 테이블에 대한 foreign key 존재 여부 판별

   관련된 테이블의 foreign-key 속성에 있는 분석 스펙으로 이를 수행합니다.

MEMBER 테이블은 ORGENTITY, MBRGRP 및 USER 테이블에 대한 ″최상위 클래스″로 사용됩니다. 테이블이 MEMBER 테이블의 부속유형에 foreign-key 제한자를 보유할 때, 이는 참조 무결성을 유지보수하는 데 유용한 ″is-a″ 패턴을 작성합니다. 그러나 모든 MEMBER 부속유형이 공통의 기본 유형을 공유하기 때문에 식별자는 부속

유형 사이에서 고유해야 합니다. 이는 ORGENTITY_ID가 MBRGRP_ID와 USER_ID 설정에서 고유한 것이여야 함을 의미합니다. 이를 완료하기 위해 KEYS 테이블은 ORGENTITY, MBRGRP 및 USER 테이블만을 참조하며 그들 식별자에 대한 배타적인 범위를 지정합니다. 개별 부속유형은 1차 키를 보유하며 각각의 1차 키는 또한 MEMBER 테이블 1차 키에 foreign key입니다.

MEMBER 및 해당 부속유형 사이의 제한자는 MEMBER 및 부속유형이 동기화된 ID를 가질 수 없는 장소를 작성합니다. ORGENTITY, MBRGRP 및 USER 테이블을 시스템에 로드하기 위해 ID 분석기는 ″is-a″ 패턴을 인식하고 적절히 처리합니다. 다음은 ID 분석기에 대한 XML 구문입니다.

```
<ORGENTITY
    ORGENTITY_ID="@orgAlias"
    ORGENTITYNAME="Test Org"
    ORGENTITYTYPE="O">
    <ISA>
       <MEMBER
    TYPE="O"
    STATE="1"
       />
    </ISA>
</ORGENTITY>
```

다음을 생성합니다.

```
<MEMBER
    MEMBER_ID="12345"
    TYPE="O"
    STATE="1"
/>
<ORGENTITY
    ORGENTITY_ID="12345"          구성원 요소로 동기화
    ORGENTITYNAME="Test Org"
    ORGENTITYTYPE="O"
/>
```

이와 같이 ID 분석기는 <isa> 부속요소를 핸들하고 동기화된 식별자를 작성합니다.

REFKEYS 테이블을 사용하여 foreign 관계 작성

REFKEYS 테이블은 데이터베이스에 이미 존재하지 않는 테이블 사이에 foreign 관계를 표시하기 위해 작성됩니다. 일반적으로 데이터베이스 스키마는 다른 테이블에 테이블의 열을 링크하는 foreign-key 선언을 작성해서 foreign 관계를 설명합니다. 데이터베이스 스키마가 정의된 foreign 관계를 가지지 않으며 식별자가 foreign-key로 분석되어야 하는 경우, 다음을 수행하십시오.

1. 다음 예 DDL로 REFKEYS 테이블을 작성하십시오.

```
CREATE TABLE "REFKEYS" (
            "FKTABLE_NAME"    CHAR(18)   NOT NULL ,
            "FKCOLUMN_NAME"   CHAR(18)   NOT NULL ,
            "TABLENAME"       CHAR(18)   NOT NULL
);
```

여기서

FKTABLE_NAME은 foreign(또는 ″하위″) 테이블의 이름
FKCOLUMN_NAME은 foreign 열의 이름
TABLENAME은 기본(또는 ″상위″) 테이블의 이름

2. 필수 foreign 관계를 설명하는 REFKEYS 테이블에서 항목을 작성하십시오.

문제점 해결 오류

식별자를 분석하는 중 오류가 발생한 경우, 다음 테이블을 참조하십시오.

오류	사용된 방법	가능한 원인		가능한 해결책
미해결된 1차 키	전체	ID 분석기는 KEYS 또는 SUBKEYS 테이블 중 하나에 지정되지 **않은** 테이블의 1차 키(식별자)를 분석하지 **않습니다**.		ID 분석기를 실행하기 전에 1차 키가 SUBKEYS 테이블로 분석될 테이블의 이름을 추가하십시오. 그리고 foreign key 관계가 데이터베이스 스키마에 있는지 확인하십시오.
	갱신	1차 키는 데이터베이스를 조회하여 분석됩니다. 데이터베이스 조회는 특성 파일 항목 정보나 주어진 테이블의 고유 색인 중 하나를 사용하여 생성됩니다. 특성 파일 항목이 우선순위를 갖습니다.		입력 파일의 고유 색인 정보가 올바른지 확인하십시오. 또한 1차 키를 분석하기 위한 적절한 데이터베이스 조회를 생성하기 위해 특성 파일에서 적절한 항목을 작성하거나 수정하려고 할 수 있습니다.
미해결된 foreign key	전체	ID 분석기는 KEYS나 SUBKEYSR 테이블 중 하나에 지정되지 **않은** 테이블에 대한 외부 참조가 있는 테이블의 foreign key를 분석하지 **않습니다**.		ID 분석기를 실행하기 전에 SUBKEYS 테이블로 참조되는 테이블의 이름을 추가하십시오. 그리고 foreign key 관계가 데이터베이스 스키마에 있는지 확인하십시오.
		foreign key는 내부 별명을 사용하거나 데이터베이스를 조회하여 분석됩니다. 데이터베이스 조회는 내부 별명이 foreign key를 분석하는 데 실패한 경우에만 수행됩니다.	내부 별명은 1차 키를 사용하고 특성 파일 항목을 사용하여 생성됩니다.	입력 파일에서 소스와 대상으로 사용되는 내부 별명 값이 올바른지 확인하십시오.
			데이터베이스 조회는 특성 파일 항목 정보나 주어진 테이블의 고유 색인 중 하나를 사용하여 생성됩니다. 특성 파일 항목이 우선순위를 갖습니다.	입력 파일의 고유 색인 정보가 올바른지 확인하십시오. 또한 1차 키를 분석하기 위한 적절한 데이터베이스 조회를 생성하기 위해 특성 파일에서 적절한 항목을 작성하거나 수정하려고 할 수 있습니다.

제 6 장 데이터 로드

데이터를 로드하기 전에 다음을 수행해야 합니다.

1. 로더(데이터가 로드된 최초 시간)를 사용하여 DTD 및 스키마를 생성하십시오.

   주: 상점 아카이브 데이터를 로드 중이고 상점 아카이브와 함께 제공된 DTD를 사용해서 XML 파일을 작성한 경우, 이 단계를 건너 뛰십시오.

2. 식별자 분석(필요한 경우)

데이터는 연관된 DTD와 함께 XML 포맷 내에 있어야 합니다. 데이터를 로드하려면 로드 명령을 호출하십시오.

로더 설정

로더 패키지는 다음을 수행하여 로더가 기능하는 방법을 설정하게 합니다.

- 입력 파일의 요소 무시
- 열에 NULL 삽입
- 시간소인 및 날짜 데이터 로드
- 현재 시간소인 로드
- 이벤트 큐 관리
- 다른 데이터베이스 소프트웨어와 운영체제로 실행
- 구성요소 대체
- 상품 어드바이저 공간 검색 동기화 사용

다음을 수행하여 로더의 이 기능을 사용자 정의할 수 있습니다.

1. 로더 사용자 정의 특성 파일을 새로 작성하십시오.

   - NT 2000 AIX Solaris Linux

     `MassLoadCustomizer.properties`는 `MassLoader.zip` 아카이브에 위치 지정됩니다. 이 파일을 추출하려면 확장자 `.properties`를 유지한 채 이름을 바꾸고 calsspath의 디렉토리에 위치 지정하십시오.
     중요: 기존 `MassLoadCustomizer.properties` 파일을 제거하거나 수정하지 마십시오.

   - 400

     `ISeries_LODWCSDTA_Customizer.properties`는 `/QIBM/ProdData/WebCommerce/properties` 디렉토리에 위치 지정됩니다. 이 파일을 `/instroot/xml` 디렉토리에 복사하고 확장자 `.properties`를 유지한 채 새 파

일의 이름을 바꿔서 새 파일에 필요한 변경을 작성하십시오.

중요: 원래의 ISeries_LODWCSDTA_Customizer.properties 파일을 제거하거나 수정하지 마십시오.

2. 새 로더 사용자 정의 특성 파일에서 지정된 특성값을 수정하십시오.
3. Load 명령의 사용자 정의 매개변수 값으로 새 파일의 이름을 지정하십시오.

입력 파일의 요소 무시

입력 파일이 대상 데이터베이스에 맵핑하지 않는 요소를 포함하는 경우, 로더 사용자 정의 특성 파일의 이 요소를 로더가 무시하도록 설정할 수 있습니다. IgnoreElements를 사용하여 무시될 요소를 지정하고 세미콜론(;)으로 이 요소를 분리하십시오. 예를 들어 import, literals 및 ProductRepository 요소를 무시하려면 로더 사용자 정의 특성 파일에서 다음을 지정하십시오.

```
IgnoreElements = import;literals;ProductRepository
```

열에 NULL 삽입

로더를 사용하여 로더 사용자 정의 특성 파일에서 EnableNULLCheck 특성을 ″true″로 설정해서 열에 NULL을 삽입할 수 있습니다. 예를 들면, 다음과 같습니다.

```
EnableNULLCheck = true
```

성능상의 이유로 이 특징은 기본값으로 사용 불가능합니다.

데이터 내의 널(Null)값의 문자열 표시를 판별하려면 NULLStringLiteral 특성을 사용하십시오. 예를 들어 문자열 ″-″가 널(Null)값을 표시하는 데 사용되도록 로더를 설정하려면 로더 사용자 정의 특성 파일에서 다음 특성과 값을 지정하십시오.

```
NULLStringLiteral = -
```

기본값으로 특성값은 ″NULL″입니다(인용부호 없이).

시간 소인 및 날짜 데이터 로드

로더는 시간소인과 날짜 데이터 유형으로 열에 데이터를 로드할 수 있습니다. 문서에서 시간소인과 날짜 데이터에 대한 날짜 포맷은 사용자 정의 가능한 패턴에 의해 판별됩니다. 기존 패턴이나 기존 목록에 추가된 패턴을 편집할 수 있습니다.

시간소인이나 날짜에 대한 데이터는 사용 가능한 패턴(마스크)에 반하여 확인됩니다. 데이터에 일치하는 첫 패턴은 데이터베이스에 로드되기 이전의 대상 시간소인 포맷으로 데이터를 변환하는 데 사용됩니다.

두 개의 사용자 정의 가능한 출력 시간소인 패턴, TimeStampFormat.JDBC 및 TimeStampFormat.Load가 있습니다.

1. TimeStampFormat.JDBC는 로더가 조작을 수행하기 위해 JDBC 연결을 사용할 때 사용됩니다.

   로더의 메소드를 반입 및 삭제하는 SQL은 데이터베이스 갱신을 위해 JDBC 연결을 사용합니다.
2. 로더가 원시 유틸리티를 사용할 때 TimeStampFormat.Load가 사용됩니다.

   로더의 로드 메소드 및 반입은 원시 유틸리티를 사용합니다.

로더 사용자 정의 특성 파일에서 마스크를 수정하거나 추가해서 시간소인 포맷을 사용자 정의할 수 있습니다.

다음 입력 시간소인 마스크가 제공됩니다.

```
InputTimeStampFormat.1 = yyyy-DD hh:mm:ss.SSSSSS
InputTimeStampFormat.2 = yyyy-MM-dd hh:mm:ss.SSSSSS
InputTimeStampFormat.3 = yyyy-DD-hh.mm.ss.SSSSSS
InputTimeStampFormat.4 = yyyy-MM-dd-HH.mm.ss.SSSSSS
InputTimeStampFormat.5 = yyyy-MM-dd-hh.mm.ss.SSSSSS
InputTimeStampFormat.6 = yyyy-MM-dd HH:mm:ss.SSSSSS
InputTimeStampFormat.7 = yyyy-DD HH:mm:ss.SSSSSS
```

입력 날짜 포맷에 대한 기본 패턴은 다음과 같습니다.

```
InputDateFormat.1 = MM-dd-yyyy
InputDateFormat.2 = yyyy-dd-MM
InputDateFormat.3 = yyyy-MM-dd
InputDateFormat.4 = MM/dd/yyyy
InputDateFormat.5 = yyyy/dd/MM
InputDateFormat.6 = yyyy-DD
```

시간소인과 날짜 마스크를 수정하거나 원하는 수의 마스크를 추가할 수 있습니다. 입력 시간소인과 비교하고 싶은 수치 순서로 로더 사용자 정의 특성 파일에서 마스크를 지정하십시오. 입력 시간소인을 추가할 경우, 현재 순서 다음 번호를 사용해야 합니다(예를 들어, 위 목록에 추가할 경우 다음 입력 시간소인은 `InputTimeStampFormat.8`입니다).

시간소인과 날짜에 대한 출력에 입력 데이터를 포맷팅하는 패턴은 다음과 같습니다.

```
TimeStampFormat.JDBC = yyyy-MM-dd hh:mm:ss.SSSSSS
TimeStampFormat.Load = yyyy-MM-dd-hh.mm.ss.SSSSSS

DateFormat.JDBC = yyyy-MM-dd
DateFormat.Load = yyyy-MM-dd
```

일반적으로 출력 날짜와 시간소인 포맷은 사용자 정의되지 ***않습니다***.

현재 시간소인 로드

로더는 요일 시계 읽기를 기반으로 하는 시간소인 데이터로 열에 값을 삽입할 수 있습니다. 예를 들어 WebSphere Commerce의 판매 설정 STARTDATE와 ENDDATE는 테이블에 삽입된 판매 설정 시간을 기반으로 하는 값을 가질 수 있습니다. 이것을 기능

적으로 지원하기 위해 로더 패키지는 MLTIME 테이블이 시간소인 인스턴스를 유지하도록 사용합니다. 이 테이블에 대한 스키마는 다음과 같습니다.

```
table MLTIME
  (
  INSTANCEID BIGINT not null,
  MLTIMESTAMP TIMESTAMP
  )
```

로더 사용자 정의 특성 파일에서 다음 특성을 변경해서 테이블 및 해당 열의 이름을 사용자 정의할 수 있습니다.

```
TimestampTableName = MLTIME
TimestampIdColumn = INSTANCEID
TimestampValueColumn = MLTIMESTAMP
```

현재 시간소인 값을 지정하기 위한 입력 데이터는 시간소인 문자열 패턴을 기반으로 합니다. 시간소인의 지속 기간을 지정하기 위해 다음 마스크가 사용됩니다.

```
%D for days
%M for months
%Y for years
%H for hours
%m for minutes
%s for seconds
```

로더 사용자 특성 파일에서 마스크를 수정하거나 추가해서 현재 시간소인 포맷을 사용자 정의할 수 있습니다. 다음 입력 마스크가 제공됩니다.

```
InputCurrentTimestampFormat.1 = CURRENT TIMESTAMP
InputCurrentTimestampFormat.2 = CURRENT TIMESTAMP %D DAYS
InputCurrentTimestampFormat.3 = CURRENT TIMESTAMP %D DAYS %M MONTHS
InputCurrentTimestampFormat.4 = CURRENT TIMESTAMP %D DAYS %M MONTHS %Y YEARS
InputCurrentTimestampFormat.5 = CURRENT TIMESTAMP %Y YEARS %M MONTHS %D DAYS
InputCurrentTimestampFormat.6 = SYSDATE
InputCurrentTimestampFormat.7 = ADDDAYS(SYSDATE,%D)
InputCurrentTimestampFormat.8 = ADDDAYS(ADDMONTHS(SYSDATE,%M),%D)
InputCurrentTimestampFormat.9 = ADDDAYS(ADDMONTHS(ADDYEARS(SYSDATE,%Y),%M),%D)
```

현재 시간소인에 대한 입력 데이터는 지정된 패턴과 일치합니다. 데이터가 지정된 입력 패턴에 일치할 경우, 패턴은 입력 데이터를 구문 분석하는 데 사용되고 로더는 데이터를 데이터베이스에 삽입하기 전에 해당 출력 포맷으로 변환합니다. 새 패턴은 순차적으로 정렬된 첨자 번호를 제공하는 위 목록에 추가될 수 있습니다.

현재 시간소인을 지정하기 위한 두 개의 대상 출력 포맷이 있습니다.

1. CurrentTimestampFormat.Load는 로더가 로드 모드 또는 반출 모드에서 운영 중일 때 사용됩니다.
2. CurrentTimestampFormat.JDBC는 로더가 데이터베이스에서 JDBC를 사용해서 값을 삽입, 갱신 또는 삭제할 때 사용됩니다.

로더에서 기본 대상 패턴은 다음과 같습니다.

```
CurrentTimestampFormat.Load = CURRENT TIMESTAMP %Y YEARS %M MONTHS %D DAYS
  %h HOURS %m MINUTES %s SECONDS
CurrentTimestampFormat.JDBC = CURRENT TIMESTAMP %Y YEARS %M MONTHS %D DAYS
  %h HOURS %m MINUTES %s SECONDS
```

또한 이 패턴에 대한 특성은 로더 사용자 정의 특성 파일에서도 사용자 정의될 수 있습니다. CurrentTimestampFormat.Load와 CurrentTimestampFormat.JDBC 특성을 사용자 정의할 때, 주어진 데이터베이스 관리 시스템에 대한 결과 명령문의 구문이 올바른지 확인해야 합니다.

로더가 CurrentTimestampLiteral 특성을 사용해서 시간소인 열 값이 현재 시간소인 포맷인지 여부를 초기에 판별합니다. 그로 인해 값이 시간소인의 문자열 표시 여부를 판별하기 위한 값비싼 전산처리 비용을 피할 수 있습니다.

```
CurrentTimestampLiteral = CURRENT TIMESTAMP
```

DB2 DB2의 특성에 대한 기본값은 CURRENT TIMESTAMP입니다.

Oracle Oracle 데이터베이스에 대한 기본값은 SYSDATE입니다.

현재 시간소인 로드 예

로더에는 10123의 OFFER_ID로 판매 설정을 갱신하기 위한 아래의 정보가 주어집니다. 시작 날짜는 "CURRENT TIMESTAMP"의 값을 보유하고, 종료 날짜는 "CURRENT TIMESTAMP + 14 DAYS"의 값을 보유합니다.

```
<OFFER
  OFFER_ID="10123"
  STARTDATE="CURRENT TIMESTAMP">
  ENDDATE="CURRENT TIMESTAMP + 14 DAYS"
/>
```

로더는 STARTDATE 및 ENDDATE 열이 데이터베이스에서 시간소인 데이터 유형인지 인식합니다. CurrentTimeStampLiteral 특성에 기반하는 값은 현재 시간소인 포맷에 지정되는 값을 가지는지 여부로 판별됩니다. STARTDATE의 값은 InputCurrentTimeStampFormat.1 패턴에 일치하고 이것은 CurrentTimeStampFormat.JDBC 특성이 지정한 패턴으로 변환됩니다. ENDDATE 값은 InputCurrentTimeStampFormat.2 특성의 포맷에 일치하고 또한 CurrentTimeStampFormat.JDBC 특성이 지정한 패턴으로 변환됩니다.

현재 시간소인에 지속 기간 추가 예

로더는 현재 시간소인에 지속 기간을 추가할 수 있게 합니다. 예를 들어 특정 데이터를 입력하지 않고 판매 설정을 로드해도 됩니다. 그렇게 하려면, 시작 날짜 후의 일부 지속 기간인 종료 날짜를 작성해야 합니다. 다음은 올바른 DB2 작업의 예입니다.

```
<Offer
  Startdate="Current Timestamp"
  Enddate="Current Timestamp +14 Days +4 Months +1 Year +0 Hours
    +0 Minutes +0 Seconds"
/>
```

그러나 플랫폼 독립 방식으로 현재 시간소인 지속 기간을 핸들하려면, 로더 사용자 정의 특성 파일에서 마스크를 수정해서 현재 시간소인 포맷을 사용자 정의해야 합니다. 사용자 정의된 시간소인 특성 스펙의 예입니다.

```
CurrentTimestampLiteral=Current Timestamp

InputCurrentTimestampFormat.0=Current Timestamp
InputCurrentTimestampFormat.1=Current Timestamp %D Days
InputCurrentTimestampFormat.2=Current Timestamp %M Months
InputCurrentTimestampFormat.3=Current Timestamp %Y Years
InputCurrentTimestampFormat.4=Current Timestamp %D Days %M Months
InputCurrentTimestampFormat.5=Current Timestamp %D Days %M Months %Y Years
InputCurrentTimestampFormat.5=Current Timestamp %H Hours %m Minutes %s Seconds

CurrentTimestampFormat.JDBC=Current Timestamp %D Days %M Months %Y Years
  %H Hours %m Minutes %s Seconds
```

판매 설정 예와 이 특성 스펙을 사용하면, 판매 설정에 대한 종료 날짜는 InputCurrentTimestampFormat.5 패턴에 일치합니다. 이는 CurrentTimestampFormat.JDBC를 사용하는 다음 판매 설정 정보를 산출합니다.

```
<Offer
  Startdate="Current Timestamp"
  Enddate="Current Timestamp +14 Days +4 Months +1 Year +0 Hours
  +0 Minutes +0 Seconds"
/>
```

위 예는 로더가 현재 복수 시간소인 포맷을 입력해서 원하는 출력 포맷으로 적절히 포맷할 수 있는 방법을 표시합니다. 다음 예는 플랫폼 독립 포맷을 핸들해서 플랫폼 지정 출력 포맷으로 맵핑할 수 있는 방법을 표시합니다.

```
<Offer
  Startdate="Now"
  Enddate="Now +14D +4M +1Y"
/>

CurrentTimestampLiteral=Now

InputCurrentTimestampFormat.0=Now
InputCurrentTimestampFormat.1=Now %DD
InputCurrentTimestampFormat.2=Now %MM
InputCurrentTimestampFormat.3=Now %YY
InputCurrentTimestampFormat.4=Now %DD %MM
InputCurrentTimestampFormat.5=Now %DD %MM %YY
InputCurrentTimestampFormat.5=Sysdate %HH %mm %ss
```

```
CurrentTimestampFormat.JDBC=AddYears(AddMonths(AddDays(AddHours
(AddMinutes(AddSeconds(Sysdate,%s),%m),%H),%D),%M),%Y)
```

주: 위 명령문은 단지 예에 불과합니다. 이는 가상 데이터베이스 관리 시스템에 대한 사용자 정의 기능을 설명하는 데 사용될 뿐입니다. 이는 DB2나 Oracle 데이터베이스에 대해서는 유효하지 않습니다.

판매 설정 예와 이 특성 스펙을 사용하면, 판매 설정의 종료 날짜는 InputCurrentTimestampFormat.5 패턴과 일치할 것입니다. 이는 CurrentTimestampFormat.JDBC를 사용하는 다음 판매 설정 정보를 산출합니다.

```
<Offer
  Startdate="Current Timestamp"
  Enddate="AddYears(AddMonths(AddDays(AddMinutes(AddSeconds(Sysdate,0)
  ,0),0),14),4),1)"
/>
```

이벤트 큐 관리

로더 사용자 정의 특성 파일의 설정을 수정해서 로더의 이벤트 큐를 사용자 정의할 수 있습니다. 예를 들면, 다음과 같습니다.

```
QueueLowCount = 35
QueueHighCount = 90
```

큐 요소의 수가 최고 한계에 도달하면, 이벤트의 상승을 방해해서 큐를 채우는 이벤트의 소스가 차단됩니다. 큐 요소의 수가 최저 한계 이하에 도달하면, 큐는 다시 이벤트를 받아들이기 시작합니다.

다른 데이터베이스 소프트웨어 및 운영 시스템으로 실행

다른 데이터베이스 소프트웨어와 운영 시스템을 지정하는 로더 사용자 정의 특성 파일의 다음 요소에 대한 매개변수를 수정해서 로더가 다른 데이터베이스 소프트웨어 및 운영체제로 실행하도록 사용자 정의할 수 있습니다.

- 데이터베이스 연결 명령
- 데이터베이스 로드 테이블 명령
- 데이터베이스 반입 명령
- 로드를 호출하는 시스템 명령

이들 항목중 하나를 사용자 정의하려면, 로더 사용자 정의 특성 파일에서 우선하는 명령인 이중 슬래시 명령 문자(//)를 제거하고 기본값을 수정하십시오.

데이터베이스 연결 명령: 기본값 수정을 원할 경우, 데이터베이스 연결 명령의 매개변수를 변경할 수 있습니다(DB2 경우).

```
DBConnectCommand = connect to {0} user {1} using {2};
```

여기서

0 = 데이터베이스 이름
1 = 데이터베이스 사용자
2 = 사용자 암호

데이터베이스 로드 테이블 명령: 기본값 수정을 원할 경우, 데이터베이스 로드 테이블 명령의 매개변수를 변경할 수 있습니다.

```
DBLoadTableCommand = load from {0} of del modified by coldel{1}
chardel{2} insert into {3} ({4});
```

여기서

0 = 파일 이름
1 = 열 분리문자
2 = 문자 분리문자
3 = 테이블 이름
4 = 쉼표(,)로 분리된 열 이름

데이터베이스 반입 명령: 기본값 수정을 원할 경우, 데이터베이스 반입 명령의 매개변수를 변경할 수 있습니다.

```
DBImportCommand = import from {0} of del modified by coldel{1} chardel{2}
insert_update into {3} ({4});
```

여기서

0 = 파일 이름
1 = 열 분리문자
2 = 문자 분리문자
3 = 테이블 이름
4 = 쉼표(,)로 분리된 열 이름

로드를 호출하는 시스템 명령: 기본값 수정을 원할 경우, 로드를 호출하는 시스템 명령의 매개변수를 변경할 수 있습니다. 이 명령은 원시 로드를 실행하고 로더가 생성한 스크립트를 반출합니다.

- DB2

  ```
  DBLoadCommand = db2clpex DB2 -z {0} -astvf {1}
  ```

  여기서

0 = 로그 파일 이름
1 = 명령 파일 이름

예를 들어, AIX에서 DB2를 실행하려면 DBLoadCommand 특성값이 다음과 같습니다.

```
db2 -tvf {1} -z {0}
```

- AIX Solaris Linux Oracle

```
DBLoadCommand = sqlldr log={0} control={1} USERID={2}
```

여기서

0 = 로그 파일 이름
1 = 명령 파일 이름
2 = 데이터베이스 사용자 이름

다양한 데이터베이스 및 운영체제 조합에는 다음 설정을 사용하십시오.

- NT 2000 DB2

  Windows NT 또는 Windows 2000에서 sqlimport, load, import 또는 delete 메소드로 실행 중인 DB2에서 시스템 환경 변수 중 classpath가 db2/dbconnect.zip을 포함하도록 설정하십시오.

- AIX Solaris Linux DB2

  AIX, Solaris 또는 Linux 환경에서 실행 중인 DB2에서 다음을 수행하십시오.

  – sqlimport, load, import 또는 delete 메소드로 시스템 환경 변수 중 classpath가 db2/dbconnect.zip을 포함하도록 설정하십시오.

  – load 또는 import 메소드로 로더 사용자 정의 특성 파일에서 다음 특성을 수정하십시오.

```
/**
  * Connection command. (Default is for DB2)
  * parameter 0 = dbName
  * parameter 1 = dbUser
  * parameter 2 = userPasswd
*/

DBConnectCommand = connect to {0} user {1} using {2};

/**
  * Load Data into Table command. (Default is for DB2)
  * parameter 0 = filename
  * parameter 1 = column delimiter
  * parameter 2 = character delimiter
  * parameter 3 = name of the table
  * parameter 4 = name of the columns, separated by comma(,)s
*/
```

```
DBLoadTableCommand = load from {0} of del modified by coldel{1}
insert into {3} ({4});

/**
  * Insert Data into Table command. (Default is for DB2)
  * parameter 0 = filename
  * parameter 1 = column delimiter
  * parameter 2 = character delimiter
  * parameter 3 = name of the table
  * parameter 4 = name of the columns, separated by comma(,)s
*/

DBUpdateTableCommand = import from {0} of del modified by coldel{1}
insert_update into {3} ({4});

/**
  * System command to invoke load (Default is for DB2)
  * parameter 0 = logFileName
  * parameter 1 = commandFileName
*/

DBLoadCommand = db2 -z {0} -tf {1}
```

- 400 DB2 sqlimport, load, import 또는 deletesql 메소드로 iSerise에서 실행 중인 DB2에 대한 로더 사용자 특성 파일에서 다음 특성을 수정하십시오.

```
/**
  * The connect string.
*/

ConnectStringID = jdbc:db2://

/**
  * The JDBC driver information.
*/

JDBCDriverName = com.ibm.db2.jdbc.app.DB2Driver
DbVendorName=DB2/iSeries

/**
  * Custom writer for load/import methods.
*/

WriterName=com.ibm.wca.MassLoader.Writer.ISeriesWriter
```

- NT 2000 AIX Solaris Linux Oracle

  Windows NT, Windows 2000, AIX, Solaris 또는 Linux 환경에서 sqlimport, load, import 또는 delete 메소드로 실행 중인 Oracle 데이터베이스에서 시스템 환경 변수 중 classpath가 `oracle/dbconnect.zip`을 포함하도록 설정하십시오.

구성요소 대체

로더 사용자 정의 특성 파일의 다음 구성요소에 기본 구현을 대체하고 싶은 클래스 값을 부여해서 로더 구성요소를 대체할 수 있습니다.

ParserName

사용될 구문 분석기의 이름

ValidatorName

사용될 유효성 확인자의 이름

FormatterName

사용될 포맷터의 이름

JDBCFormatterName

SQL import 메소드가 사용될 때의 포맷터 이름

WriterName

사용될 writer의 이름

JDBCWriterName

SQL import 메소드가 사용될 때의 writer 이름

예를 들어 로더의 기본값 writer(DefaultWriter)를 com.abc.writer.SpecialWriter writer로 대체하려면 로더 특성 파일에서 다음을 지정하십시오.

```
WriterName = com.abc.writer.SpecialWriter
```

로더는 작성 기능을 수행하기 위해 "com.abc.writer.SpecialWriter"를 사용합니다.

상품 어드바이저 공간 검색 동기화 사용

상품 어드바이저 공간 검색 동기화를 사용하려면 다음을 수행하십시오.

1. "PASyncInfo.xml"로 명명된 동기화에 대해 XML 구성 정보 파일을 작성하십시오.
2. PASyncInfo.xml에서 PASync.xsd를 사용될 XML 스키마로 지정하십시오. 예를 들면, 다음과 같습니다.

   ```
   <PASync
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation='PASync.xsd'
   ```

   PASync.xsd 파일이 제공됩니다. 다음 텍스트는 PASync.xsd의 컨텐츠를 표시합니다.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'>

     <xsd:element name="PASync">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="SearchScheme" />
          <xsd:element ref="Command" minOccurs="1" maxOccurs="unbounded" />
        </xsd:sequence>
        <xsd:attribute name = "member" type="xsd:string" use="required" />
        <xsd:attribute name = "store" type="xsd:string" use="required" />
      </xsd:complexType>
     </xsd:element>

     <xsd:element name="SearchScheme">
   ```

```
 <xsd:complexType>
   <xsd:sequence>
      <xsd:element ref="RelatedTable" />
        <xsd:element ref="Search" minOccurs="1" maxOccurs="unbounded" />
   </xsd:sequence>
   <xsd:attribute name = "tableName" type="xsd:string" use="required" />
   <xsd:attribute name = "primary" type="xsd:string" use="required" />
   <xsd:attribute name = "colName" type="xsd:string" use="required" />
 </xsd:complexType>
</xsd:element>

<xsd:element name="RelatedTable">
 <xsd:complexType>
   <xsd:attribute name = "tableName" type="xsd:string" use="required" />
   <xsd:attribute name = "from" type="xsd:string" use="required" />
   <xsd:attribute name = "to" type="xsd:string" use="required" />
 </xsd:complexType>
</xsd:element>

<xsd:element name="Search">
 <xsd:complexType>
   <xsd:attribute name = "value" type="xsd:string" use="required" />
 </xsd:complexType>
</xsd:element>

<xsd:element name="Command">
 <xsd:complexType>
   <xsd:attribute name = "tableName" type="xsd:string" use="required" />
   <xsd:attribute name = "idColumnName" type="xsd:string" use="required" />
   <xsd:attribute name = "addCommand" type="xsd:string" />
   <xsd:attribute name = "updateCommand" type="xsd:string" />
   <xsd:attribute name = "deleteCommand" type="xsd:string" />
 </xsd:complexType>
</xsd:element>

</xsd:schema>
```

3. PASyncInfo.xml에서 완료될 필요가 있는 동기화에 대한 구성원 ID 및 상점 엔티티 ID를 지정하십시오. 예를 들면, 다음과 같습니다.

```
member = "-2000"
store = "10351"
```

4. PASyncInfo.xml의 검색 스키마 요소 아래에 공간 검색을 구성하는 CATGROUP 식별자를 지정하십시오. 예를 들면, 다음과 같습니다.

```
<SearchScheme
  tableName = "catgroup"
  primary = "CATGROUP_ID"
  colName = "identifier" >

  <RelatedTable
    tableName = "catgpenrel"
    from = "CATGROUP_ID"
    to = "CATENTRY_ID" />

  <Search value="Pants" />
  <Search value="Shirts" />

</SearchScheme>
```

예에 ″Pants″와 ″Shirts″가 지정됩니다. 원하는 수량의 CATGROUP 식별자를 지정할 수 있습니다.

5. PASyncInfo.xml에서 계획할 명령을 판별할 속성을 지정하십시오. 예를 들면, 다음과 같습니다.

```
<Command tableName = "CATENTRY" idColumnName = "CATENTRY_ID"
  updateCommand = "UpdateSearchSpaces"
  deleteCommand = "RemoveProductsFromAllSearchSpaces"
/>

<Command tableName = "CATENTDESC" idColumnName = "CATENTRY_ID"
  addCommand = "UpdateSearchSpaces"
  updateCommand = "UpdateSearchSpaces"
  deleteCommand = "UpdateSearchSpaces"
/>

<Command tableName = "LISTPRICE" idColumnName = "CATENTRY_ID"
  addCommand = "UpdateSearchSpaces"
  updateCommand = "UpdateSearchSpaces"
  deleteCommand = "UpdateSearchSpaces"
/>

<Command tableName = "ATTRVALUE" idColumnName = "CATENTRY_ID"
  addCommand = "UpdateSearchSpaces"
  updateCommand = "UpdateSearchSpaces"
  deleteCommand = "UpdateSearchSpaces"
/>

<Command tableName = "CATENTATTR" idColumnName = "CATENTRY_ID"
  addCommand = "UpdateSearchSpaces"
  updateCommand = "UpdateSearchSpaces"
  deleteCommand = "UpdateSearchSpaces"
/>

<Command tableName = "CATGPENREL" idColumnName = "CATENTRY_ID"
  addCommand = "AddProductsToSearchSpace"
  deleteCommand = "RemoveProductsFromSearchSpace"
/>

</PASync>
```

6. 새 로더 사용자 정의 특성 파일에서 XML 구성 정보 파일을 지정하십시오. 예를 들면, 다음과 같습니다.

```
PASyncDocumentURL = PASyncInfo.xml
```

7. 새 로더 사용자 정의 특성 파일에서 동기화를 사용으로 설정하십시오. 예를 들면, 다음과 같습니다.

```
PASyncEnabled = true
```

8. 로드 명령으로 USQL import 또는 delete 메소드 둘 중 하나를 사용하십시오. 다음은 로더에 대한 적절한 XML 입력의 예입니다.

```
<store-asset>

   <catentry
       CATENTRY_ID="10351"
       MEMBER_ID="-2000"
       PARTNUMBER="000051"
       CATENTTYPE_ID="ProductBean"
       MFPARTNUMBER="m000051"
```

```
    MARKFORDELETE="0"
    BUYABLE="1"
  />

<catentry
    CATENTRY_ID="10352"
    MEMBER_ID="-2000"
    PARTNUMBER="000052"
    CATENTTYPE_ID="ProductBean"
    MFPARTNUMBER="m000052"
    MARKFORDELETE="0"
    BUYABLE="1"
  />

<catentry
    CATENTRY_ID="10353"
    MEMBER_ID="-2000"
    PARTNUMBER="000053"
    CATENTTYPE_ID="ProductBean"
    MFPARTNUMBER="m000053"
    MARKFORDELETE="0"
    BUYABLE="1"
  />

<catentry
    CATENTRY_ID="10358"
    MEMBER_ID="-2000"
    PARTNUMBER="000058"
    CATENTTYPE_ID="ProductBean"
    MFPARTNUMBER="m000058"
    MARKFORDELETE="0"
    BUYABLE="1"
  />

<catentry
    CATENTRY_ID="10365"
    MEMBER_ID="-2000"
    PARTNUMBER="000065"
    CATENTTYPE_ID="ProductBean"
    MFPARTNUMBER="m000065"
    MARKFORDELETE="0"
    BUYABLE="1"
  />

<catentry
    CATENTRY_ID="10372"
    MEMBER_ID="-2000"
    PARTNUMBER="000072"
    CATENTTYPE_ID="ProductBean"
    MFPARTNUMBER="m000072"
    MARKFORDELETE="0"
    BUYABLE="1"
  />

  <catgpenrel
    CATGROUP_ID="10354"
    CATENTRY_ID="10372"
    CATALOG_ID="10351"
```

```
      SEQUENCE="3"
    />

    <catgpenrel
      CATGROUP_ID="10354"
      CATENTRY_ID="10365"
      CATALOG_ID="10351"
      SEQUENCE="4"
    />

    <catgpenrel
      CATGROUP_ID="10354"
      CATENTRY_ID="10358"
      CATALOG_ID="10351"
      SEQUENCE="5"
    />

    <catgpenrel
      CATGROUP_ID="10355"
      CATENTRY_ID="10372"
      CATALOG_ID="10351"
      SEQUENCE="3"
    />

  </store-asset>
```

주: 상품 어드바이저 공간 검색 동기화를 사용하지 않으면 더 나은 로더 성능을 제공합니다. 따라서 이 기능은 필요한 경우에만 사용하십시오.

상품 어드바이저 공간 검색 동기화 사용자 정의

로더 패키지는 다음을 수행하기 위해 로더 사용자 정의 특성 파일을 수정해서 상품 어드바이저 공간 검색 동기화를 사용자 정의할 수 있게 합니다.

- **동기화를 사용 또는 사용 안함으로 설정**

  로더 사용자 정의 특성 파일의 다음 특성값으로 true나 false를 지정해서 동기화를 사용 또는 사용하지 않을 수 있습니다.

  ```
  PASyncEnabled = true
  ```

- **동기화에 대한 구성 정보 파일을 지정**

  로더 사용자 정의 특성 파일의 다음 특성에 대한 값을 설정해서 동기화가 사용하는 XML 구성 정보 파일을 지정할 수 있습니다.

  ```
  PASyncDocumentURL = PASyncInfo.xml
  ```

- **스케줄 조회 길이 지정**

  로더 사용자 정의 특성 파일에서 다음 특성값을 설정해서 스케줄 조회 길이를 지정할 수 있습니다.

  ```
  PAScheduleQueryLength = 30
  ```

  이 특성값은 20 ~ 900의 범위 내에 있어야 합니다.

- **계획된 시작 시간 지정**

  로더 사용자 정의 특성 파일의 PAScheduledStartTime 특성값으로 절대 시간소인, 현재 시간소인, 또는 지속 기간을 지닌 현재 시간소인을 지정해서 계획된 시작 시간을 지정할 수 있습니다.

  주: 시간소인의 포맷은 데이터베이스에 대해 적합해야 합니다.

  다음은 로드 후 5분의 계획된 스케줄을 실행할 DB2에 대한 예입니다.

  ```
  PAScheduledStartTime = CURRENT TIMESTAMP + 5 MINUTES
  ```

  Oracle 다음은 즉시 작업을 실행할 Oracle 데이터베이스에 대한 예입니다.

  ```
  PAScheduledStartTime = SYSDATE
  ```

로더 사용 시의 데이터 처리 메소드 판별

로더는 로드 명령을 사용하는 데이터 처리에 관한 다음 옵션을 제공합니다.

- 로드
- 반입
- SQL 반입 기능 사용

데이터를 로드하기 전에 최상의 결과를 가져올 처리 메소드를 판별해야 합니다.

load 메소드 선택

다음 경우에 load 메소드를 고려하십시오.

- 데이터가 정리되었음을 알고 있고 데이터베이스에 어떠한 데이터도 포함되어 있지 않은 경우
- 데이터가 정리되었고 데이터베이스에 로드되고 있는 데이터가 포함되어 있지 않음을 알고 있는 경우
- 데이터가 정리되었음을 알고 있고 대상 테이블에 어떠한 1차 키도 포함되지 않을 경우 및 데이터베이스에 로드되고 있는 데이터가 포함되어 있지 않음을 알고 있는 경우
- 로드 시간이 주요 관심사항일 경우
- 데이터베이스가 로컬 DB2 데이터베이스인 경우

400 load 메소드로 데이터는 데이터베이스에 로드됩니다. 데이터가 이미 존재하면 명령은 duplicate-key 오류 및 duplicate-error 메시지가 표시하는 결과로 실패합니다.

import 메소드 선택

NT 2000 AIX Solaris Linux DB2용 import 메소드로, 데이터가 데이터베이스에 로드됩니다. 데이터가 이미 존재할 경우, 삭제되지 않고 새 값으로 갱신됩니다. 다음 경우에 이 메소드를 고려하십시오.

- 데이터베이스 관리 시스템이 DB2인 경우
- 데이터가 정리되었는지 알 수 없는 경우
- 열 레벨에서 동질인 데이터의 포괄적인 설정을 갱신해야 하는 경우
- 로드 시간이 주요 관심사항이 아닌 경우
- 데이터가 반입된 테이블이 1차 키를 가질 경우

400 import 메소드로 데이터가 데이터베이스에 로드됩니다. 데이터가 이미 존재할 경우, 삭제되지 않고 새 값으로 갱신됩니다. 다음 경우에 이 메소드를 고려하십시오.

- 데이터가 정리되었는지 알 수 없는 경우
- 데이터베이스에 데이터가 이미 존재할 경우
- 로드 시간이 주요 관심사항이 아닌 경우
- 데이터가 반입된 테이블이 1차 키를 가질 경우

SQL import 메소드 선택

SQL import 메소드로 데이터베이스에 데이터를 갱신 또는 삽입하는 데 JDBC 및 SQL 문이 사용됩니다. 데이터가 존재하지 않을 경우 데이터는 삽입되고 기존 데이터는 갱신됩니다. 다음 경우에 이 메소드를 고려하십시오.

- 기존 데이터를 갱신 중이고 열 레벨 갱신이 필요한 경우

  이 메소드로 제한자 위반 및 데이터 유형 오류에 대한 더 나은 오류가 보고됩니다.
- 데이터의 일부가 정리되지 않은 경우
- 데이터베이스 무결성이 주요 관심사항일 경우
- 데이터베이스가 로컬이 아닌 경우
- 상품 어드바이저 공간 검색 동기화를 사용 중인 경우

기타 고려사항

- **load 메소드 사용 제한**

  load 메소드는 비트 데이터 필드에서 데이터를 삽입 또는 갱신할 수 없습니다.

  DB2 load 메소드로 새 레코드만 데이터베이스에 삽입되고 기존 레코드는 갱신되지 않습니다.

DB2 load 메소드는 원격이 아닌 로컬 DB2 데이터베이스용으로만 사용됩니다.

- **import 메소드 사용 제한**

  데이터베이스 관리 시스템은 import 메소드를 사용하기 위해 DB2이어야 합니다.

  import 메소드는 비트 데이터 필드에서 데이터를 삽입이나 갱신할 수 없습니다.

  import 메소드로 로더는 정의된 1차 키를 가지는 테이블을 삽입 및 갱신합니다. 반입 방법은 1차 키를 가지지 않은 테이블에서 삽입 및 갱신을 할 수 없습니다. 출력 레코드는 기본 열에 대해서만 값을 가질 경우 레코드는 거부됩니다.

- **SQL import 및 load 메소드 비교**

  SQL import 메소드는 foreign 참조를 포함하는 데이터 일관성을 확인하고 기존 데이터를 갱신하게 합니다. load 메소드는 그렇지 않습니다.

- **import 및 SQL import 메소드 비교**

  import 및 SQL import 메소드는 유사한 기능을 수행합니다. import 메소드는 보통 더 빠르지만 임시 파일에 대한 디스크 공간을 요구합니다.

  import 메소드는 정의된 1차 키를 가지는 테이블을 삽입하고 갱신할 뿐입니다. 반면에 SQL import 메소드는 테이블이 1차 키를 가질 것을 요구하지 않습니다.

- **사용된 데이터베이스 상품에 기반한 메소드 비교**

  import 및 load 메소드는 DB2에 최적화된 원시 유틸리티를 사용하는 반면, SQL import 메소드는 JDBC 호출을 사용합니다(많은 데이터베이스 제품에 일반적임).

- **추가 고려사항**

  delete 메소드는 데이터베이스에서 입력 XML 문서의 데이터를 삭제하는 데 사용됩니다. 요소는 테이블에 대한 1차 키 또는 고유 색인의 값을 포함해야 합니다. 삭제된 데이터가 "계단식 삭제" 사용으로 다른 테이블의 데이터에 종속되는 경우, 종속 데이터 또한 삭제됩니다.

  상품 어드바이저 공간 검색 동기화를 사용 중인 경우, 데이터 로딩을 위해 SQL import 메소드를 사용해야 합니다.

큰 문서 로드

로더 패키지 유틸리티를 사용하여 큰 문서를 데이터베이스에 로드할 때, 다음 항목을 고려하십시오.

- **JVM(Java Virtual Machine) 힙 크기**

  JVM에 할당된 메모리 최대량의 기본값은 64MB입니다. 증가하지 않을 경우, JVM은 결국 로드를 처리하는 동안 메모리가 부족하게 됩니다. Java 힙에 대한 메모리 최대 할당량은 Java 명령어 중 JVM-mx 옵션 사용으로 달라질 수 있습니다.

- **로그 작성 추적**

  큰 XML 문서를 로드할 때 추적 로거(logger)는 JVM 힙을 소모할 수 있습니다. 실행이 실패할 경우, 정보 추적은 대부분 실행을 디버깅하는 데 사용됩니다. 로드 처리

추적이 필요하지 않을 경우, 추적을 꺼야 합니다. 추적을 끄면 상당한 성능 이익이 있습니다. 로그 작성 구성 XML 문서를 수정해서 추적을 끕니다.

기본 로그 작성 구성 파일은 WCALoggerConfig.xml입니다. 추적 로그 작성을 끄려면, 로더에 대한 추적 로거(logger) 구성을

```
<logger type="trace">
   <handler type="file">
      <filePath>MassLoadTrace.log</filePath>
 <filter type="Any">
          <messageType name="PUBLIC" />
      </filter>
   </handler>
</logger>
```

에서 다음과 같이 변경하십시오.

```
<logger type="trace">
   <handler type="file">
      <filePath>MassLoadTrace.log</filePath>
 <filter type="Any">
      </filter>
   </handler>
</logger>
```

WCALoggerConfig.xml 파일을 수정에 대한 추가 정보는 72 페이지의 『로더 패키지에 대한 로그 작성 사용자 정의』를 참조하십시오.

- **확약 계수**

  로더가 SQL 입력 모드에서 운영 중일 때, 로더에 대한 기본 확약 계수는 1입니다. 따라서 기본값으로 데이터베이스로의 모든 삽입과 갱신에 대해 트랜잭션이 확약됩니다. 큰 문서에 대한 로더의 성능을 개선하려면, 확약 계수가 증가되어야 합니다. 값 ″100″이 제안되나 이는 DBMS 트랜잭션 로그의 크기, 서버의 실제 메모리의 양 등에 따라 더 커질 수 있습니다.

  로더에 대한 확약 계수는 로드 명령(여기서 *count*는 확약된 트랜잭션 이전에 실행된 명령문의 수)에 대한 -commitcount *count* 옵션을 사용하여 변경됩니다.

문제점 해결 추가정보

데이터를 로드하는 중 진행이 현저하게 느리면, 이는 로더에 대한 로거(logger)가 올바르게 구성되지 않은 파일 핸들러를 가진 경우일 수 있습니다. 다음 원인 중 하나일 수 있습니다.

- 로더를 호출하는 사용자에게 디렉토리 쓰기 또는 로그 작성 구성 문서에 지정된 파일의 갱신이 허용되지 않습니다.
- 로그 작성 구성 문서의 파일 위치로 지정된 디렉토리가 존재하지 않습니다.
- 로그 작성 구성 문서의 파일 위치로 지정된 드라이브가 충분한 공간을 가지지 않습니다.

이와 같은 문제를 수정할 때, 로그 작성 구성 문서를 수정해서 파일의 위치 지정을 변경할 필요가 있을 수 있습니다(기본값은WCALoggerConfig.xml). 파일 핸들러 및 WCALoggerConfig.xml 파일에 대한 추가 정보는 72 페이지의 『로더 패키지에 대한 로그 작성 사용자 정의』를 참조하십시오.

제 7 장 데이터 추출

추출기를 사용하여 데이터베이스에서 데이터를 추출하려면 추출 필터 파일을 사용해서 데이터베이스에서 추출하려는 데이터를 지정해야 합니다. 사용하는 추출 필터는 추출을 원하는 데이터의 유형에 따라 결정됩니다.

다음은 추출 필터로 MemberSubsystemFilter.xml을 사용해서 데이터베이스에서 구성원 서브시스템을 추출하는 예입니다.

- NT 2000 AIX Solaris Linux

```
java com.ibm.wca.MassExtract.Extract -filter MemberSubsystemFilter.xml
-outfile MemberSubsystemExtracted.xml -dbname mall -dbuser myname
-dbpwd mypassword -customizer MemberSubsystemCustomizer
```

- 400

```
QWEBCOMM/EXTWCSDTA FILTER(MemberSubsystemFilter.xml)
OUTFILE(MemberSubsystemExtracted.xml) DATABASE(database_name)
SCHEMA(mall) INSTROOT(/QIBM/UserData/WebCommerce/instances/mser)
PASSWD(mypassword)
```

추출 필터 작성

CATGROUP, CATGRPDESC, CATGRPREL, CATENTRY, CATENTSHIP, OFFER, CATENTREL, CATGPENREL, CATENTDESC 및 ATTRVALUE 테이블에서 카테고리와 상품 정보를 추출하는 추출 필터의 예를 들어, 다음과 같습니다.

```
<sqlx>

    <!-- ************************** -->
    <!-- extract Category information -->
    <!-- ************************** -->

    <functionDef id="Category" description="Extract Categories" schemaentity="catgroup">
      <paramDef name=":lastRecord" type="string" value="10301" description="Last record
       before loading new data" />
      <body>
        select * from catgroup where catgroup_id > :lastRecord
      </body>
    </functionDef>

    <execute id="Category" description="Extract Categories" schemaentity="catgroup">
      <param name=":lastRecord" type="string" value="10300" description="Last record
       before loading new data" />
    </execute>

    <functionDef id="Category Description" description="Extract Category Descriptions
     for a Locale" schemaentity="catgrpdesc">
      <paramDef name=":lastRecord" type="string" value="10300" description="Last record
       before loading new data" />
      <body>
        select * from catgrpdesc where catgroup_id > :lastRecord
      </body>
    </functionDef>
```

```
<execute id="Category Description" description="Extract Category Descriptions
 for a Locale" schemaentity="catgrpdesc">
  <param name=":lastRecord" type="string" value="10300" description="Last record
   before loading new data" />
</execute>

<functionDef id="Category Relationship" description="Extract Category-Relations
 for a Locale" schemaentity="catgrprel">
  <paramDef name=":lastRecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catgrprel where catgroup_id_child > :lastRecord
  </body>
</functionDef>

<execute id="Category Relationship" description="Extract Category-Relations for
 a Locale" schemaentity="catgrprel">
  <param name=":lastRecord" type="string" value="10300" description="Last record
   before loading new data" />
</execute>

<!-- ************************** -->
<!-- extract Product information -->
<!-- ************************** -->

<functionDef id="Product" description="Extract Product" schemaentity="catentry">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catentry where catentry_id > :lastrecord
  </body>
</functionDef>

<execute id="Product" description="Extract Product" schemaentity="catentry">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
</execute>

<functionDef id="Product Relationship" description="Extract Product Ship
 information" schemaentity="catentrel">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catentrel where catentry_id_child > :lastrecord
  </body>
</functionDef>

<execute id="Product Relationship" description="Extract Product Ship information"
 schemaentity="catentrel">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
</execute>

<functionDef id="Product Description" description="Extract Product Description"
 schemaentity="catentdesc">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catentdesc where catentry_id > :lastrecord
  </body>
</functionDef>

<execute id="Product Description" description="Extract Product Description"
 schemaentity="catentdesc">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
</execute>

<functionDef id="Product Ship" description="Extract Product Ship information"
```

```
 schemaentity="catentship">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catentship where catentry_id > :lastrecord
  </body>
 </functionDef>

 <execute id="Product Ship" description="Extract Product Ship information"
  schemaentity="catentship">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
 </execute>

 <functionDef id="Category Product Relationship" description="Extract Category
  Product Relations" schemaentity="catgpenrel">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from catgpenrel where catgroup_id > :lastrecord
  </body>
 </functionDef>

 <execute id="Category Product Relationship" description="Extract Category Product
  Relations" schemaentity="catgpenrel">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
 </execute>

 <!-- ************************************* -->
 <!-- Extract Product Attribute Information    -->
 <!-- ************************************* -->

 <functionDef id="Product Attribute Values" description="Extract Product Attribute
  values for a Locale" schemaentity="attrvalue">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from attrvalue where catentry_id > :lastrecord
  </body>
 </functionDef>

 <execute id="Product Attribute Values" description="Extract Product Attribute values
  for a Locale" schemaentity="attrvalue">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
 </execute>

 <!-- ********************************************** -->
 <!-- Extract Product Price Information                -->
 <!-- ********************************************** -->

 <functionDef id="Offer" description="Extract Offer" schemaentity="offer">
  <paramDef name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
  <body>
    select * from offer where catentry_id > :lastrecord
  </body>
 </functionDef>

 <execute id="Offer" description="Extract Offer" schemaentity="offer">
  <param name=":lastrecord" type="string" value="10300" description="Last record
   before loading new data" />
 </execute>

</sqlx>
```

추출기 설정

추출기가 사용하는 데이터베이스 드라이버를 변경하려면 다음을 수행하십시오.

1. 추출기 사용자 정의 특성 파일을 새로 작성하십시오.
   - NT 2000 AIX Solaris Linux

     DB2ConnectionCustomizer.properties는 MassExtract.zip 아카이브에 위치합니다. 이 파일을 추출하려면 확장자 .properties를 유지한 채 이름을 바꾸고 calsspath의 디렉토리에 위치 지정하십시오.

     중요: 기존 DB2ConnectionCustomizer.properties 파일을 제거하거나 수정하지 마십시오.
   - 400

     ISeries_EXTWCSDTA_Customizer.properties는 /QIBM/ProdData/WebCommerce/properties 디렉토리에 위치합니다. 이 파일을 /*instroot*/xml 디렉토리에 복사하고 확장자 .properties를 유지한 채 새 파일의 이름을 바꿔서 새 파일에 필요한 변경을 작성하십시오.

     중요: 원래의 ISeries_EXTWCSDTA_Customizer.properties 파일을 제거하거나 수정하지 마십시오.
2. 새 파일에서 database-driver 값을 수정하십시오.
3. 추출 명령의 사용자 정의된 매개변수 값으로 새 파일의 이름을 지정하십시오.

다음은 추출기 사용자 정의 특성 파일에서 인용한 것입니다.

```
DBVendorName = DB2
DBDriverName = COM.ibm.db2.jdbc.app.DB2Driver
DBURL = jdbc:db2:
```

여기서

- DBVendorName은 데이터베이스의 유형을 선택하는 데 사용됩니다.

  옵션은 다음과 같습니다.
  - iSeries(DB2/iSeries)용 DB2 Universal Database
  - 기타 운영체제(DB2)용 DB2
  - Oracle 데이터베이스(Oracle)
- DBDriverName은 JDBC 드라이버를 선택하는 데 사용됩니다.

  옵션은 다음과 같습니다.
  - iSeries(com.ibm.db2.jdbc.app.DB2Driver)용 DB2 Universal Database
  - 기타 운영체제(COM.ibm.db2.jdbc.app.DB2Driver)용 DB2
  - Oracle 데이터베이스(oracle.jdbc.driver.OracleDriver)
- DBURL은 데이터베이스에 액세스할 URL을 지정하는 데 사용됩니다.

옵션은 다음과 같습니다.

– iSeries(`jdbc:db2://`)용 DB2 Universal Database
– 기타 운영체제(`jdbc:db2:`)용 DB2
– Oracle 데이터베이스(`jdbc:oracle:oci8:@`)

제 8 장 로더 패키지 로거(logger) 사용

로더 패키지의 개별 유틸리티는 프로그램 추적 정보를 제공하는 메시지 뿐 아니라 성공, 실패 및 오류를 표시하는 메시지를 작성합니다.

로더 패키지의 유틸리티는 WCALoggerConfig.xml 파일을 참조합니다.

- NT 2000 AIX Solaris Linux

  이 파일은 시스템 환경 변수 중 classpath에 지정된 디렉토리에 존재합니다. 또한 com.ibm.wca.logging.configFile Java 시스템 특성이 이 파일을 지정할 수 있습니다.

- 400

  이 파일은 /*instroot*/xml 디렉토리에 위치 지정됩니다.

WCALoggerConfig.xml은 개별 유틸리티가 제공하는 로그 작성 정보 및 정보의 표시 또는 저장 위치를 판별합니다. 이 파일을 사용자 정의할 수 있으며 로그된 메시지의 유형 및 작성된 로그 유형을 지정할 수 있습니다.

Windows NT, Windows 2000, AIX, Linux 및 Solaris 시스템 환경에서 로그 작성 구성

사용자 환경에서 로그 작성을 설정하려면 classpath 시스템 환경 변수가 WCALoggerConfig.xml 파일을 포함하거나 com.ibm.wca.logging.configFile 시스템 특성을 지정하도록 설정해야 합니다.

classpath 변수 설정의 예

Windows NT 시스템의 d:\WebSphere\CommerceServer\xml\loader 디렉토리에 WCALoggerConfig.xml 파일이 있는 경우, 다음 명령문을 사용해서 classpath 변수를 설정할 수 있습니다.

```
SET CLASSPATH=%CLASSPATH%;D:\WebSphere\CommerceServer\xml\loader
```

com.ibm.wca.logging.configFile 시스템 특성 지정의 예

com.ibm.wca.logging.configFile 시스템 특성을 지정하려면 Java 해석기를 호출할 때 -D 옵션을 사용하십시오. 예는 다음과 같습니다.

```
java -Dcom.ibm.wca.logging.configFile=D:\ice_tea\src\classlib\logger\xml\WC.xml
com.ibm.wca.DTDGenerator.GenerateDTD
```

로더 패키지에 대한 로그 작성 사용자 정의

로더 패키지에 대한 로그 작성을 사용자 정의하려면 WCALoggerConfig.xml 파일을 사용하십시오.

- NT 2000 AIX Solaris Linux

  이 파일은 시스템 환경 변수 중 classpath에 지정된 디렉토리에 존재합니다. 또한 com.ibm.wca.logging.configFile Java 시스템 특성이 이 파일을 지정할 수 있습니다.

- 400

  이 파일은 /*instroot*/xml 디렉토리에 위치 지정됩니다.

WCALoggerConfig.xml은 하나 이상의 구성요소 태그를 포함합니다(예: <component name="DTDGenerator">). 이 개별 태그 내에 로거(logger)와 핸들러를 추가할 수 있습니다. 시스템과 함께 제공된 유틸리티와 로거(logger) 태그를 변경하면 *안됩니다*. 그러나 로거(logger)에 핸들러 태그를 추가할 수는 있습니다. 이 파일에 포함할 수 있는 것에 대한 정보는 WCALogger.dtd 파일을 참조하십시오.

로더 패키지 로그는 다음 디렉토리의 messages.txt 파일에 위치합니다.

- NT *drive*:\WebSphere\CommerceServer\instances*instance_name*\logs
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\instances*instance_name*\logs
- AIX /usr/WebSphere/CommerceServer/instances/*instance_name*/logs
- Solaris Linux /opt/WebSphere/CommerceServer/instances/*instance_name*/logs
- 400 /QIBM/UserData/WebCommerce/instances/*instance_name*/logs

핸들러

로거(logger)에 핸들러를 추가하려면 WCALoggerConfig.xml 파일에 핸들러 유형을 지정하십시오. 로거(logger)에 하나 이상의 핸들러를 추가할 수 있습니다. 개별 핸들러는 해당 속성을 가지며 다른 핸들러에 적용될 필요 없는 종속 태그를 가짐에 유의하십시오. 핸들러 유형은 다음을 포함합니다.

핸들러 유형	설명 및 속성
콘솔	일반적으로 명령 행인 표준 출력에 메시지를 보내십시오.
파일	텍스트 파일 내의 상점 메시지 종속 태그처럼 이 핸들러에 "<filePath>log path</filePath>"를 추가해야 합니다.
복수 파일	파일의 순환 로그를 작성하십시오. "<filePath>log path</filePath>"를 지정해야 합니다. *n* 전체에 로그 파일이 1이 작성됩니다. 다음 속성을 추가할 수 있습니다. **MaxFiles** 첫 로그 파일이 지워지기 전에 사용할 로그 파일 수를 표시하는 정수 **MaxKBFileSize** 개별 로그 파일에 지정할 최대 KB 수를 표시하는 정수
NT 2000 AIX Solaris Linux 데이터베이스	순환 로그의 DB2 테이블의 상점 메시지. 다음 속성을 추가할 수 있습니다. **brand** 데이터베이스 브랜드 이름. DB2가 현재 지원하는 유일한 데이터베이스입니다. **maxRows** 가장 오래된 항목을 지우기 전에 테이블에 저장된 레코드의 최대 수 종속 태그처럼 "<jdbc/>"를 포함할 수 있으며 다음 속성을 포함합니다. **url** 데이터베이스(예: "Jdbc:db2:"*wcm*"", 여기서 *"wcm"*은 데이터베이스 이름)에 액세스하는 JDBC에 사용된 URL. 데이터베이스는 유틸리티를 실행하기 전에 존재해야 합니다. **table** 메시지가 로그될 데이터베이스 테이블의 이름. 다음 DB2 명령문으로 작성되어야 합니다. "CREATE TABLE"*tablename* (KEY char(13) FOR BIT DATA NOT NULL, COMPONENTNAME VARCHAR(30), ENTRY VARCHAR(2000), PRIMARY KEY(key))"" **userid** 데이터베이스 사용자 이름. 테이블 갱신 권한이 지정되어야 합니다. 다음 DB2 명령문이 이를 실행합니다. "GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE"*tablename* TO USER *userid*"" **password** 지정된 사용자 이름에 대한 데이터베이스 암호

다음은 로거(logger)에 "데이터베이스" 유형의 핸들러를 추가하는 예입니다.

```
<handler type="database" brand="DB2" maxRows="50">
 <jdbc url="jdbc:db2:wcm"
```

```
    table="wcm.log"
    userid= "wasuser"
    password="123456"/>
 <filter type="Any">
   <messageType name="FATAL"/>
   <messageType name="ERROR"/>
       <messageType name="WARNING"/>
 </filter>
</handler>
```

필터

메시지 유형을 포함 및 제공하는 핸들러에서 필터를 추가할 수 있거나 제거할 수 있습니다. 로거(logger)에 필터가 없을 경우, 메시지가 로그되지 않습니다. 개별 필터 태그는 일반적으로 다음 중 하나인 메시지 유형을 나열하는 종속 messageType 태그를 가집니다.

- INFO
- ERROR
- FATAL
- WARNING

다른 메시지 유형은 WCALogger.dtd 파일에 나열되나 일반적으로 대부분은 로더 패키지와 함께 사용되지 않습니다.

필터 유형은 다음을 포함합니다.

필터 유형	설명 및 속성
Any	지정된 messageType 유형 중 하나로 플래그되는 모든 메시지를 로그 파일에 포함합니다. 예를 들어 messageType 목록이 ERROR를 포함하고 응용프로그램이 ERROR 유형 메시지를 생성하는 경우, 메시지가 로그됩니다.
All	지정된 모든 messageType 유형 속성이 로그에 포함되기 전에 메시지가 이를 가져야 합니다.
Exclude	messageType 태그의 목록에 지정되지 않은 모든 메시지를 로그합니다.

다음 예와 같이 핸들러에 필터를 추가하려면 ERROR 및 FATAL 메시지 유형을 로그하고 다른 메시지는 무시합니다.

```
<handler type="file">
 <filter type="Any">
  <messageType name="FATAL"/>
  <messageType name="ERROR"/>
   </filter>
</handler>
```

포맷

메시지 포맷팅에 대한 두 포맷터 중의 하나를 지정할 수 있습니다.

포맷터 유형	설명 및 속성
safe(기본값)	메시지가 특성 파일에서 발견되지 않을 경우, 예외 설정을 방해합니다. 이 포맷터는 자원 누락을 표시하는 메시지를 작성합니다.
xml	XML 포맷에서 메시지를 포맷하십시오. 메시지가 발견될 수 없을 경우, 이 포맷터는 또한 예외를 설정하는 대신 메시지를 작성합니다.

예: *WCALoggerConfig.xml* 및 *WCALogger.dtd*

WCALoggerConfig.xml

```
 <?xml version="1.0" standalone="no"?>
 <!DOCTYPE WCALoggerConfig SYSTEM "WCALogger.dtd">
 <WCALoggerConfig>
 <component name="MassLoader">
  <logger type="message">
  <handler type="file">
    <filePath>c:\temp\out.txt</filePath>
<filter type="Any">
  <messageType name="FATAL"/>
  <messageType name="ERROR"/>
      <messageType name="WARNING"/>
      <messageType name="INFO"/>
     </filter>
  </handler>
  </logger>
  <logger type="trace">
  <handler type="file">
    <filePath>out2.txt</filePath>
<filter type="Any">
      <messageType name="PUBLIC"/>
     </filter>
  </handler>
  </logger>
  <logger type="typedMessage">
  <handler type="file">
    <filePath>tout.txt</filePath>
<filter type="Any">
  <messageType name="FATAL"/>
  <messageType name="ERROR"/>
      <messageType name="WARNING"/>
      <messageType name="INFO"/>
     </filter>
  </handler>
  </logger>
  <logger type="progress">
   <handler type="console" format="safe">
<filter type="Any">
  <messageType name="FATAL"/>
```

```
  <messageType name="ERROR"/>
      <messageType name="WARNING"/>
      <messageType name="INFO"/>
    </filter>
  </handler>
  </logger>
 </component>
 <component name="DTDGenerator">
  <logger type="message">
   <handler type="console">
<filter type="Any">
  <messageType name="FATAL"/>
  <messageType name="ERROR"/>
      <messageType name="WARNING"/>
      <messageType name="INFO"/>
    </filter>
  </handler>
  </logger>
  <logger type="trace">
   <handler type="console">
<filter type="Any">
  <messageType name="FATAL"/>
  <messageType name="ERROR"/>
      <messageType name="WARNING"/>
      <messageType name="INFO"/>
    </filter>
  </handler>
  </logger>
 </component>
 </WCALoggerConfig>
```

WCALogger.dtd

```
<!-- This DTD describes how a WCALoggerConfig XML can be structured.
A WCALoggerConfig XML document is the input configuration file for
the WCALoggerFactory class.
-->

<!ELEMENT WCALoggerConfig (component)+>

<!ELEMENT component (logger)+>
<!ATTLIST component name CDATA #REQUIRED>
<!ELEMENT logger (handler+,messageFile?)>
<!ATTLIST logger type (message | trace | typedMessage | progress) "typedMessage">

<!-- messageFile is an optional default properties files that can be used to
make messages locale specific
-->
<!ELEMENT messageFile (#PCDATA)>
<!ELEMENT handler (filePath?, filter, jdbc?)>
<!ATTLIST handler
type ( file|multiFile|console|error|textArea|database|ejbQueue|queue ) "console">

<!-- maxFiles & maxKBFileSize only applies to the multiFile type of handler
-->
<!-- filePath & encoding applies only when the handler is of type file or
multiFile
-->
<!ATTLIST handler maxFiles CDATA #IMPLIED>
<!ATTLIST handler maxKBFileSize CDATA #IMPLIED>
<!ATTLIST handler encoding CDATA #IMPLIED>
```

```
<!ATTLIST handler format (safe | xml) "safe">
<!-- maxRecords & brand are only applicable to database handler type
-->
<!ATTLIST handler maxRecords CDATA #IMPLIED>
<!ATTLIST handler brand (DB2) #IMPLIED>
<!-- the jdbc tag must be present within a database handler type tag
-->
<!ELEMENT jdbc EMPTY>
<!ATTLIST jdbc url CDATA #IMPLIED>
<!ATTLIST jdbc table CDATA #IMPLIED>
<!ATTLIST jdbc userid CDATA #IMPLIED>
<!ATTLIST jdbc password CDATA #IMPLIED>

<!ELEMENT filter (messageType+)>
<!ATTLIST filter type (Any | All | Exclude ) "Any">

<!-- the messageType attribute name is one of these JLog IRecordType
constants
-->
<!ELEMENT messageType EMPTY>
<!ATTLIST messageType name ( NONE | ALL | INFO |
INFORMATION | WARN | WARNING | ERR | ERROR |
FATAL | DEFAULT_MESSAGE | API | CALLBACK |
ENTRY_EXIT | ENTRY | EXIT | ERROR_EXC |
MISC_DATA | OBJ_CREATE | OBJ_DELETE |
PRIVATE | PUBLIC | STATIC | SVC | PERF |
LEVEL1 | LEVEL2 | LEVEL3 ) "ALL">
<!ELEMENT filePath (#PCDATA)>
```

제 9 장 로더 패키지 오류 보고서 사용

오류가 있을 경우, 로더 및 ID 분석기는 예외 문서를 생성하는 오류 보고서를 포함합니다.

기본값으로 예외 문서는 다음 디렉토리에 쓰여집니다.

- NT 2000 AIX Solaris Linux 입력 문서가 있는 디렉토리
- 400 */instroot*/logs

예외 문서가 쓰여진 디렉토리를 지정하려면 Java 특성 com.ibm.wcm.ErrorReporterDir을 사용하십시오. Windows NT 환경에서 로더에 대한 예는 다음과 같이 시작합니다.

```
java -Dcom.ibm.wcm.ErrorReporterDir=d:\massloaderrors
com.ibm.wca.MassLoader.MassLoad -dbname . . .
```

주: 사용자는 지정된 디렉토리에 쓸 수 있는 권한을 가져야 합니다.

다음은 오류 보고서에 대한 DTD(store-all-error.dtd) 견본입니다.

```
<!ENTITY % TABLE "calrule | catentry">
<!ELEMENT store-asset (error, (%TABLE;)*)>
<!ELEMENT message (#PCDATA) >
<!ELEMENT error ( message ) >
<!ATTLIST error
  locus             CDATA     #REQUIRED
  id                CDATA     #REQUIRED
>
<!ELEMENT calrule (error)>
<!ATTLIST calrule
  identifier        CDATA     #REQUIRED
  calrule_id        CDATA     #REQUIRED
  calcode_id        CDATA     #REQUIRED
  startdate         CDATA     #IMPLIED
  taxcgry_id        CDATA     #IMPLIED
  enddate           CDATA     #IMPLIED
  sequence          CDATA     #REQUIRED
  combination       CDATA     #REQUIRED
  calmethod_id      CDATA     #REQUIRED
  calmethod_id_qfy  CDATA     #REQUIRED
  flags             CDATA     #REQUIRED
  field1            CDATA     #IMPLIED
  field2            CDATA     #IMPLIED
>
<!ELEMENT catentry (error)>
<!ATTLIST catentry
  catentry_id       CDATA     #REQUIRED
  member_id         CDATA     #REQUIRED
  catenttype_id     CDATA     #REQUIRED
  partnumber        CDATA     #IMPLIED
  mfpartnumber      CDATA     #IMPLIED
```

```
  mfname            CDATA     #IMPLIED
  markfordelete     CDATA     #REQUIRED
  url               CDATA     #IMPLIED
  field1            CDATA     #IMPLIED
  field2            CDATA     #IMPLIED
  lastupdate        CDATA     #IMPLIED
  field3            CDATA     #IMPLIED
  onspecial         CDATA     #IMPLIED
  onauction         CDATA     #IMPLIED
  field4            CDATA     #IMPLIED
  field5            CDATA     #IMPLIED
  buyable           CDATA     #IMPLIED
>
```

다음은 로더에서의 오류 보고 문서 견본입니다.

```
<?xml version="1.0"?>
<!DOCTYPE store-asset SYSTEM "store-all-error.dtd">
<store-asset>
      <error
      locus="Parser"
      id="SAXParseFatalError" >
         <message>
         Error The string "--" is not permitted within comments. : 155 : 18
         </message>
      </error>
   <calrule
      calcode_id="30"
      enddate="2100-01 10:20:30.000000"
      calmethod_id="-47"
      identifier="7"
      taxcgry_id="9"
      calmethod_id_qfy="-46"
      startdate="1900-01-01-00.00.00.000000"
      flags="1"
      combination="2"
      calrule_id="44"
      sequence="9.0E+1">
      <error
         locus="Writer"
         id="SQLException" >
         <message>
            A SQL Exception was received [IBM][CLI Driver][DB2/NT] SQL0530N
            The insert or update value of the FOREIGN KEY
            "JANTONY.CALRULE.F_CALRULE4" is not equal to any value of the
            parent key of the parent table.  SQLSTATE=23503
         </message>
      </error>
   </calrule>
   <catentry
      catentry_id="10118"
      member_id="-2001"
      partnumber="1254"
      mfpartnumber="sku-163"
      mfname="InFashion"
      markfordelete="0"
      buyable="1"
      field1="abc" >
      <error
```

```
            locus="Formatter"
            id="FormattingError" >
            <message>
               Error when formatting value for CATENTRY.FIELD1 : abc with error
               [class java.lang.NumberFormatException(abc)].
            </message>
         </error>
      </catentry>
   </store-asset>
```

제 10 장 로더 패키지 명령 및 스크립트 구성

로더 패키지를 실행해서 해당 명령을 수행하려면, WebSphere Commerce 디렉토리에 제공된 스크립트나 명령을 사용하십시오.

- NT *drive*:\WebSphere\CommerceServer\bin
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\bin
- AIX /usr/WebSphere/CommerceServer/bin
- Solaris Linux /opt/WebSphere/CommerceServer/bin
- 400 QWEBCOMM native library

스크립트와 명령은 다음과 같습니다.

400

GENWCSDTD
DTD 생성 명령

RESWCSID
ID 분석 명령

EXTWCSDTA
추출 명령

LODWCSDTA
로드 명령

TRNWCSTXT
텍스트 변환 명령

TRNWCSXML
XML 변환 명령

NT 2000

dtdgen.cmd
DTD 생성 명령

idresgen.cmd
ID 분석 명령

massextract.cmd
추출 명령

massload.cmd

로드 명령

txttransform.cmd

텍스트 변환 명령

xmltransform.cmd

XML 변환 명령

AIX Solaris Linux

dtdgen.sh

DTD 생성 쉘 스크립트

idresgen.sh

ID 분석 쉘 스크립트

massextract.sh

추출 쉘 스크립트

massload.sh

로드 쉘 스크립트

txttransform.sh

텍스트 변환 쉘 스크립트

xmltransform.sh

XML 변환 쉘 스크립트

제 3 부 웹 편집기 사용

이 부분은 Catalog Manager 웹 편집기를 관리하고 사용하는 방법을 설명합니다.

웹 편집기는 웹 브라우저를 통한 카탈로그 데이터의 작성, 삭제 및 변경을 가능하게 합니다. 정보 보기 및 갱신을 위한 데이터 항목 양식이 웹 편집기의 중앙입니다. 가장 단순한 경우, WebSphere Commerce 데이터베이스의 테이블에 양식이 일치합니다. 운영자는 제공된 기본 양식을 사용을 선택하거나 가능한 양식을 사용자 정의할 수 있습니다.

주: 웹 편집기는 Internet Explorer 5 이상을 사용합니다.

제 11 장 웹 편집기 설정

운영자는 WebSphere Commerce 스키마에 대한 확장자와 사용자 정의를 포함해서 데이터를 WebSphere Commerce에 넘겨주도록 웹 편집기를 구성할 수 있습니다. 웹 편집기는 WebSphere Commerce의 특정 인스턴스에 고유한 웹 항목의 세트로 디자인되지 않았습니다. 웹 편집기는 다양한 조직의 개별 요구 및 역할을 지원하기 위해 탄력적으로 사용자 정의되도록 디자인되었습니다.

데이터베이스 보기는 데이터베이스의 하나 또는 그 이상의 테이블에 저장된 조회입니다. Catalog Manager 설치 중, WebSphere Commerce 상품의 논리적 보기를 작성하는 견본 파일은 다음 디렉토리의 각 시스템별 서브디렉토리(db2, oracle 및 os400)에 위치합니다.

- NT *drive*:\WebSphere\CommerceServer\schema
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\schema
- AIX /usr/WebSphere/CommerceServer/schema
- Solaris Linux /opt/WebSphere/CommerceServer/schema
- 400 /QIBM/ProdData/WebCommerce/schema

wcs.view.sql 파일은 복수 테이블에서 상품에 관한 정보를 결합하기 위해 작성된 상품 보기를 포함합니다. 이는 상품 보기의 SQL 데이터 정의를 포함합니다. 데이터베이스 자체의 보기 개발을 계획하기 위해 운영자가 이 파일을 연구할 수 있습니다.

기본 XML 양식 설명 파일(forms51_be.xml)은 WebSphere Commerce 데이터베이스에서 데이터를 추가, 편집 및 삭제를 용이하게 하도록 디자인된 양식을 포함합니다. 이 파일의 사본은 다음 디렉토리에 위치 지정됩니다.

- NT *drive*:\WebSphere\CommerceServer\xml\wcwebeditor\xml
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\xml\wcwebeditor\xml
- AIX /usr/WebSphere/CommerceServer/xml/wcwebeditor/xml
- Solaris Linux /opt/WebSphere/CommerceServer/xml/wcwebeditor/xml
- 400 /*instroot*/xml/wcwebeditor/xml

웹 편집기를 설정하는 운영자는이 구성 파일을 있는 그대로 사용하거나 구성 파일을 변경 및 강화할 수 있습니다. XML 양식 설명 파일을 사용자 정의하려면 이 절의 다음에 포함된 지시사항을 참조하십시오.

주: DTD 작성기는 웹 편집기가 사용하는 양식을 자동으로 작성할 수 있습니다.

웹 편집기 구성

이 절은 웹 편집기를 구성하는 방법에 관한 정보를 제공합니다. 설치 과정이 초기에 이 구성을 핸들하더라도 운영자는 다른 데이터베이스를 사용하여 웹 편집기를 다시 구성하는 등의 작업에 이 정보를 사용할 수 있습니다.

webeditor.properties 파일 편집

웹 편집기는 webeditor.properties 파일 내에 설정된 특정 응용프로그램 매개변수를 가집니다. 이것은 다음 디렉토리에 위치합니다.

- NT *drive*:\WebSphere\AppServer\installedApps\WC_Enterprise_App_demo.ear\wcwebeditor.war\WEB-INF\classes\webeditor.properties
- 2000 *drive*:\Program Files\WebSphere\AppServer\installedApps\WC_Enterprise_App_demo.ear\wcwebeditor.war\WEB-INF\classes\webeditor.properties
- AIX /usr/WebSphere/AppServer/installedApps/WC_Enterprise_App_demo.ear/wcwebeditor.war/WEB-INF/classes/webeditor.properties
- Solaris Linux /opt/WebSphere/AppServer/installedApps/WC_Enterprise_App_demo.ear/wcwebeditor.war/WEB-INF/classes/webeditor.properties
- 400 /QIBM/UserData/WEBASADV4/*was_instance_name*/installedApps/WC_Enterprise_App_*wcs_instance_name*.ear/wcwebeditor.war/WEN-INF/classes

webeditor.properties 파일을 편집해서 운영자는 웹 편집기에 표시된 양식을 설명하기 위해 사용된 파일을 변경하는 등의 작업을 수행할 수 있습니다. webeditor.properties 파일의 컨텐츠의 예입니다.

```
# Properties file for WebEditor

(The following specifies where the customized process list and the Catalog Manager
utility configuration envelopes are located.)
# URI Location of Process and WCM Subsystem configuration envelopes
ProcessConfigFile=file:///D:/WebSphere/CommerceServer/xml/wcwebeditor/xml
/weProcessList.xml

(The following specifies where the forms are defined.)
# Location of Forms file
FormsURL=file:///D:/WebSphere/CommerceServer/xml/wcwebeditor/xml/forms51_be.xml

(The following setting specifies where the XML-to-HTML style sheet is located)
# Location of XML to HTML StyleSheet
StyleSheetURI=file:///D:/WebSphere/CommerceServer/xml/wcwebeditor/xsl/webeditor.xsl
```

```
(The following setting specifies the location for temporary files)
# Location of Temporary Directory
#temp.dir=

# WebSphere datasource. This is used to build form drop-down lists and field
# default values. However, when publishing and searching, the Web Editor utilizes
# the WCM subsystems ID Resolver, Mass Loader, and Mass Extractor. Database access
# for these WCM subsystems must be configured separately.
# Name of WAS database source
dbsource=jdbc/WebSphere Commerce DB2 DataSource demo

# Name of WAS database. If specified, this takes preference over the database name
# value in the forms XML file.  This value will be utilized as a parameter when
# invoking the WCM subsystems such as the Mass Loader.
dbname=mall

(The following setting specifies the character set to use.  By default, it is
set to the single-byte character set.  For other countries, choose from the values
that are commented out.)
# Encoding Character Set
Encoding=ISO-8859-1

# CN
#Encoding=gb2312
# TW
#Encoding=Big5
# KR
#Encoding=EUC-KR
# JP
#Encoding=Shift_JIS

(The following setting is used to allow images to be previewed.  The hostname of
the server plus any leading directory information before the image information that
is stored in WebSphere Commerce should be specified here.  The WebSphere Commerce
catalog information is appended to this value to construct an image URL.)
# Specifies the base href location for images
# This should be set so that this info plus the info stored in WCS
# (e.g. /image/char.gif) combines to create a URL to an image

#imageRootURL=http://%HOSTDOMAIN
imageRootURL=http://localHost/webeditor

(The following setting is used to set a date format for the application.  If these
properties are not available (i.e., they are commented out), then the Java-locale
specific format will be used.  Below is a reference table for setting these values.)
# Use these properties to specify a date format if the Java locale-specific
# format is not desired

#dateFormat=yyyy-MM-dd
#dateTimeFormat=yyyy-MM-dd HH:mm:ss

# Symbol Meaning               Presentation             Example
# ------ -------               ------------             -------
# G      era designator        (Text)                   AD
# y      year                  (Number)                 1996
# M      month in year         (Text & Number)          July & 07
# d      day in month          (Number)                 10
# h      hour in am/pm         (1~12) (Number)          12
# H      hour in day           (0~23) (Number)          0
# m      minute in hour        (Number)                 30
# s      second in minute      (Number)                 55
# S      millisecond           (Number)                 978
# E      day in week           (Text)                   Tuesday
# D      day in year           (Number)                 189
# F      day of week in month  (Number)                 2 (2nd Wed in July)
```

```
# w      week in year          (Number)                27
# W      week in month         (Number)                2
# a      am/pm marker          (Text)                  PM
# k      hour in day           (1~24) (Number)         24
# K      hour in am/pm         (0~11) (Number)         0
# z      time zone             (Text)                  Pacific Standard Time
# '      escape for text       (Delimiter)
# ''     single quote          (Literal)               '
```

임시 파일의 위치 변경

webeditor.properties 파일의 temp.dir 특성에 대한 값을 추가하고 설명 표시를 제거해서 임시 파일의 위치를 변경할 수 있습니다.

선택적으로 java.io.tmpdir Java 특성이 임시 파일의 작성 위치를 판별하는 데 사용됩니다.

DTD 작성기를 사용하여 XML 양식 설명 파일 작성

forms51_be.xml 파일은 XML 양식 설명 파일의 예입니다. 이는 웹 편집기가 사용하는 유형의 세트를 제공합니다. 이 파일의 사본은 다음 디렉토리에 위치 지정됩니다.

- NT *drive*:\WebSphere\CommerceServer\xml\wcwebeditor\xml
- 2000 *drive*:\Program Files\WebSphere\CommerceServer\xml\wcwebeditor\xml
- AIX /usr/WebSphere/CommerceServer/xml/wcwebeditor/xml
- Solaris Linux /opt/WebSphere/CommerceServer/xml/wcwebeditor/xml
- 400 /*instroot*/xml/wcwebeditor/xml

다음 단계는 시스템 운영자가 새 XML 양식을 추가하기 위해 DTD 작성기를 사용하는 방법을 설명합니다.

주: 아래에 설명된 프로시저를 수행하기 전에 웹 편집기 디렉토리의 기존 forms51_be.xml 이름을 바꾸기하십시오. 다른 방법으로는 새 이름으로 다음 프로시저를 수행하는 동안 출력 파일을 작성한 후, 새로 작성된 파일을 사용하도록 웹 편집기를 다시 구성하십시오. 이에 대한 지시사항은 다음 절을 참조하십시오.

XML 양식을 작성하려면 DTD 생성 명령을 실행하십시오.

다음 단계는 XML 양식을 작성하는 방법을 설명합니다.

1. 유형에서 사용하려는 테이블의 이름이 포함된 "tables.txt"로 명명된 임시 파일을 작성하십시오.

   다음 예와 같이 단일 행에 테이블 각각의 이름을 입력하십시오.

```
catentry
catentdesc
catentship
inventory
```

2. DTD 생성 명령에 위치 지정된 디렉토리에 tables.txt를 저장하십시오(이 명령의 설치 위치는 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에서 참조).
3. 운영 시스템 명령 프롬프트에서 DTD 생성 명령이 위치 지정된 디렉토리를 변경하십시오.
4. 다음을 입력하여 DTD 생성 명령을 실행하십시오.

   NT 2000

   ```
   dtdgen -infile tables.txt -outfile tables51.dtd
   -dbname dbname -dbuser userid -dbpwd password
   -xmlTableDesc tableFORMS.xml -schemaname schema -propfile filename
   ```

   AIX Solaris Linux

   ```
   ./dtdgen.sh -infile tables.txt -outfile tables51.dtd
   -dbname dbname -dbuser userid -dbpwd password
   -xmlTableDesc tableFORMS.xml -schemaname schema -propfile filename
   ```

   400

   ```
   QWEBCOMM/GENWCSDTD DATABASE(database) SCHEMA(schema)
   INSTROOT(instroot) PASSWD(password) OUTFILE(tables51.dtd)
   INFILE(tables.txt) XMLTABDESC(tableFORMS.xml)
   ```

   테이블 설명 스위치(-xmlTableDesc 또는 XMLTABDESC)는 DTD에 추가된 테이블의 새 양식 설명을 작성하는 DTD 작성기를 불러옵니다.

   NT 2000 AIX Solaris Linux -propFile 옵션은 도움말 텍스트, 기본값 및 필드 설명 정보가 저장된 외부 특성 파일의 옵션을 지정합니다.
5. 이전 절에 설명된 대로 새로 작성된 파일을 사용하려면 웹 편집기를 다시 구성하십시오.
6. WebSphere 고급 관리 콘솔에서 웹 편집기를 다시 시작하십시오. 이를 위해 다음 단계를 수행하십시오.
   a. **WebSphere 관리 도메인**을 펼치십시오.
   b. **엔터프라이즈 응용프로그램**을 펼치십시오.
   c. **WebSphere Commerce 엔터프라이즈 응용프로그램 - demo**를 마우스 오른쪽 버튼으로 누르고 **중지**를 선택하십시오.
   d. 응용프로그램이 중지됐다는 메시지가 표시될 때까지 기다리십시오.
   e. **WebSphere Commerce 엔터프라이즈 응용프로그램 - demo**를 마우스 오른쪽 버튼으로 누르고 **시작**을 선택하십시오.

f. 엔터프라이즈 응용프로그램이 시작됐다는 메시지가 표시될 때까지 기다리십시오.

7. 웹 브라우저에서 새 양식을 보려면 다음 URL을 여십시오.

   ```
   https://host_name:8000/wcm/webeditor
   ```

   여기서 *host_name*은 WebSphere Application Server의 완전한 HTTP 호스트 이름입니다.

   웹 편집기는 모든 테이블 이름의 목록으로 브라우저 창에서 표시됩니다.

XML 양식 설명 사용자 정의

이 절은 운영자가 웹 편집기가 표시하는 양식을 강화하는 방법에 대해서 설명합니다.

XML 양식 설명 파일 자체 분리 또는 특성 파일 둘 중 하나에서 속성과 값을 설정해서 XML 양식 설명을 사용자 정의 및 향상시킬 수 있습니다.

주: 특성 파일의 이름은 formList 태그의 resourcePackage 속성 값으로 지정되어야 합니다. 파일 이름이 클래스 경로의 디렉토리의 서브디렉토리에 나타날 경우, 이는 패키지(dot) 스펙을 사용해야 합니다.

다음 테이블은 운영자가 교체할 수 있는 웹 편집기 양식 필드 속성을 나열합니다.

필드 속성	설명
Currency	로케일 고유 수치 분리 문자(미국에서 천 단위 분리 문자로 쉼표를 사용하는 것처럼)로 값을 표시하게 합니다.
DbColumn	특성 파일 키에 필드 이름을 맵핑하는 데 사용됩니다. 로케일 고유 특성 파일이 사용되고 있을 경우, 이 항목값이 특성 파일에 입력된 것에 일치해야 합니다. DTD 작성기는 이 항목에 스키마를 추가합니다.

DefaultValue	사용자가 새 유형을 기입할 때 데이터 항목에 표시되는 값을 지정합니다.
	
	DTD 작성기는 데이터베이스 기본값에 이 속성을 설정할 수 있습니다. 이는 정적 문자열일 수 있지만 단일 열 테이블에 반하는 SQL 스칼라 조회를 포함할 수도 있습니다. 외부 데이터를 조회하려면 사용자 정의된 함수를 조회에 사용할 수 있습니다. 예를 들면, 다음과 같습니다.
	
	`DefaultValue="SELECT CURRENT TIMETAMP FROM EXEC"`
	
	여기서 EXEC는 정의되어 다음과 같이 대량 자료 반입됩니다.
	
	`CREATE TABLE EXEC (A CHAR(1));`
	`INSERT INTO A VALUES('A');`

dynamicSqlSelectionList	드롭 다운 메뉴가 개별 유형을 다시 작성하게 합니다.

필드 속성	설명
FieldDescription	유형의 항목 필드 다음에 표시되는 설명을 제공하십시오. 이 속성을 작성할 때 일부 설명이 있을 경우, DTD 작성기는 열의 설명을 사용합니다. 열에 설명이 없는 경우, 기본값은 열 이름입니다. 이 속성은 로케일 고유 특성 파일 또는 XML 양식 설명 파일에서 설정될 수 있습니다. 특성 파일에 값이 지정된 경우, 이는 다른 것에 우선합니다.

필드 속성	설명
FieldHelp	필드가 양식에 집중될 때 브라우저의 맨 아래 메시지 바에 표시되는 필드의 간단한 도움말 설명을 제공합니다. 기본값으로 데이터의 열 유형과 함께 주어지는 필드에 대해 데이터를 입력하는 단순한 메시지를 포함합니다. 이 속성은 로케일 고유 특성 파일 또는 XML 양식 설명 파일에서 설정될 수 있습니다. 특성 파일에 값이 지정된 경우, 이는 다른 것에 우선합니다.
formatNumber	웹 편집기가 어떤 방법으로든 번호를 처리하지 않도록 알리는 데 사용됩니다. 조회(여기서 값은 문자열이 될 인용구에 지정되지 않음)를 추출하는 동안을 제외하고는 문자열로 입력된 값을 다루기 위해 formatNumber 속성을 ″false″로 설정하십시오. 이 속성의 기본값은 ″true″입니다.
Hidden	값이 양식에 표시되지 않지만 여전히 HTML의 숨겨진 필드로 사용 가능함을 표시합니다.
HideOnCreate	새 유형이 작성될 때 필드가 사용 가능함을 표시합니다. showInCreateMode=″false″와 비슷하지만 숨겨진 유형으로 필드 이름을 추가합니다.
Maxlength	데이터베이스 열의 길이를 지정합니다. 이것은 데이터베이스에 저장될 수 있는 길이보다 큰 값을 입력하지 않도록 보증하는 데 사용됩니다.
MinOccurs	필드가 필수적인지 여부를 표시합니다. ″1″ 값은 필수를, ″0″ 값은 선택적임을 의미합니다.
Name	데이터베이스 열에 이름을 지정합니다.

필드 속성	설명
readOnly readOnlyForCreate readOnlyForEdit	필드의 편집용으로 사용 가능한 시기 및 필드가 읽기 전용인 시기를 제어합니다. readOnly=″true″는 필드가 항상 읽기 전용 모드임을 의미합니다. readOnlyForCreate=″true″는 새 양식이 작성되고 있을 때 필드가 읽기 전용 모드임을 의미합니다. readOnlyForEdit=″true″는 기존 유형이 편집되고 있을 때 필드가 읽기 전용 모드임을 의미합니다.
ShowColumnInList	필드가 ″true″로 설정될 때, 데이터의 복수 레코드 보기를 구성하는 열의 하나인 필드를 지정합니다. DTD 작성기는 처음 6열을 ″true″로 설정합니다.

필드 속성	설명
showInCreateMode	작성된 양식에 필드를 숨기는 데 사용됩니다. 표시되는 작성된 양식에 필드를 숨기기 위해 showInCreateMode 속성을 ″false″로 설정합니다. 이 속성의 기본값은 ″true″입니다.
ShowInSearchMode	검색 기준 페이지에서 특정 열을 숨기는데 사용됩니다. 이 속성이 ″false″로 설정된 경우, 검색 양식에 주어진 필드가 표시되지 않습니다.
SqlSelectionList	새 양식을 작성할 때 선택할 드롭 다운 메뉴를 작성합니다. 조회는 하나 또는 두 개의 열 결과 설정에 리턴해야 합니다. 첫 번째 열은 사용자가 선택할 수 있는 레이블 목록이며 두 번째 열은 데이터베이스 테이블에 저장되는 실제 값 목록입니다. 단지 한 개의 열이 사용될 경우, 표시된 값은 테이블에 저장됩니다. 이 기능은 foreign-key 관계를 향상시키는 데 상당한 도움이 됩니다.
Type	데이터베이스 열의 유형 및 데이터에서 수행해야 할 유효성 확인의 종류를 표시합니다. 예를 들어 유형이 ″integer″일 경우, 응용프로그램은 올바른 정수만이 이 필드에 입력되었음을 보증합니다.
UniqueKey	필드가 테이블에 1차 키임을 표시합니다. 이 속성이 ″true″로 설정될 경우, 응용프로그램은 이 열에 입력된 데이터에 고유 제한자를 강화합니다. 데이터베이스의 레코드에 확장하지 않을 경우, 이 확인은 응용프로그램에 로드된 레코드에서만 수행됩니다.
ValidateInput	유효성 확인을 끄는 데 사용됩니다. 이 속성은 ID 분석기나 XML 변환기가 핸들할 수치 필드에 텍스트를 입력하게 합니다.

양식 이름 편집

XML 양식 설명의 양식 태그는 이름과 displayName 속성 두 개 모두를 가집니다.

- 이름 속성은 데이터베이스 테이블의 이름 또는 이것이 표시하는 보기로 설정되어야 합니다.
- displayName 속성은 웹 에디터 응용프로그램에서 easy-to-use 이름을 제공합니다. 다른 방법으로는 양식에 대한 displayName 속성이 formList 태그의 resourcePackage 속성에 지정된 특성 파일에 설정될 수 있습니다(resourcePackage 속성은 다중 언어 지원용으로 사용됩니다).

  주: 등록 특성 파일에서 항목이 나타날 경우, 이것은 XML 양식 설명 파일에서 입력된 것에 우선합니다.

필드 설명 변경

다음은 XML 양식 설명 파일의 필드 설명의 예입니다.

```
<form name = "CATALOG.CATENTRY"
  .
  .
  <field name="MEMBER_ID"
  showInCreateMode="false"
    fieldDescription="MEMBER_ID"
    type="integer"
    maxlength="19"
    defaultValue=""
    .
    .
<field name="CATENTTYPE_ID"
    fieldDescription="CATENTTYPE_ID"
    type="string"
    maxlength="16"
    defaultValue=""
    .
    .
```

다음과 같이 변경할 수도 있습니다.

```
<form name = "CATALOG.CATENTRY"
  displayName="Product"
  .
  .
  <field name="MEMBER_ID"
  showInCreateMode="false"
  fieldDescription="Member Identifier"
    type="integer"
    maxlength="19"
    defaultValue=""
    .
    .
<field name="CATENTTYPE_ID"
    fieldDescription="Product Type"
    type="string"
```

```
    maxlength="16"
    defaultValue=""
    .
    .
```

드롭 다운 메뉴 추가

드롭 다운 메뉴의 양식에 있는 선택 목록을 웹 편집기 양식에 추가해서 유효한 선택 세트에서 선택하도록 사용자를 제한하고 양식을 완료하기 쉽도록 작성할 수 있습니다.

선택사항 목록을 최신 정보로 고치려면 운영자는 웹 에디터를 다시 시작해야 합니다. 또한 양식 설명 필드 속성 및 dynamicSqlSelectionList가 있어서 운영자는 드롭 다운 메뉴가 다시 로드하는 각 시간을 확인하도록 이를 ″true″로 설정할 수 있습니다.

선택사항 목록에 대한 조회는 하나 또는 두 개의 열 중 하나를 리턴하는 결과 설정을 가질 수 있습니다. 결과가 두 열로 리턴될 경우, 두 번째 열은 저장된 실제 값을 포함하고 첫 번째 열은 사용자 레이블을 포함합니다.

필드 태그의 sqlSelectionList 속성에서 SQL 조회를 입력하거나 XML 양식 설명 파일에서 열거를 작성하여 선택사항 목록을 작성할 수 있습니다. 다음은 이 두 메소드의 예입니다.

```
<field name="MEMBER_ID"
  showInCreateMode="false"
  fieldDescription="Member Identifier"
  .
  .
  readOnly="false"
  sqlSelectionList="select orgentityname,orgentity.orgentity_id
   from member,orgentity where member.type='O'
   and member.member_id=orgentity.orgentity_id"
  .
  .
  fieldHelp=""
  .
  .
    </field>
      <field name="MARKFORDELETE"
        fieldDescription="Mark for Delete"
        type="NMTOKEN"
        sqlSelectionList=""
      >
      <datatype source="integer">
        <enumeration label="No" value="0"/>
        <enumeration label="Yes" value="1"/>
      </datatype>
    </field>
>
```

true와 false로 각각 1과 0을 사용하는 WebSphere Commerce 데이터베이스 테이블에는 필드가 있습니다. 좀 더 직관적인 필드를 작성하려면 이 필드의 열거를 설정할 수 있습니다. Catalog Manager는 ″NUMDESC″로 명명된 열거 테이블을 작성하기 위한

SQL 스크립트를 제공합니다. 이 스크립트는 "createEnum.sql"로 명명됩니다. 운영자는 아래 예에 표시된 바와 같이 ENUMDESC 테이블을 사용하는 선택사항 목록을 작성하기 위해 sqlSelectionList 속성을 편집할 수 있습니다.

```
fieldDescription="On Special"
.
.
readOnly="false"
sqlSelectionList="select description,value from
  enumdesc where columnname='ALL' and type='YESNO'"
fieldHelp=""
.
.
```

필드 도움말 추가

필드 도움말은 현재 필드에 대한 웹 브라우저 창의 맨 아래에 표시됩니다. XML 양식 설명 파일 내의 필드 태그의 fieldHelp 속성 또는 특성 파일 내의 fieldHelp 키는 이 값을 설정하는 데 사용될 수 있습니다.

예를 들어 특성 파일은 다음 field-help 스펙을 포함할 수도 있습니다.

```
CATEGORY.MARKFORDELETE.defaultValue=No
CATEGORY.MARKFORDELETE.fieldDescription=Delete Entry
```

값이 지정되지 않은 경우, "Enter a value for *field_name* here *field_type*""의 기본 메시지가 작성됩니다.

주: 양식 정보를 포함하는 특성 파일이 있을 경우, 이는 XML 양식 설명 파일의 항목에 우선합니다.

검색 결과 및 작업 세션 항목 사용자 정의

검색 결과 페이지에 표시되는 열의 세트는 사용자 정의될 수 있습니다. XML 양식 설명에는 필드 태그의 showColumnInList 속성이 있습니다. 검색 결과에 필드를 포함하려면 "true"로 이 속성을 설정하십시오. 그렇지 않으면 검색 결과 보기 열 중 하나로 나타나지 않습니다. 양식이 표시될 때, 필드가 나타납니다.

weProcessList 파일 편집

weProcessList 파일은 웹 편집기 작업 세션이 처리될 때 실행되는 Catalog Manager 유틸리티를 사용자 정의하도록 합니다.

- DB2 NT 2000 AIX Solaris Linux weProcessList.xml 파일은 다음 디렉토리에 위치합니다.
  - NT *drive*:\WebSphere\CommerceServer\xml\wcwebeditor\xml

  - 2000 *drive*:\Program Files\WebSphere\CommerceServer\xml\wcwebeditor\xml
  - AIX /usr/WebSphere/CommerceServer/xml/wcwebeditor/xml
  - Solaris Linux /opt/WebSphere/CommerceServer/xml/wcwebeditor/xml

- Oracle NT 2000 AIX Solaris Linux weProcessListOracle.xml 파일은 다음 디렉토리에 위치합니다.
  - NT *drive*:\WebSphere\CommerceServer\xml\wcwebeditor\xml
  - 2000 *drive*:\Program Files\WebSphere\CommerceServer\xml\wcwebeditor\xml
  - AIX /usr/WebSphere/CommerceServer/xml/wcwebeditor/xml
  - Solaris Linux /opt/WebSphere/CommerceServer/xml/wcwebeditor/xml

- 400 weProcessListAS400.xml 파일은 다음 디렉토리에 위치합니다.
  - /*instroot*/xml/wcwebeditor/xml

이 파일은 다양한 유틸리티에 대한 엔벨로프(envelope) 템플리트를 포함합니다. 또한 운영자가 실행하길 바라는 사용자 정의 응용프로그램에 대한 참조를 포함합니다.

이 파일에는 사용될 수 있는 시스템 변수의 세트가 존재합니다. 예를 들어 시스템 변수 %-dbname%은 로더와 같은 유틸리티의 주어진 호출에 생성되는 엔벨로프(envelope)에 삽입될 데이터베이스 이름을 불러옵니다. XML 양식 설명 파일은 추가, 편집 또는 삭제를 위해 호출되어야 하는 것을 나타내는 이 처리에 대한 참조를 포함합니다.

다음은 weProcessList.xml 파일의 예입니다.

```
<?xml version="1.0" encoding="UTF-8"?>
<processSet>
  <!-- Do not change name of extract -->
  <process name="extract"
      subsystem="com.ibm.wca.MassExtract.extract.ExtractSubSystem">
      <envelope-input xmlns='saf_params.xsd'>
         <param name="-dbname" type="scalar" value="%-dbname%"/>
         <param name="-dbuser" type="scalar" value="%-dbuser%"/>
         <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
        <param name="-outfile" type="file" reside="local" value="%tempFilePath%"/>
        <param name="-filter" type="file" reside="local" value="%tempFileURI1%"/>
      </envelope-input>
  </process>
  <process name="transformer"
      subsystem="com.ibm.wca.XMLTransformer.XMLTransformerSubSystem">
      <envelope-input xmlns='saf_params.xsd'>
         <param name="-infile" type="file" reside="local" value="%tempFileURI%"/>
         <param name="-transform" type="file" reside="local"
           value="%webEditorDir%/xsl/ViewsToWCS51.XSL"/>
         <param name="-outfile" type="file" reside="local" value="%tempFilePath1%"/>
         <param name="-param" value="root=%-dbname%"/>
         <param name="-param" value="dtdname=%-dtdname%"/>
      </envelope-input>
  </process>
```

```
<process name="transformerForDelete"
    subsystem="com.ibm.wca.XMLTransformer.XMLTransformerSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-infile" type="file" reside="local" value="%tempFileURI%"/>
       <param name="-transform" type="file" reside="local"
         value="%webEditorDir%/xsl/ViewsToWCS51.XSL"/>
       <param name="-outfile" type="file" reside="local" value="%tempFilePath1%"/>
       <param name="-param" value="root=%-dbname%"/>
       <param name="-param" value="dtdname=%-dtdname%"/>
       <param name="-param" value="forDelete=true"/>
    </envelope-input>
</process>
<process name="resolver"
    subsystem="com.ibm.wca.IdResGen.IdResGenSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-dbname" type="scalar" value="%-dbname%"/>
       <param name="-dbuser" type="scalar" value="%-dbuser%"/>
       <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
       <param name="-infile" type="file" reside="local"
         value="%previousOutFileAsURI%"/>
       <param name="-outfile" type="file" reside="local"
         value="%tempFilePath2%"/>
       <param name="-propfile" type="file" reside="local"
         value="propertyFiles.IdKeys"/>
       <param name="-method" type="scalar" value="mixed"/>
    </envelope-input>
</process>
<!-- Resolver as first process -->
<process name="resolverFirstProcess"
    subsystem="com.ibm.wca.IdResGen.IdResGenSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-dbname" type="scalar" value="%-dbname%"/>
       <param name="-dbuser" type="scalar" value="%-dbuser%"/>
       <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
       <param name="-infile" type="file" reside="local" value="%tempFileURI%"/>
       <param name="-outfile" type="file" reside="local" value="%tempFilePath2%"/>
       <param name="-propfile" type="file" reside="local" value="propertyFiles.IdKeys"/>
       <param name="-method" type="scalar" value="mixed"/>
    </envelope-input>
</process>
<process name="loader"
    subsystem="com.ibm.wca.MassLoader.MassLoadSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-dbname" type="scalar" value="%-dbname%"/>
       <param name="-dbuser" type="scalar" value="%-dbuser%"/>
       <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
       <param name="-commitcount" type="scalar" value="1000"/>
       <param name="-infile" type="file" reside="local" value="%previousOutFileAsURI%"/>
       <param name="-method" type="scalar" value="sqlimport"/>
       <param name="-noprimary" type="scalar" value="insert"/>
    </envelope-input>
</process>
<process name="loaderFirstProcess"
    subsystem="com.ibm.wca.MassLoader.MassLoadSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-dbname" type="scalar" value="%-dbname%"/>
       <param name="-dbuser" type="scalar" value="%-dbuser%"/>
       <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
       <param name="-commitcount" type="scalar" value="1000"/>
       <param name="-infile" type="file" reside="local" value="%tempFileURI%"/>
       <param name="-method" type="scalar" value="sqlimport"/>
       <param name="-noprimary" type="scalar" value="insert"/>
    </envelope-input>
</process>
<process name="loaderForDelete"
    subsystem="com.ibm.wca.MassLoader.MassLoadSubSystem">
    <envelope-input xmlns='saf_params.xsd'>
       <param name="-dbname" type="scalar" value="%-dbname%"/>
       <param name="-dbuser" type="scalar" value="%-dbuser%"/>
       <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
       <param name="-commitcount" type="scalar" value="1000"/>
       <param name="-infile" type="file" reside="local" value="%previousOutFileAsURI%"/>
```

```
        <param name="-delete" type="scalar" value=""/>
      </envelope-input>
  </process>
  <process name="loaderForDeleteFirstProcess"
      subsystem="com.ibm.wca.MassLoader.MassLoadSubSystem">
      <envelope-input xmlns='saf_params.xsd'>
         <param name="-dbname" type="scalar" value="%-dbname%"/>
         <param name="-dbuser" type="scalar" value="%-dbuser%"/>
         <param name="-dbpwd" type="scalar" value="%-dbpwd%"/>
         <param name="-commitcount" type="scalar" value="1000"/>
         <param name="-infile" type="file" reside="local" value="%tempFileURI%"/>
         <param name="-delete" type="scalar" value=""/>
      </envelope-input>
  </process>
 <process name="saveToFile"
           cmd="cmd.exe /c c:\temp\theBatchFile.bat"
           args="-infile %tempFilePath% -dbname %-dbname%"
  />
</processSet>
```

주: 참조 파일은 대소문자가 구분됩니다.

다음 테이블은 응용프로그램이 이해하는 올바른 대체 변수의 목록을 포함합니다.

% 대체 변수	리턴
%-dbname%	현재 데이터베이스의 이름
%-dbuser%	데이터베이스 사용자 이름
%-dtdname%	XML 파일에 대한 DTD의 URI 위치
%-dbpwd%	데이터베이스 사용자 이름에 대한 암호
%tempFilePath% %tempFilePath1% %tempFilePath2%	임시 파일에 대한 전체 경로 이는 고유 임시 파일 이름입니다. 이것은 엔벨로프(envelope) 정의 또는 명령 행의 템플리트 구문에 지정될 수도 있습니다. 예를 들어 %tempFilePath%가 엔벨로프(envelope) 템플리트의 -infile 매개변수의 속성 값에 지정될 경우, 웹 편집기는 임시 파일 위치에 작업 세션의 데이터를 작성합니다.
서브시스템용 %tempFileURI% %tempFileURI1% %tempFileURI2%	임시 URI 임시 URI는 %tempFilePath%%tempFilePath2%가 표시한 동일한 파일의 URI입니다. 이것은 첨가된 파일 세트가 아니라 다양한 구문으로 리턴되는 동일하게 생성된 임시 파일을 검색하는 방법입니다.
%previousOutFileAsURI%	URI와 같은 이전의 tasks -outfile 매개 변수의 URI 표시를 제공합니다.
%webEditorDir%	웹 편집기 설치 위치

webeditor.xsl 파일 편집

웹 편집기는 기본 XML 스타일 시트와 같은 webeditor.xsl 파일을 사용합니다. 이것은 다음 디렉토리에 위치합니다.

- NT *drive*:\WebSphere\CommerceServer\xml\wcwebeditor\xsl

- 2000 *drive*:\Program Files\WebSphere\CommerceServer\xml\wcwebeditor\xsl
- AIX /usr/WebSphere/CommerceServer/xml/wcwebeditor/xsl
- Solaris Linux /opt/WebSphere/CommerceServer/xml/wcwebeditor/xsl
- 400 /*instroot*/xml/wcwebeditor/xsl

webeditor.xsl 파일을 편집해서, 운영자는 웹 편집기 출력의 포맷을 변경할 수 있습니다. 다음은 webeditor.xsl 파일 컨텐츠의 견본입니다.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="html"/>
  <!-- Largest single line entry field value for fields larger than this a
    TEXTAREA is created -->
  <xsl:variable name="maxEntryFieldSize" select="80"/>
  <xsl:template match ="/">
    <xsl:apply-templates/>
 </xsl:template>
 <!-- Read Only field -->
 <xsl:template match="readOnly" name="readOnly">
    <xsl:element name="input">
      <xsl:attribute name="name"><xsl:value-of select="@name"/></xsl:attribute>
      <xsl:attribute name="type">hidden</xsl:attribute>
      <xsl:attribute name="value"><xsl:value-of select="@defaultValue"/></xsl:attribute>
        <!--
        <xsl:attribute name="onFocus">this.blur()</xsl:attribute>
        <xsl:attribute name="style">border-style:groove</xsl:attribute>
         -->
  </xsl:element>
      <table border="1" cellpadding="0" cellspacing="0" width="155" bgcolor="#C0C0C0">
         <tr>
           <td>
              <img border="0" src="/webeditor/image/space.gif" width="1"
                height="17"/><xsl:value-of select="@defaultValue"/>
           </td>
       </tr>
     </table>
 </xsl:template>
</xsl:stylesheet>
```

제 12 장 카탈로그로 작업

다음 프로시저는 카탈로그 데이터를 포함하는 데이터베이스에서 테이블에서 레코드를 추가, 수정 및 삭제하는 웹 편집기를 사용하여 카탈로그로 작업하는 방법을 설명합니다.

웹 편집기를 사용하여 테이블에 레코드 추가

웹 편집기를 사용해서 테이블에 레코드를 추가하려면 다음을 수행하십시오.

1. 웹 브라우저에서 다음 URL을 여십시오.

   `https://`*`host_name`*`:8000/wcm/webeditor`

   여기서 *host_name*은 WebSphere Application Server의 완전한 HTTP 호스트 이름입니다.

   웹 편집기 데이터베이스 로그온 창이 표시됩니다.
2. 데이터베이스 사용자 이름 및 암호를 입력한 후 **로그온**을 누르십시오.

   웹 편집기는 왼쪽 메뉴 표시줄에 있는 테이블 이름의 목록으로 브라우저 창에 표시됩니다.
3. 왼쪽 메뉴 표시줄의 서브메뉴 추가에서 해당 하이퍼링크를 누르십시오.

   해당 양식이 표시됩니다.
4. 양식에 필요한 모든 데이터를 입력하십시오.
5. **작업 세션으로 이동**을 누르십시오.

   양식에 대한 작업 세션 결과가 표시됩니다. 이 결과는 편집, 추가 및 삭제되었지만 웹 편집 작업 세션 동안 지워지지 않거나 처리되지 않은 모두를 포함합니다.
6. 작업 세션에서 어떤 레코드 변경을 제거하려면, 제거를 원하는 개별 레코드 변경 앞의 상자를 확인하고, **선택된 것 지우기**를 누르십시오.
7. 데이터베이스에 선택된 변경을 제출하려면 **처리 작업 세션**을 누르십시오.

   상태 페이지에는 처리가 완료된 메시지 상태가 표시됩니다.
8. 웹 사이트를 탐색하려면, 해당 하이퍼링크를 누르고 변경의 작성 여부를 검증하십시오.

웹 편집기를 사용하여 테이블에서 레코드 수정

웹 편집기를 사용하여 테이블의 레코드를 수정하려면 다음을 수행하십시오.

1. 웹 브라우저에서 다음 URL을 여십시오.

   `https://host_name:8000/wcm/webeditor`

   여기서 *host_name*은 WebSphere Application Server의 완전한 HTTP 호스트 이름입니다.

   웹 편집기 데이터베이스 로그온 창이 표시됩니다.
2. 데이터베이스 사용자 이름 및 암호를 입력한 후 **로그온**을 누르십시오.

   웹 편집기는 왼쪽 메뉴 표시줄에 있는 테이블 이름의 목록으로 브라우저 창에 표시됩니다.
3. 왼쪽 메뉴 표시줄의 서브메뉴 검색에서 해당 하이퍼링크를 누르십시오.

   적합한 탐색 페이지가 표시됩니다.
4. 다음을 수행하여 검색 기준을 지정하십시오.
   a. 검색에서 지정을 원하는 개별 속성 옆의 확인 상자를 선택하십시오.
   b. 탐색에 사용하길 원하는 로직을 선택하려면 개별적으로 선택된 속성에 대한 해당 드롭 다운 메뉴를 사용하십시오.
   c. 개별적으로 선택된 속성에 대한 다음 필드에서 탐색에서 사용하길 원하는 값을 입력 또는 선택하십시오.
5. **찾기**를 누르십시오.

   이것은 웹 편집기에 검색 기준을 제출합니다.

   상태 페이지에는 탐색 기준에 부합하는 레코드의 수로 표시합니다.
6. 다음 중 하나를 수행하십시오.
   - 발견된 레코드의 목록을 보려면 **로드 데이터**를 누르십시오.

     웹 편집기는 조회에서 검색된 레코드의 항목을 표시합니다.

     6단계로 가십시오.
   - 탐색 페이지로 리턴하려면 **새로 검색**을 누르십시오.

     3단계로 돌아가십시오.
7. 편집을 원하는 레코드를 선택하십시오.

   해당 양식이 표시됩니다.
8. 편집을 원하는 필드 아래로 화면이동하고 해당 컨텐츠를 변경하십시오.
9. **작업 세션으로 이동**을 누르십시오.

   양식에 대한 작업 세션 결과가 표시됩니다. 이 결과는 편집, 추가 및 삭제되었지만 웹 편집 작업 세션 동안 지워지지 않거나 처리되지 않은 모두를 포함합니다.

10. 작업 세션에서 어떤 레코드 변경을 제거하려면, 제거를 원하는 개별 레코드 변경 앞의 상자를 확인하고 **선택된 것 지우기**를 누르십시오.
11. 데이터베이스에 선택된 변경을 제출하려면 **처리 작업 세션**을 누르십시오.

    상태 페이지에는 처리가 완료된 메시지 상태가 표시됩니다.
12. 웹 사이트를 탐색하려면, 해당 하이퍼링크를 누르고 변경의 작성 여부를 검증하십시오.

웹 편집기를 사용하여 테이블에서 레코드 삭제

웹 편집기를 사용하여 테이블에서 레코드를 삭제하려면 다음을 수행하십시오.

1. 웹 브라우저에서 다음 URL을 여십시오.

   `https://`*`host_name`*`:8000/wcm/webeditor`

   여기서 *host_name*은 WebSphere Application Server의 완전한 HTTP 호스트 이름입니다.

   웹 편집기 데이터베이스 로그온 창이 표시됩니다.
2. 데이터베이스 사용자 이름 및 암호를 입력한 후 **로그온**을 누르십시오.

   웹 편집기는 왼쪽 메뉴 표시줄에 있는 테이블 이름의 목록으로 브라우저 창에 표시됩니다.
3. 왼쪽 메뉴 표시줄의 서브메뉴 검색에서 해당 하이퍼링크를 누르십시오.

   적절한 탐색 페이지가 표시됩니다.
4. 다음을 수행하여 검색 기준을 지정하십시오.
   a. 검색에서 지정을 원하는 개별 속성 옆의 확인 상자를 선택하십시오.
   b. 탐색에 사용하길 원하는 로직을 선택하려면 개별적으로 선택된 속성에 대한 해당 드롭 다운 메뉴를 사용하십시오.
   c. 개별적으로 선택된 속성에 대한 다음 필드에서 탐색에서 사용하길 원하는 값을 입력 또는 선택하십시오.
5. **찾기**를 누르십시오.

   이것은 웹 편집기에 검색 기준을 제출합니다.

   상태 페이지에는 탐색 기준에 부합하는 레코드의 수로 표시합니다.
6. 다음 중 하나를 수행하십시오.
   - 발견된 레코드의 목록을 보려면 **로드 데이터**를 누르십시오.

     웹 편집기는 조회에서 검색된 레코드의 항목을 표시합니다.

     6단계로 가십시오.
   - 탐색 페이지로 리턴하려면 **새로 검색**을 누르십시오.

3단계로 돌아가십시오.

7. 삭제를 원하는 개별 레코드 옆의 확인 상자를 선택하십시오.
8. **목록 삭제로 이동**을 누르십시오.
9. 왼쪽 메뉴 표시줄의 서브메뉴 세션에서 해당 하이퍼링크를 누르십시오.

   양식에 대한 작업 세션 결과가 표시됩니다. 이 결과는 편집, 추가 및 삭제되었지만 웹 편집 작업 세션 동안 지워지지 않거나 처리되지 않은 모두를 포함합니다.
10. 작업 세션에서 어떤 레코드 변경을 제거하려면, 제거를 원하는 개별 레코드 변경 앞의 상자를 확인하고, **선택된 것 지우기**를 누르십시오.
11. 데이터베이스에 변경을 제출하려면 **처리 작업 세션**을 누르십시오.

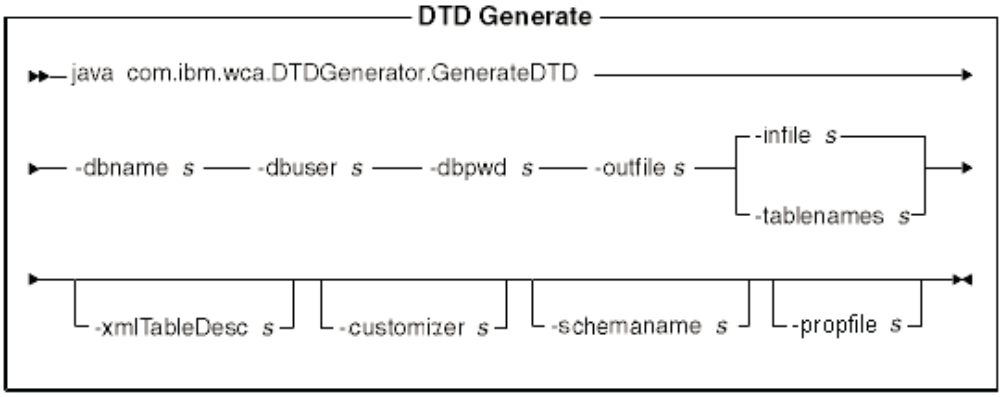
    상태 페이지에는 처리가 완료된 메시지 상태가 표시됩니다.
12. 웹 사이트를 탐색하려면, 해당 하이퍼링크를 누르고 변경의 작성 여부를 검증하십시오.

제 4 부 명령 참조

제 13 장 DTD 생성 명령

이 명령은 로더 패키지로 사용하기 위한 DTD 및 스키마 파일을 작성합니다.

Windows, AIX, Linux 및 Solaris 시스템용 DTD 생성 명령



주:

1. 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.
2. 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

-dbname
대상 데이터베이스의 이름

-dbuser
데이터베이스에 연결하는 사용자 이름

-dbpwd
데이터베이스에 연결하는 사용자에 대한 암호

-outfile
출력 DTD 파일의 이름

-infile 개별 행에 데이터베이스 테이블 이름을 포함하는 입력 파일의 이름

-tablenames

쉼표로 분리된 테이블의 이름

-xmlTableDesc

작성될 스키마 파일의 파일 경로

-customizer

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 DB2ConnectionCustomizer.properties입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer d:\wc\prop\dtdgen.properties
```

이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer dtdgen
```

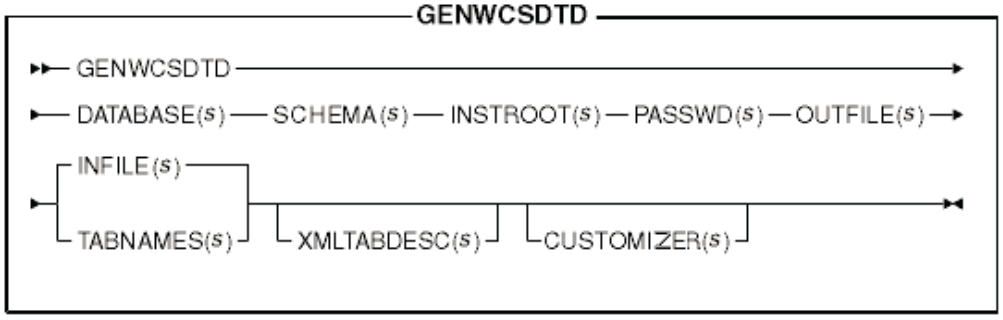
-schemaname

대상 데이터베이스 스키마의 이름

-propFile

도움말 텍스트, 기본값, 및 필드 설명 정보가 웹 편집기 양식 설명에 저장될 수 있는 외부 특성 파일과 같은 특성을 포함하는 파일

iSeries 시스템용 DTD 생성 명령



주: 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

DATABASE

관계형 데이터베이스 디렉토리에 표시된 대상 데이터베이스의 이름

SCHEMA

인스턴스 이름과 같은 대상 데이터베이스 스키마의 이름

INSTROOT

WebSphere Commerce 인스턴스 루트 경로의 전체 이름.
(예: /QIBM/UserData/WebCommerce/instances/*instance_name*)

PASSWD

WebSphere Commerce 인스턴스에 대한 암호

OUTFILE

출력 DTD 파일의 이름

INFILE

개별 행에 데이터베이스 테이블 이름을 포함하는 입력 파일의 이름

TABNAMES

쉼표로 분리된 테이블의 이름

XMLTABDESC

작성될 스키마 파일의 파일 경로. 이 매개변수는 선택적입니다.

CUSTOMIZER

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 ISeries_GENWCSDTD_Customizer.properties입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(/wc/prop/dtdgen.properties)
```

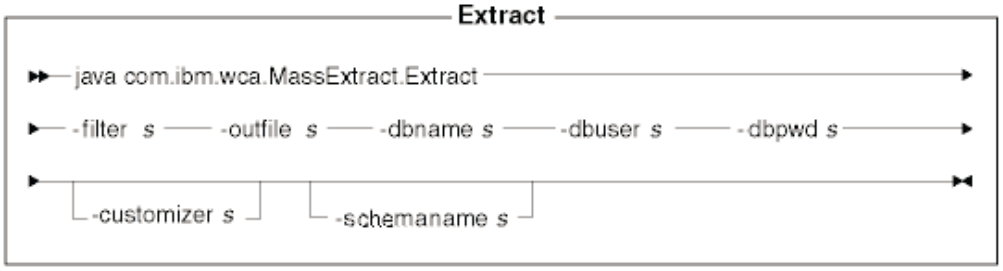
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(dtdgen)
```

제 14 장 추출 명령

이 명령은 XML 파일의 데이터베이스에서 데이터의 선택된 서브세트를 추출합니다.

Windows, AIX, Linux 및 Solaris 시스템용 추출 명령



주:

1. 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.
2. 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

-filter 추출 필터 파일의 이름

-outfile

저장될 추출 데이터가 있는 출력 XML 파일의 이름

-dbname

추출되고 있는 데이터에서 데이터베이스의 이름

-dbuser

추출되고 있는 데이터에서 데이터베이스에 대한 데이터베이스 사용자 이름

-dbpwd

추출되고 있는 데이터에서 데이터베이스에 대한 사용자 이름으로 연관된 암호

-customizer

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 DB2ConnectionCustomizer.properties입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer d:\wc\prop\extract.properties
```

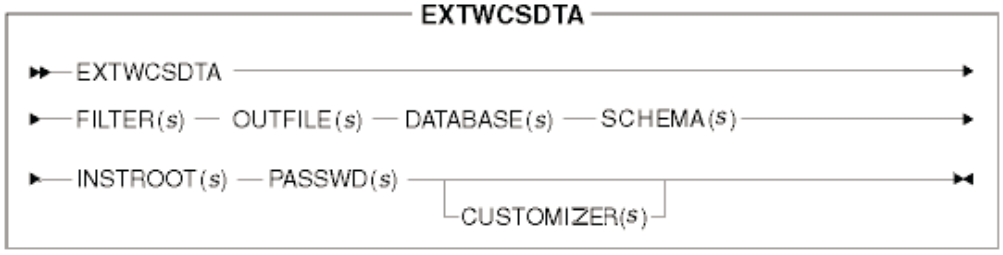
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer extract
```

-schemaname

대상 데이터베이스 스키마의 이름

iSeries 시스템용 추출 명령



주: 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

FILTER

추출 필터 파일의 이름

OUTFILE

저장될 추출 데이터가 있는 출력 XML 파일의 이름

DATABASE

관계형 데이터베이스 디렉토리에 표시된 추출되고 있는 데이터에서 데이터베이스의 이름

SCHEMA

인스턴스 이름과 같은 추출되고 있는 데이터의 데이터베이스 스키마의 이름

INSTROOT

WebSphere Commerce 인스턴스 루트 경로의 전체 이름.
(예: /QIBM/UserData/WebCommerce/instances/*instance_name*)

PASSWD

WebSphere Commerce 인스턴스에 대한 암호

CUSTOMIZER

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 ISeries_EXTWCSDTA_Customizer.properties입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(/wc/prop/extract.properties)
```

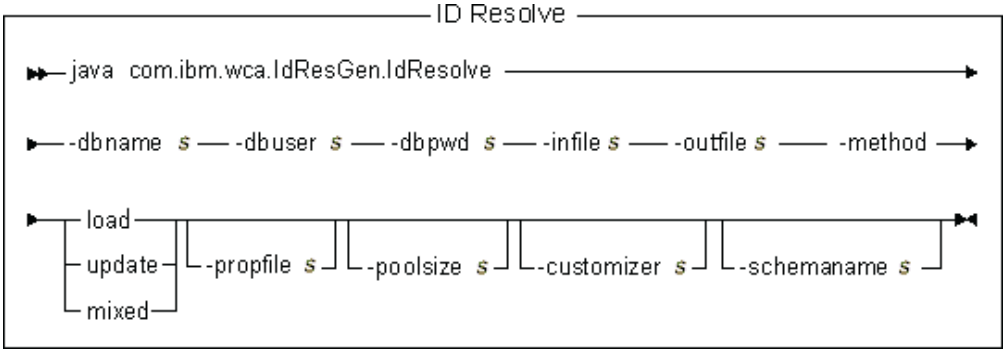
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(extract)
```

제 15 장 ID 분석 명령

이 명령은 데이터베이스에 로드하기 전에 요구된 XML 요소에 대한 식별자를 생성합니다.

Windows, AIX, Linux 및 Solaris 시스템용 ID 분석 명령



주:

1. 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.
2. 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

-dbname
대상 데이터베이스의 이름

-dbuser
데이터베이스에 연결하는 사용자 이름

-dbpwd
데이터베이스에 연결하는 사용자에 대한 암호

-infile 테이블 레코드 설명자를 포함하는 입력 XML 문서의 이름

-outfile
로더에 입력으로 사용될 수 있는 출력 XML 파일의 이름

-method

입력 파일을 처리하는 데 사용되는 메소드

- 파일의 **모든** 레코드가 데이터베이스에 ***존재하지 않을*** 경우 load 메소드를 사용하여 입력 파일을 처리하십시오.
- 파일의**모든** 레코드가 데이터베이스에 ***존재할*** 경우 update 메소드를 사용하여 입력 파일을 처리하십시오.
- 파일의 ***단지 일부*** 레코드가 데이터베이스에 ***존재할*** 경우 mixed 메소드를 사용하여 입력 파일을 처리하십시오.

기본 메소드는 load입니다.

-propfile

이름 값 쌍의 양식의 Java 특성 파일을 포함하는 텍스트 파일. 이 파일은 foreign-key 식별자 찾아보기에 대한 look-aside 열 이름을 정의하고 선택은 기본 테이블(CATEGORY 및 PRODUCT와 같은) 조회에 대한 서술자를 선택하는 데 사용됩니다. 식별자가 포함되지 않도록 정의된 고유 색인을 보유하는 테이블에 대한 이 파일에서 항목을 생략할 수 있습니다. 기본 파일은 `IdResolveKeys.properties`입니다. 이 파일이 확장자 `.properties`를 가지더라도, 값을 지정할 때 이 확장자를 사용하지 마십시오.

-poolsize

예약된 식별자의 이름. 기본 번호는 50입니다.

-customizer

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 `DB2ConnectionCustomizer.properties`입니다. 사용자 정의 특성 파일은 다음 두 가지 예 중 하나에 표시된 것처럼 지정될 수 있습니다.

```
-customizer d:\wc\prop\idres.properties
```

```
-customizer d:\wc\prop\idres
```

이 파일이 현재 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer idres.properties
```

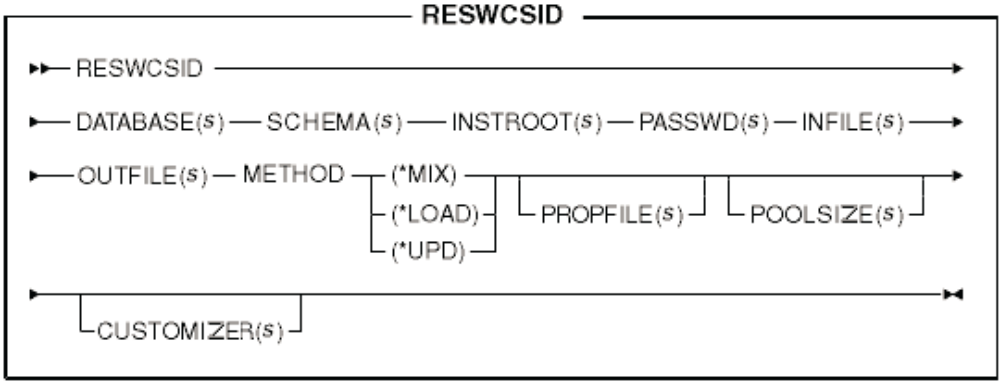
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer idres
```

-schemaname

대상 데이터베이스 스키마의 이름

iSeries 시스템용 ID 분석 명령



주: 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

DATABASE

관계형 데이터베이스 디렉토리에 표시된 대상 대이터베이스의 이름

SCHEMA

인스턴스 이름과 같은 대상 데이터베이스 스키마의 이름

INSTROOT

WebSphere Commerce 인스턴스 루트 경로의 전체 이름
(예: `/QIBM/UserData/WebCommerce/instances/`*instance_name*)

PASSWD

WebSphere Commerce 인스턴스에 대한 암호

INFILE

테이블 레코드 설명자를 포함하는 입력 XML 문서의 이름

OUTFILE

로더에 입력으로 사용될 수 있는 출력 XML 파일의 이름

METHOD

입력 파일을 처리하는 데 사용되는 메소드

- 파일의 **모든** 레코드가 데이터베이스에 ***존재하지 않을*** 경우, load 메소드 (*LOAD)를 사용하여 입력 파일을 처리하십시오.
- 파일의**모든** 레코드가 데이터베이스에 ***존재할*** 경우, update 메소드(*UPD)를 사용하여 입력 파일을 처리하십시오.
- 파일의 ***단지 일부*** 레코드가 데이터베이스에 ***존재할*** 경우, mixed 메소드 (*MIX)를 사용하여 입력 파일을 처리하십시오.

PROPFILE

이름 값 쌍의 양식의 Java 특성 파일을 포함하는 텍스트 파일. 이 파일은 foreign-key 식별자 찾아보기에 대한 look-aside 열 이름을 정의하고 선택은 기본 테이블(CATEGORY 및 PRODUCT와 같은) 조회에 대한 서술자를 선택하는 데 사용됩니다. 식별자가 포함되지 않도록 정의된 고유 색인을 보유하는 테이블에 대한 이 파일에서 항목을 생략할 수 있습니다. 기본 파일은 IdResolveKeys.properties입니다. 이 파일이 확장자 .properties를 가지더라도, 값을 지정할 때 이 확장자를 사용하지 마십시오.

POOLSIZE

예약된 식별자의 이름. 기본 번호는 50입니다.

CUSTOMIZER

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 ISeries_RESWCSID_Customizer.properties입니다. 사용자 정의 특성 파일은 다음 두 가지 예 중 하나에 표시된 것처럼 지정될 수 있습니다.

```
CUSTOMIZER(/wc/prop/idres.properties)
```

```
CUSTOMIZER(/wc/prop/idres)
```

이 파일이 현재 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(idres.properties)
```

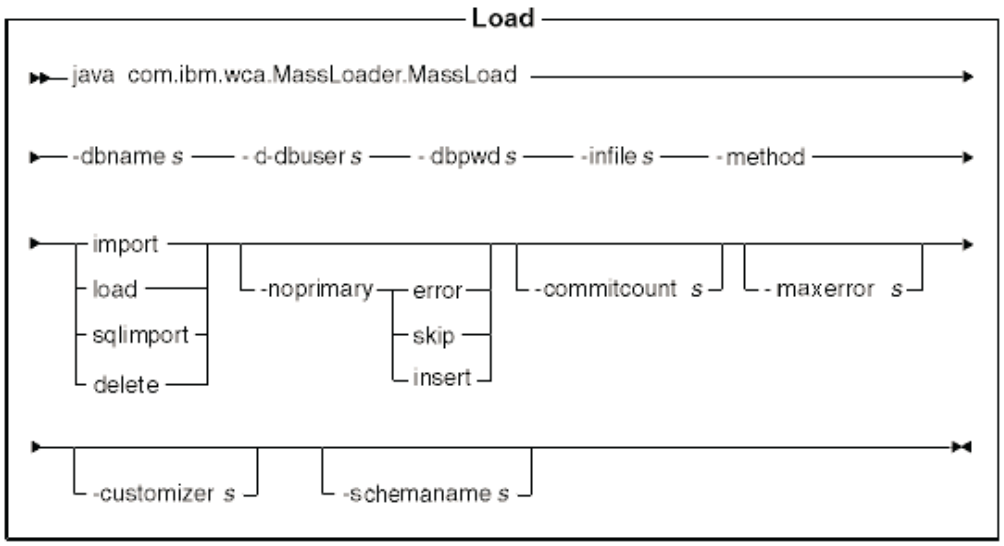
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(idres)
```

제 16 장 로드 명령

이 명령은 대상 데이터베이스에 XML 입력 파일을 로드합니다.

Windows, AIX, Linux 및 Solaris 시스템용 로드 명령



주:

1. 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.
2. 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

-dbname
대상 데이터베이스의 이름

-dbuser
데이터베이스에 연결하는 사용자 이름

-dbpwd
데이터베이스에 연결하는 사용자에 대한 암호

-infile 입력 XML 파일의 이름

-method

데이터베이스에 삽입될 때 사용되는 로더에 대한 조작 모드.

- load 메소드는 데이터베이스 공급업체로부터 원시 로더를 사용합니다. 사용자는 load 메소드를 로컬 및 원격 Oracle 데이터베이스 모두에 사용할 수 있으나 load 메소드는 단지 로컬 DB2 데이터베이스에 사용됩니다.
- 원격 DB2 데이터베이스에 데이터를 로드를 위해 반입 메소드를 사용하십시오. 반입 또는 갱신 옵션이 데이터베이스 공급업체에서 사용 가능할 경우, import 메소드는 반입 또는 갱신 옵션을 사용합니다. 반입이나 갱신 옵션을 사용할 수 없는 데이터베이스(예: Oracle)에 대해 이 방법을 지정할 경우, JDBC를 사용하는 SQL 문은 데이터베이스 갱신에 사용됩니다.
- SQL 반입(sqlimport) 메소드는 로컬 및 원격 데이터베이스 모두에 사용될 수 있습니다.
- delete 메소드는 데이터베이스로부터 데이터를 삭제합니다.

상품 어드바이저 공간 검색 동기화를 사용할 경우, sqlimport 또는 delete 메소드 둘 중 하나를 사용해야 합니다.

-noprimary

입력 파일의 레코드에 대한 1차 키가 누락될 때, 로더가 취하는 조치. 오류 옵션이 오류 및 종료로 1차 키가 누락되었음을 보고해야 함을 표시합니다. 생략 옵션은 1차 키를 보유하지 않는 입력 파일에서 어떤 레코드이든지 생략합니다. 삽입 옵션은 데이터 처리(삽입 또는 삭제)를 시도합니다. 기본 조치는 오류입니다.

-commitcount

조작의 SQL update 운영 메소드를 사용할 때, 데이터베이스가 오류를 범하기 전에 처리된 레코드의 수. 기본 수는 1입니다.

-maxerror

조작의 SQL update 운영 메소드에서 로더가 종료한 이후의 오류 수

-customizer

사용될 사용자 정의 특성 파일의 이름. `MassLoadCustomizer.properties`는 기본 파일입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer d:\wc\prop\ml.properties
```

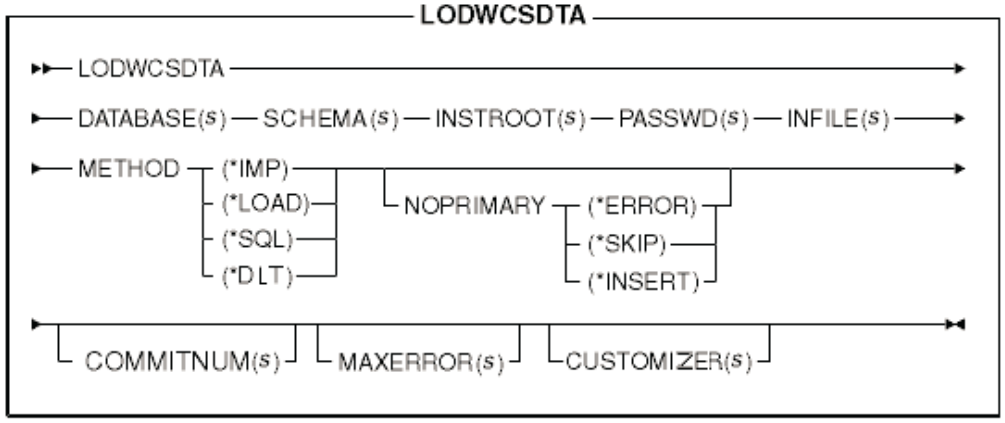
이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
-customizer ml
```

-schemaname

대상 데이터베이스 스키마의 이름

iSeries 시스템용 로드 명령



주: 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

DATABASE

관계형 데이터베이스 디렉토리에 표시된 대상 데이터베이스의 이름

SCHEMA

인스턴스 이름과 같은 대상 데이터베이스 스키마의 이름

INSTROOT

WebSphere Commerce 인스턴스 루트 경로의 전체 이름.

(예: `/QIBM/UserData/WebCommerce/instances/`*instance_name*)

PASSWD

WebSphere Commerce 인스턴스에 대한 암호

INFILE

입력 XML 파일의 이름

METHOD

데이터베이스에 삽입될 때 사용되는 로더에 대한 조작 모드.

- load 메소드(*LOAD)는 데이터베이스 공급업체로부터 원시 로더를 사용합니다. 사용자는 load 메소드(*LOAD)를 로컬 및 원격 Oracle 데이터베이스 모두에 사용할 수 있으나 load 메소드(*LOAD)는 단지 로컬 DB2 데이터베이스에 사용됩니다.
- 원격 DB2 데이터베이스에 데이터를 로드를 위해 반입 옵션(*IMP)을 사용하십시오. 반입 또는 갱신 옵션이 데이터베이스 공급업체에서 사용 가능할 경

우, import 메소드(*IMP)는 반입 또는 갱신 옵션을 사용합니다. 반입이나 갱신 옵션이 불가능할 경우 JDBC를 사용하는 SQL 문은 데이터베이스 갱신에 사용됩니다.

- SQL 반입(*SQL) 메소드는 로컬 및 원격 데이터베이스 모두에 사용될 수 있습니다.
- delete 메소드(*DLT)는 데이터베이스로부터 데이터를 삭제합니다.

NOPRIMARY

입력 파일에서 레코드에 대한 1차 키가 누락될 때, 로더가 취하는 조치. 오류 옵션(*ERROR)이 오류 및 종료로 1차 키가 누락되었음을 표시합니다. 생략 옵션(*ERROR)은 1차 키를 보유하지 않는 입력 파일에서 어떤 레코드이든지 생략합니다. 삽입 옵션(*INSERT)은 데이터 처리(삽입 또는 삭제)를 시도합니다. 기본 조치는 오류입니다.

COMMITNUM

조작의 SQL update 운영 메소드를 사용할 때, 데이터베이스가 오류를 범하기 전에 처리된 레코드의 수. 기본 수는 1입니다.

MAXERROR

조작의 SQL update 운영 메소드에서 로더가 종료한 이후의 오류 수

CUSTOMIZER

사용될 사용자 정의 특성 파일의 이름. 기본 파일은 `ISeries_LODWCSDTA_Customizer.properties`입니다. 사용자 정의 특성 파일은 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(/wc/prop/ml.properties)
```

이 파일이 classpath 시스템 환경 변수에 지정된 디렉토리에 있는 경우, 동일한 파일이 다음 예에서와 같이 지정될 수 있습니다.

```
CUSTOMIZER(ml)
```

제 17 장 텍스트 변환 명령

이 명령은 문자 분리 변수 포맷과 XML 포맷간의 데이터를 변환합니다.

Windows, AIX, Linux 및 Solaris 시스템용 텍스트 변환 명령

주: 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.

매개변수 값

매개변수 파일(*parameter.txt*)의 쉼표로 다음 값을 지정하고 분리할 수 있습니다.

- 입력 파일

  변환될 파일의 이름
- 스키마 파일

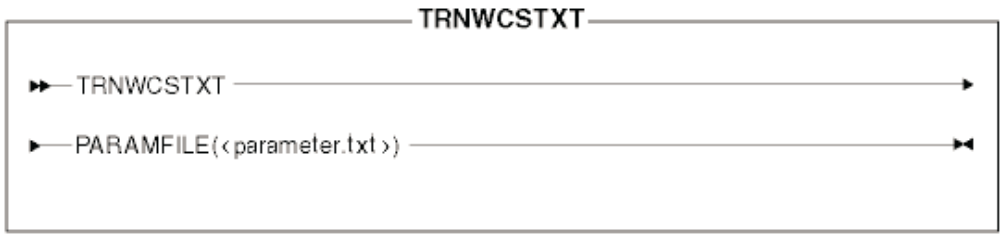
  변환에서 사용될 XML 스키마 파일의 이름
- 출력 파일

  변환된 데이터가 저장될 출력 파일의 이름
- 변환 메소드

  출력 파일에 데이터를 추가하는 데 사용될 메소드. 새 파일을 작성할 경우, **작성**을 지정하거나 출력 데이터가 기존 데이터 파일에 추가될 경우 **추가**를 지정하십시오.

주: 이 파일은 "적하 목록" 또는 "명령" 파일이라고도 합니다.

iSeries 시스템용 텍스트 변환 명령



매개변수 값

매개변수 파일(*parameter.txt*)의 쉼표로 다음 값을 지정하고 분리할 수 있습니다.

- 입력 파일

  변환될 파일의 이름
- 스키마 파일

  변환에서 사용될 XML 스키마 파일의 이름
- 출력 파일

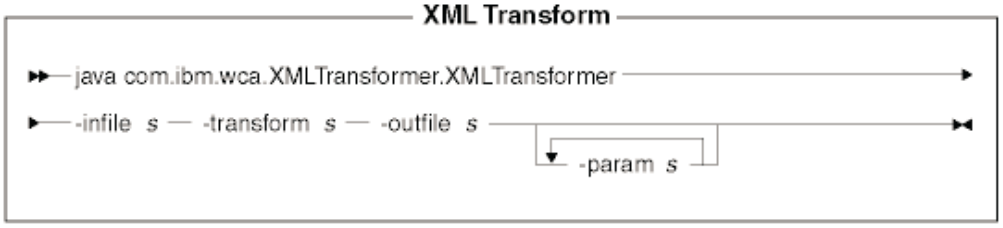
  변환된 데이터가 저장될 출력 파일의 이름
- 변환 메소드

  출력 파일에 데이터를 추가하는 데 사용될 메소드. 새 파일을 작성할 경우, **작성**을 지정하거나 출력 데이터가 기존 데이터 파일에 추가될 경우 **추가**를 지정하십시오.

제 18 장 XML 변환 명령

이 명령은 대체 XML 포맷에 XML 파일을 변환합니다.

Windows, AIX, Linux 및 Solaris 시스템용 XML 변환 명령



주:

1. 위의 도표는 기본적으로 명령 매개변수에 대한 참조로 사용하기 위한 것입니다. 명령에 대해 제공되고 83 페이지의 제 10 장『로더 패키지 명령 및 스크립트 구성』에 나열된 명령 파일 또는 스크립트는 실제 Java 명령에 대해 랩퍼로 작동하며 동일 매개변수를 승인합니다. 그러므로 Java 명령을 직접 호출하기 보다는 명령 파일이나 스크립트를 사용하는 것이 좋습니다.
2. 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

-infile 변환될 파일의 이름

-transform

변환 XSL 규칙 파일의 이름

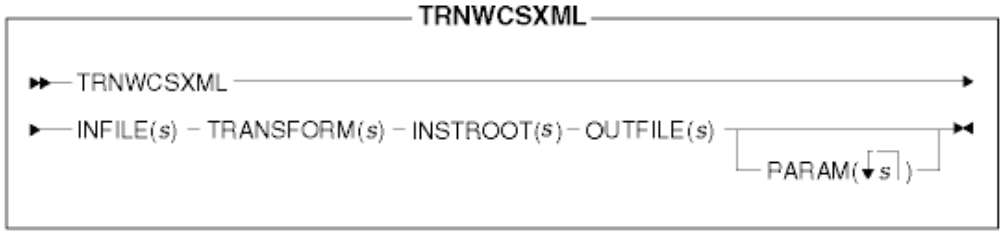
-outfile

변환된 데이터가 저장될 출력 XML 파일의 이름

-param

XML 규칙 파일에 제공될 매개변수. ***이 매개변수는 선택적입니다.*** 이 매개변수는 복수 "이름 값" 쌍을 전달하도록 복수 시간으로 지정될 수 있습니다.

iSeries 시스템용 XML 변환 명령



주: 명령에 대해 매개변수로 지정된 파일 이름 앞에는 상대 또는 절대 경로가 있을 수 있습니다.

매개변수 값

INFILE

변환될 파일의 이름

TRANSFORM

변환 XSL 규칙 파일의 이름

INSTROOT

WebSphere Commerce 인스턴스 루트 경로의 전체 이름.
(예: `/QIBM/UserData/WebCommerce/instances/`*instance_name*)

OUTFILE

변환된 데이터가 저장될 출력 XML 파일의 이름

PARAM

XML 규칙 파일에 제공될 매개변수. ***이 매개변수는 선택적입니다.*** 문자열은 복수 "이름 값" 쌍을 전달하도록 복수 값을 포함할 수 있습니다.

주의사항

이 정보는 미국에서 제공되는 제품 및 서비스용으로 작성된 것입니다. IBM은 다른 국가에서 이 책에 기술된 제품, 서비스 또는 기능을 제공하지 않을 수도 있습니다. 현재 사용할 수 있는 제품 및 서비스에 대한 정보는 한국 IBM 담당자에게 문의하십시오. 이 책에서 IBM 제품, 프로그램 또는 서비스를 언급했다고 해서 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상으로 동등한 제품, 프로그램 또는 서비스를 대신 사용할 수도 있습니다. 그러나 비IBM 제품, 프로그램 또는 서비스의 운영에 대한 평가 및 검증은 사용자의 책임입니다.

IBM은 이 책에서 다루고 있는 특정 내용에 대해 특허를 보유하고 있거나 현재 특허 출원 중일 수 있습니다. 이 책을 제공한다고 해서 해당 특허에 대한 사용권까지 부여하는 것은 아닙니다. 사용권에 대한 의문사항은 다음으로 문의하십시오.

137-270
서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
한국 아이.비.엠 주식회사
고객만족센터
전화번호: 080-023-8080

2바이트(DBCS) 정보와 관련된 사용권 문의는 한국 IBM 고객만족센터에 문의하거나 다음의 주소로 서면 문의하시기 바랍니다.

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

다음의 단락은 현지법과 상충하는 영국이나 기타 국가에서는 적용되지 않습니다. IBM은 타인의 권리 비침해, 상품성 및 특정 목적에의 적합성에 대한 묵시적 보증을 포함하여(단, 이에 한하지 않음) 묵시적이든 명시적이든 어떠한 종류의 보증없이 이 책을 『현 상태대로』 제공합니다. 일부 국가에서는 특정 거래에서 명시적 또는 암시적인 보증의 면책사항을 허용하지 않으므로 이 사항이 적용되지 않을 수도 있습니다.

이 정보에는 기술적으로 부정확한 내용이나 인쇄상의 오류가 있을 수 있습니다. 이 정보는 주기적으로 변경되며, 이 변경사항은 최신판에 통합됩니다. IBM은 이 책에서 설명된 제품 및/또는 프로그램을 사전 통고없이 개선 및/또는 변경할 수 있습니다.

이 정보에서 비IBM의 웹 사이트는 단지 편의상 제공된 것으로, 어떤 방식으로든 이들 웹 사이트를 옹호하고자 하는 것은 아닙니다. 해당 웹 사이트의 자료는 본 IBM 제품 자료의 일부가 아니므로 해당 웹 사이트의 사용으로 인한 위험은 사용자 본인이 감수해야 합니다.

IBM은 귀하의 권리를 침해하지 않는 범위 내에서 적절하다고 생각하는 방식으로 귀하가 제공한 정보를 사용하거나 배포할 수 있습니다.

(1) 독립적으로 작성된 프로그램 및 기타 프로그램(이 프로그램 포함) 간의 정보 교환 (2) 교환된 정보의 상호 이용을 목적으로 정보를 원하는 프로그램 사용권자는 다음 주소로 문의하십시오.

135-270
서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
한국 아이.비.엠 주식회사
고객만족센터

그러한 정보는 해당 및 조건에 따라 사용(예를 들어, 사용권 지불 포함)할 수 있습니다.

이 정보에 기술된 사용권 프로그램 및 가능한 모든 사용권 자료는 IBM 기본 계약, IBM 프로그램 사용권 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

이 책에 포함된 성능 데이터는 제한된 환경에서 산출된 것입니다. 그러므로 다른 운영 환경에서의 결과와 상당히 다를 수도 있습니다. 일부 측정은 개발 단계의 시스템에서 이루어진 것이므로 그 측정치가 일반적으로 사용 가능한 시스템에서도 동일하다고 보장할 수 없습니다. 또한 일부 측정치는 보외법을 통해 이루어졌으므로 실제 결과는 다를 수 있습니다. 이 문서의 사용자는 자신의 특정 환경에 적합한 데이터를 검증해야 합니다.

비IBM 제품에 관한 정보는 해당 제품의 공급업체, 공개 자료 또는 기타 범용 소스로부터 얻은 것입니다. IBM에서는 이 제품들을 테스트하지 않았으므로, 비IBM 제품과 관련된 성능의 정확성, 호환성 또는 배상 청구에 대해서는 확신할 수 없습니다. 비IBM 제품의 성능에 대한 의문사항은 해당 제품의 공급업체에 문의하십시오.

IBM의 장래 방향이나 의도에 대한 모든 진술은 통지 없이 변경되거나 취소될 수 있으며 목적과 목표만을 나타낼 뿐입니다.

표시된 모든 IBM 가격은 IBM이 제안하는 소매가이며, 최신 가격으로 통지 없이 변경될 수 있습니다. 소매업체 가격들은 다양할 수 있습니다.

이 정보는 계획 목적만을 위한 것입니다. 여기의 정보는 설명된 제품이 사용 가능하게 되기 전에 변경될 수 있습니다.

이 책에서는 일상적인 비즈니스 운영에 사용되는 데이터 및 보고서 예가 수록되어 있습니다. 가능한 한 완벽하게 표시하기 위해 예에는 개인, 회사, 브랜드 및 상품 이름이

포함됩니다. 그러한 이름들은 모두 가공의 이름으로서, 실제 회사에서 사용하는 이름 및 주소와 유사하다 하더라도 전적으로 우연입니다.

저작권

이 책에서는 소스 언어로 된 견본 응용프로그램이 들어 있어, 여러 운영체제에서의 프로그래밍 기술에 대해 설명합니다. 견본 프로그램이 작성된 운영체제에 해당하는 응용프로그램 프로그래밍 인터페이스를 따르는 응용프로그램을 개발, 사용, 마케팅, 배포하려는 목적으로 해당 견본 프로그램을 무료로 복사, 수정, 배포할 수 있습니다. 이 책의 예들은 모든 조건에서 완전하게 테스트된 것은 아닙니다. 따라서 IBM은 해당 프로그램의 신뢰성, 서비스 가능성 또는 기능에 대해 보증하거나 암시할 수 없습니다. IBM의 응용프로그램 프로그래밍 인터페이스를 따르는 응용프로그램을 개발, 사용, 마케팅, 배포하려는 목적으로 해당 견본 프로그램을 무료로 복사, 수정, 배포할 수 있습니다.

이 견본 프로그램의 각 사본이나 일부 또는 여기에서 파생된 저작물에는 다음과 같이 저작권 주의사항이 포함되어야 합니다.

이 정보를 소프트카피로 보는 경우, 사진과 컬러 그림은 나타나지 않을 수 있습니다.

상표 및 서비스표

다음 용어는 미국 또는 기타 국가에서 사용되는 IBM Corporation의 상표입니다.

AIX
DB2
DB2 Universal Database
IBM
iSeries
OS/400
WebSphere

Microsoft, Windows, Windows NT 및 Windows 2000은 미국 또는 기타 국가에서 사용되는 Microsoft Corporation의 등록상표입니다.

Oracle은 미국 또는 기타 국가에서 사용되는 Oracle Corporation의 등록상표입니다.

Solaris, Java 및 모든 Java 기반 상표와 로고는 미국 또는 기타 국가에서 사용되는 Sun Microsystems의 상표 또는 등록상표입니다.

기타 회사, 제품 및 서비스 이름은 다른 회사의 상표 또는 서비스표일 수 있습니다.

IBM 한글 지원에 관한 설문

FAX : (02) 3787-0123

보내 주시는 의견은 더 나은 고객 지원 체제를 위한 귀중한 자료가 됩니다.
독자 여러분의 좋은 의견을 기다립니다.

책 제목: IBM® WebSphere® Commerce
Catalog Manager 사용자 안내서
버전 5.4

성 명		직위/담당업무	
회 사 명		부 서 명	
주 소			
전화번호		팩스번호	
전자우편 주소			
사용중인 시스템	o 중대형 서버 o UNIX 서버 o PC 및 PC 서버		

1. IBM에서 제공하는 한글 책자와 영문 책자 중 어느 것을 더 좋아하십니까? 그 이유는 무엇입니까?
   ☐ 한글 책자 ☐ 영문 책자
   (이유:)

2. 본 책자와 해당 소프트웨어에서 사용된 한글 용어에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

3. 본 책자와 해당 소프트웨어에서 번역 품질에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

4. 본 책자의 인쇄 상태에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

5. 한글 소프트웨어 및 책자가 지원되는 분야에 대해 귀하는 어떻게 생각하십니까?
   ☐ 한글 책자를 늘려야 함 ☐ 현재 수준으로 만족
   ☐ 그다지 필요성을 느끼지 않음

6. IBM은 인쇄물 형식(hardcopy)과 화면 형식(softcopy)의 두 종류로 책자를 제공합니다. 어느 형식을 더 좋아하십니까?
   ☐ 인쇄물 형식(hardcopy) ☐ 화면 형식(softcopy) ☐ 둘 다

☞ IBM 한글 지원 서비스에 대해 기타 제안사항이 있으시면 적어주십시오.

☺ 설문에 답해 주셔서 감사합니다.
귀하의 의견은 저희에게 매우 소중한 것이며, 고객 여러분들께 보다 좋은 제품을 제공해 드리기 위해 최선을 다하겠습니다.