IBM WebSphere Business Integration Adapters

Adapter for Manugistics Demand & Fulfillment Management 사용자 안내서

버전 1.0.x

IBM WebSphere Business Integration Adapters

Adapter for Manugistics Demand & Fulfillment Management 사용자 안내서

버전 1.0.x

주!

이 정보와 이 정보가 지원하는 제품을 사용하기 전에, 137 페이지의 『주의사항』의 정보를 읽으십시오.

2003년 12월

이 개정판은 새 개정판에 별도로 명시하지 않는 한 IBM WebSphere Business Integration Adapter for Manugistics Demand & Fulfillment 버전 1.0.x 및 모든 후속 릴리스와 수정판에 적용됩니다.

이 문서에 대한 의견을 보내시려면 ibmkspoe@kr.ibm.com으로 전자 우편을 보내십시오. 고객의 의견을 기대합니다.

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

목차

이 책의 정보

IBM(R) WebSphere(R) Business Integration Adapter 포트폴리오는 주요 e-business 기술, 엔터프라이즈 응용프로그램, 레거시 및 메인프레임 시스템의 통합 연결성을 제공합니다. 제품 세트에는 비즈니스 프로세스 통합 구성요소의 사용자 정의, 작성 및 관리를 위한 도구와 템플리트가 포함됩니다.

이 책에서는 Adapter for Manugistics Demand & Fulfillment Management에 대한 설치, 구성 및 Business Object 개발에 대해 설명합니다.

이 책의 사용자

이 책은 고객 사이트에서 커넥터를 사용하는 컨설턴트, 개발자 및 시스템 관리자를 위한 것입니다.

관련 서적

이 제품과 함께 사용할 수 있는 전체 문서 세트에는 모든 WebSphere Business Integration Adapters 설치에 공통되는 사양 및 구성요소에 대한 설명과 특정 구성요소에 관한 참조 자료가 수록되어 있습니다.

다음 사이트에서 관련 문서를 설치할 수 있습니다.

- 일반 어댑터 정보, WebSphere 메시지 브로커(WebSphere MQ Integrator, WebSphere MQ Integrator Broker, WebSphere Business Integration Message Broker)와 함께 어댑터를 사용하는 경우 및 WebSphere Application Server와 함께 어댑터를 사용하는 경우에는 다음 웹 사이트를 참조하십시오.

  http://www.ibm.com/websphere/integration/wbiadapters/infocenter
- InterChange Server와 함께 어댑터를 사용하는 경우에는 다음 웹 사이트를 참조하십시오.

  http://www.ibm.com/websphere/integration/wicserver/infocenter

  http://www.ibm.com/websphere/integration/wbicollaborations/infocenter
- 메시지 브로커(WebSphere MQ Integrator Broker, WebSphere MQ Integrator 및 WebSphere Business Integration Message Broker)에 대한 자세한 정보:

  http://www.ibm.com/software/integration/mqfamily/library/manualsa/
- WebSphere Application Server에 대한 자세한 정보:

  http://www.ibm.com/software/webservers/appserv/library.html

이 사이트에서는 문서를 다운로드 및 설치하고 보는 데 필요한 간단한 지시사항을 제공합니다.

인쇄 규칙

이 책에서는 다음 규칙을 사용합니다.

`courier font`	명령어, 파일 이름, 사용자 입력 정보 또는 시스템이 화면에 인쇄하는 정보와 같은 리터럴 값을 표시합니다.
굵은체	처음 나타나는 새 용어를 표시합니다.
기울임	변수 이름 또는 상호 참조를 표시합니다.
파란색 윤곽선	온라인으로 매뉴얼을 볼 경우 나타나는 파란색 윤곽선은 상호 참조 하이퍼링크를 표시합니다. 참조 오브젝트를 건너뛰려면 윤곽선 안을 누르십시오.
ProductDir	IBM WebSphere Business Integration Adapters 제품이 설치되는 디렉토리를 나타냅니다. 기본 제품 디렉토리는 WebSphereAdapters입니다.
{ }	구문 행에서 중괄호는 하나만을 선택해야 하는 옵션 세트를 묶습니다.
|	구문 행에서 파이프는 하나만을 선택해야 하는 옵션 세트를 구분합니다.
[ ]	구문 행에서 대괄호는 선택적 매개변수를 묶습니다.
...	구문 행에서 말줄임표(...)는 이전 매개변수의 반복을 표시합니다. 예를 들어 `option[,...]`은 쉼표로 구분되는 여러 옵션을 입력할 수 있음을 의미합니다.
< >	꺽쇠 괄호는 이름의 개별 요소를 묶어 이들을 서로 구별합니다(예: `<server_name><`*`connector_name`*`>tmp.log`).
/, \	이 책에서 백슬래시(\)는 디렉토리 경로에 대한 규약으로 사용됩니다. UNIX 설치의 경우 백슬래시를 슬래시(/)로 대체하십시오. 모든 제품 경로 이름은 Connector for JDBC가 시스템에 설치된 디렉토리에 관련됩니다.
UNIX:/Windows:	이들 중 하나로 시작하는 단락은 운영 체제 차이점을 나열하는 참고를 표시합니다.
%text% 및 *$text*	퍼센트(%) 부호 내의 텍스트는 Windows `text` 시스템 변수 또는 사용자 변수를 표시합니다. UNIX 환경의 해당 표기법은 *`text`* UNIX 환경 변수의 값을 표시하는 *`$text`*입니다.

이 릴리스의 새로운 기능

릴리스 1.0의 새로운 기능

버전 1.0은 *Adapter for Manugistics Demand & Fulfillment Management 사용자 안내서*의 첫 번째 릴리스입니다.

제 1 장 커넥터 개요

커넥터는 **커넥터 프레임워크** 및 **응용프로그램 특정 구성요소**로 구성됩니다. 코드가 모든 커넥터에 공통인 커넥터 프레임워크는 통합 브로커와 응용프로그램 특정 구성요소의 매개체로 작동합니다. 응용프로그램 특정 구성요소는 특정 응용프로그램이나 기술에 맞게 조정된 코드를 포함합니다(이 경우 JDBC). 커넥터 프레임워크는 통합 브로커와 응용프로그램 특정 구성요소 사이에 다음 서비스를 제공합니다.

- Business Object 수신 및 전송
- 시작 및 관리 메시지의 교환 관리

이 장에서는 IBM WebSphere Business Integration Adapter for Manugistics Demand & Fulfillment Management의 커넥터 구성요소에 대해 설명합니다. 이 문서에는 커넥터 프레임워크와 응용프로그램 특정 구성요소 모두에 대한 정보가 포함되어 있습니다. 여기에서 두 가지를 모두 커넥터라고 합니다. 통합 브로커와 커넥터의 관계에 대한 자세한 정보는 *IBM WebSphere InterChange Server System Administration Guide* 또는 *IBM WebSphere Business Integration Adapters Implementation Guide for WebSphere MQ Integrator Broker*를 참조하십시오.

이 장은 다음 섹션으로 구성됩니다.

커넥터 구성요소

Connector for Manugistics Demand & Fullfillment Management를 사용하면 JDBC 2.0 이상의 스펙을 따르는 드라이버가 지원하는 데이터베이스에 빌드된 응용프로그램을 Business Object와 교환할 수 있습니다. 이 섹션에서는 커넥터 구조 및 다른 JDBC 드라이버 사용에 대한 상위 레벨에 대해 설명합니다.

커넥터가 데이터베이스에 연결하기 위해 사용할 드라이버를 지정하려면 17 페이지의 『다중 드라이버 지원 사용 가능』을 참조하십시오

커넥터는 JDBC 연결 메커니즘을 사용하여 응용프로그램 데이터베이스에 연결합니다. 하나의 커넥터 특정 구성 매개변수(DatabaseURL)를 사용하여 커넥터가 연결해야 하는 데이터베이스 서버의 이름을 지정할 수 있습니다. 구성 매개변수에 대한 정보는 18 페이지의 『커넥터 구성』을 참조하십시오

커넥터가 시작되면 데이터베이스와의 연결 풀을 설정합니다. 커넥터는 모든 데이터베이스 트랜잭션 처리에 이 풀 연결을 사용합니다. 커넥터가 종료되면 풀의 모든 연결이 종료됩니다.

커넥터 구조

그림 1에서는 비즈니스 통합 시스템 내의 커넥터 구성요소와 해당 관계를 보여 줍니다.

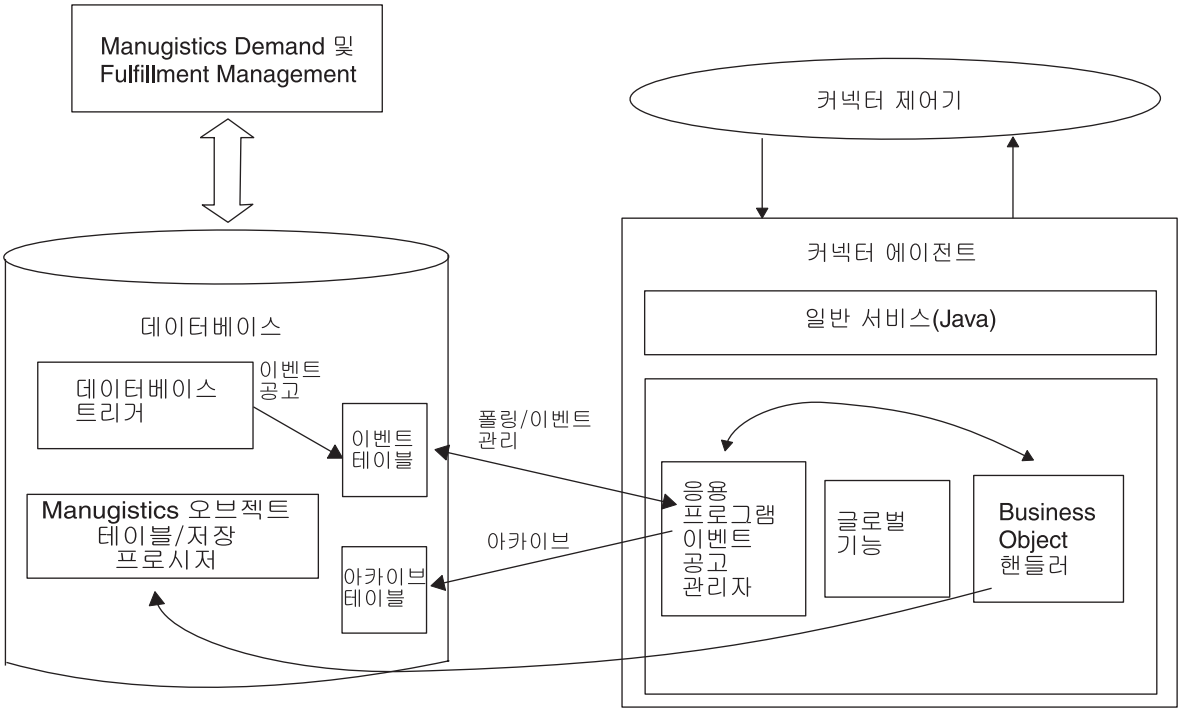


그림 1. Business Object 요청 구조

커넥터 작동 방법

이 섹션에서는 메타 데이터가 커넥터의 유연성을 개선하는 방법에 대해 설명하고 Business Object 처리와 이벤트 공고의 상위 레벨 설명을 보여 줍니다.

커넥터 및 메타 데이터

커넥터는 메타 데이터로 구동됩니다. IBM WebSphere Business Integration Adapter 환경의 **메타 데이터**는 Business Object에 저장되고 응용프로그램과의 상호 작용에서 커넥터를 지원하는 응용프로그램 특정 데이터입니다. 메타 데이터 구동 커넥터는 커넥터에 하드 코딩된 명령어가 아닌 Business Object 정의에 인코딩된 메타 데이터에 기반하여 지원하는 각 Business Object를 처리합니다.

Business Object 메타 데이터는 Business Object의 구조, 속성 등록 정보의 설정 및 응용프로그램 특정 정보의 내용을 포함합니다. 커넥터가 메타 데이터 구동 방식이기 때문에 이는 커넥터 코드를 수정할 필요 없이 수정되거나 새 Business Object를 처리할 수 있습니다.

커넥터는 SQL 문과 저장 프로시저를 실행하여 데이터베이스/응용프로그램에서 데이터를 검색하거나 변경합니다. 동적 SQL 문 또는 저장 프로시저를 빌드하기 위해 커넥터는 응용프로그램 특정 메타 데이터를 사용합니다. 이들 SQL 문과 저장 프로시저는 Business Object 및 커넥터가 처리 중인 Verb에 대한 데이터베이스/응용프로그램에서 필수 검색이나 변경을 수행합니다. 응용프로그램 특정 정보 사용에 대한 정보는 37 페이지의 제 3 장 『커넥터의 Business Object 이해』를 참조하십시오.

Business Object 처리

이 섹션에서는 커넥터가 Business Object 요청과 응용프로그램 이벤트를 처리하는 방법의 개요를 제공합니다. 자세한 정보는 43 페이지의 『Business Object Verb 처리』를 참조하십시오.

Business Object 요청 처리

커넥터가 응용프로그램 조작을 수행하는 요청을 수신할 때 커넥터는 계층 구조 Business Object를 반복적으로 처리합니다. 즉, 이는 모든 개별 Business Object를 처리할 때까지 각 하위 Business Object에 동일한 단계를 수행합니다. 커넥터가 하위 Business Object 및 상위 레벨 Business Object를 처리하는 순서는 하위 Business Object가 소유권을 포함하는지 여부 또는 단일 카디널리티나 다중 카디널리티로 포함되는지 여부에 따라 다릅니다.

주: **계층 구조** Business Object는 모든 레벨에 있는 모든 하위 Business Object를 포함하여 완전한 Business Object를 말하는 것입니다. **개별** Business Object는 포함하거나 포함될 수 있는 하위 Business Object와 별개인 단일 Business Object를 말합니다. **최상위 레벨** Business Object는 그 자체에 상위 Business Object가 없는 계층 구조 맨 위의 개별 Business Object를 말합니다.

Business Object 검색: 커넥터가 데이터베이스에서 계층 구조 Business Object를 검색하도록 통합 브로커가 요청할 때 커넥터는 이 Business Object의 현재 데이터베이스 표시와 정확히 일치하는 Business Object를 리턴하려 시도합니다. 즉, 통합 브로커에 리턴된 각 개별 Business Object의 모든 단순 속성은 데이터베이스에 있는 해당 필드의 값에 일치합니다. 또한 리턴된 Business Object에 의해 포함된 각 배열에 있는 개별 Business Object의 수는 이 배열에 대한 데이터베이스에 있는 하위의 수와 일치합니다.

이러한 검색을 수행하기 위해 커넥터는 최상위 레벨 Business Object의 1차 키 값을 사용하여 데이터베이스에서 반복적으로 해당 데이터를 찾아 아래로 내려옵니다.

Business Object RetrievalByContent: 커넥터가 최상위 레벨 Business Object의 키가 아닌 속성에 기반하여 계층 구조 Business Object를 검색하도록 통합 브로커가 요청할 때 커넥터는 널값이 아닌 모든 속성의 값을 데이터 검색의 기준으로 사용합니다.

Business Object 작성: 커넥터가 데이터베이스에서 계층 구조 Business Object를 작성하도록 통합 브로커가 요청할 때 커넥터는 다음 단계를 수행합니다.

1. 소유권과 함께 포함된 각각의 단일 카디널리티 하위 Business Object를 데이터베이스에 반복적으로 작성합니다.
2. 소유권 없이 포함된 각각의 단일 카디널리티 하위 Business Object를 처리합니다.
3. 데이터베이스에서 최상위 레벨 Business Object를 작성합니다.
4. 하위에 상위/하위 관계를 저장하는 각각의 단일 카디널리티 하위 Business Object를 작성합니다.
5. 각 다중 카디널리티 하위 Business Object를 작성합니다.

Business Object 수정: 커넥터가 데이터베이스에서 계층 구조 Business Object를 갱신하도록 통합 브로커가 요청할 때 커넥터는 다음 단계를 수행합니다.

1. 소스 Business Object의 1차 키 값을 사용하여 데이터베이스에서 해당 엔티티를 검색합니다.
2. 최상위 레벨 Business Object의 모든 단일 카디널리티 하위를 반복적으로 갱신합니다.
3. 상위에 관계를 저장하는 단일 카디널리티 하위 Business Object의 경우 상위의 각 외부 키 값을 해당 단일 카디널리티 하위 Business Object의 1차 키의 값으로 설정합니다.
4. 소스 Business Object의 해당 속성이 CxIgnore 값을 포함하는 속성을 제외하고 검색된 Business Object의 모든 단순 속성을 갱신합니다.
5. 상위/하위 관계를 하위에 저장하는(다중 카디널리티와 단일 카디널리티 모두) 각 하위의 모든 외부 키 값을 해당 상위 Business Object의 1차 키 값으로 설정합니다.
6. 검색된 Business Object의 배열을 모두 처리합니다.

Business Object 삭제: 커넥터가 데이터베이스에서 계층 구조 Business Object를 삭제하도록 통합 브로커가 요청할 때 커넥터는 다음 단계를 수행합니다.

1. 단일 카디널리티 하위를 삭제합니다.
2. 다중 카디널리티 하위를 삭제합니다.
3. 최상위 레벨 Business Object를 삭제합니다.

응용프로그램 이벤트 처리

커넥터는 아래에 설명된 방식으로 응용프로그램이 생성한 Create, Update 및 Delete 이벤트를 처리합니다.

Create 공고: 커넥터가 이벤트 테이블에서 Create 이벤트를 만날 때 이는 이벤트에 의해 지정된 유형의 Business Object를 작성하고 Business Object에 대한 키 값을 설정한 다음(이벤트 테이블에 지정된 키 사용) 데이터베이스에서 Business Object를 검색합니다. Business Object를 검색한 후 커넥터는 Create Verb와 함께 이를 통합 브로커에 전송합니다.

Update 공고: 커넥터가 이벤트 테이블에서 Update 이벤트를 만날 때 이는 이벤트에 의해 지정된 유형의 Business Object를 작성하고 Business Object에 대한 키 값을 설정한 다음(이벤트 테이블에 지정된 키 사용) 데이터베이스에서 Business Object를 검색합니다. Business Object를 검색한 후 커넥터는 Update Verb와 함께 이를 통합 브로커에 전송합니다.

Delete 공고: 커넥터가 이벤트 테이블에서 Delete 이벤트를 만날 때 이는 이벤트에 의해 지정된 유형의 Business Object를 작성하고 Business Object에 대한 키 값을 설정한 다음(이벤트 테이블에 지정된 키 사용) 이를 Delete Verb와 함께 통합 브로커에 전송합니다. 키 값이 아닌 모든 값은 CxIgnore로 설정됩니다. 키가 아닌 필드가 사이트에 중요한 경우 필요한 대로 필드의 값을 수정하십시오.

커넥터는 응용프로그램에 의해 트리거되는 논리 및 실제 Delete 조작을 처리합니다. 실제 삭제의 경우 Delete 이벤트를 이벤트 테이블에 삽입하기 전에 SmartFiltering 메커니즘이 모든 Business Object의 처리되지 않은 이벤트(예: Create 또는 Update)를 제거합니다. 논리 삭제의 경우 커넥터는 다른 Business Object 이벤트를 제거하지 않고 이벤트 테이블에 Delete 이벤트를 삽입합니다.

이벤트 처리를 위한 Business Object 검색: 이벤트 처리를 위한 Business Object의 검색은 두 가지 방법으로 수행될 수 있습니다. 첫 번째 방법은 Business Object의 키 속성에 따른 검색입니다. 두 번째 방법은 키 속성 및 키 속성이 아닌 속성에 따른 검색입니다. 이런 경우 Business Object는 `RetrieveByContent` Verb를 지원해야 하며 오브젝트 키에 `name_value` 쌍을 사용해야 합니다.

주: 오브젝트 키가 `name_value` 쌍을 사용하지 않는 경우 오브젝트 키 필드의 키는 Business Object의 키와 동일한 순서를 따라야 합니다.

이벤트 공고

커넥터의 이벤트 감지 메커니즘은 이벤트 테이블, 아카이브 테이블, 저장 프로시저 및 데이터베이스 트리거를 사용합니다. 이벤트 처리와 연관된 잠재적 장애 지점이 있기 때문에 이벤트 관리 프로세스는 이벤트가 아카이브 테이블에 삽입되어야 이벤트 테이블에서 이벤트를 삭제합니다.

데이터베이스 트리거는 관련있는 이벤트가 데이터베이스에 발생할 때마다 이벤트 테이블을 채웁니다. 커넥터는 정기적이고 구성 가능한 간격으로 이 테이블을 폴링하고 이벤

트를 검색하며 먼저 우선순위에 의해 이벤트를 처리한 후 순차적으로 이벤트를 처리합니다. 커넥터가 이벤트를 처리하면 이벤트의 상태가 갱신됩니다.

주: 트리거를 설치 프로시저의 일부로 데이터베이스에 추가해야 합니다.

ArchiveProcessed 등록 정보의 설정은 커넥터가 상태를 갱신한 후 이벤트를 아카이브 테이블에 아카이브하는지 여부를 판별합니다. ArchiveProcessed 등록 정보에 대한 자세한 정보는 18 페이지의 『커넥터 구성』을 참조하십시오.

표 1에서는 ArchiveProcessed 등록 정보의 설정에 따른 아카이브 작동에 대해 설명합니다.

표 1. 아카이브 작동

처리된 설정 아카이브	이벤트 테이블에서 삭제된 이유	커넥터 작동
true 또는 값이 없음	처리됨	InterChange에 전송 상태로 아카이브됨
	처리되지 못함	오류 상태로 아카이브됨
	Business Object에 대한 등록이 없음	미등록 상태로 아카이브됨
false	처리됨	이벤트 테이블에서 아카이브 및 삭제되지 않음
	처리되지 못함	오류 상태로 이벤트 테이블에 남아 있음
	Business Object에 대한 등록이 없음	이벤트 테이블에 미등록 상태로 남음

SmartFiltering은 통합 브로커와 커넥터가 수행해야 하는 처리의 수량을 최소화하는 데이터베이스 트리거 내의 메커니즘입니다. 예를 들어 커넥터가 이벤트를 마지막으로 폴링한 후 응용프로그램이 Contract Business Object를 15번 갱신한 경우 SmartFiltering은 해당 변경사항을 단일 Update 이벤트로 저장합니다.

끊어진 데이터베이스 연결 처리

데이터베이스 연결이 끊어지는 것에는 많은 원인이 있습니다. 데이터베이스 연결이 끊어지면 커넥터는 종료됩니다. JDBC 스펙은 연결이 끊어지는 것을 감지하기 위한 메커니즘을 제공하지 않습니다. 이 커넥터는 다양한 데이터베이스를 지원하므로 데이터베이스 연결이 끊어진 것에 대한 단일 오류 코드 정의는 없습니다.

이 감지를 처리하기 위해 PingQuery 등록 정보가 제공됩니다. 서비스 호출 요청 중 장애가 발생하는 경우 커넥터는 이 PingQuery를 실행하여 장애의 원인이 데이터베이스 연결이 끊어졌기 때문이 아닌지 확인합니다. PingQuery가 실패하고 AutoCommit 등록 정보가 false로 설정되면 커넥터가 데이터베이스로 새로 연결을 작성하려고 시도합니다. 커넥터는 데이터베이스 연결을 새로 작성하는 데 성공하면 처리를 계속하고 그렇지 않은 경우에는 APPRESPONSETIMEOUT을 리턴하고 결국 커넥터가 종료됩니다.

트랜잭션 유형의 데이터베이스에 액세스할 때 장애가 발생하는 경우 PingQuery가 실행됩니다. 예를 들어 다음과 같습니다.

- 이벤트 및 아카이브 테이블에 액세스하는 동안

- 이벤트에 관련된 Business Object를 검색하는 동안
- Business Object에 관련된 레코드를 작성 또는 갱신하는 동안

로케일 종속 데이터 처리

2바이트 문자 세트를 지원하고 지정된 언어로 메시지 텍스트를 제공할 수 있도록 커넥터는 국제화되어 있습니다. 커넥터가 데이터를 하나의 문자 코드 세트를 사용하는 위치로부터 다른 코드 세트를 사용하는 위치로 전송하는 경우 문자 변환을 수행하여 데이터의 의미를 유지합니다.

JVM(Java 가상 시스템) 내의 Java 런타임 환경은 Unicode 문자 코드 세트로 데이터를 표현합니다. Unicode에는 가장 잘 알려진 문자 코드 세트(단일 바이트 및 다중 바이트 모두)의 문자에 대한 인코딩이 들어 있습니다. WebSphere Business Integration System에 있는 대부분의 구성요소는 Java로 작성된 것입니다. 그러므로 대부분의 WebSphere Business Integration System 구성요소 사이에서 데이터를 전송할 때 문자 변환이 필요없습니다.

해당 국가나 지역에 대해 해당 언어로 오류 및 정보 메시지를 로그하려면 사용자 환경의 로케일 표준 구성 등록 정보를 구성하십시오. 이 등록 정보에 대한 자세한 정보는 99 페이지의 부록 A『커넥터의 표준 구성 등록 정보』를 참조하십시오.

주: 다양한 문자 코드 세트를 사용 가능하게 하려면 Manugistics에서 Manugistics Demand & Fulfillment Managment 문서를 참조하십시오.

제 2 장 커넥터 설치 및 구성

이 장에서는 IBM WebSphere Business Integration Adapter for Manugistics Demand & Fulfillment Management를 설치하고 구성하는 방법과 커넥터에 대한 작업을 수행하기 위해 응용프로그램을 구성하는 방법을 설명합니다. 이 섹션에는 다음이 포함됩니다.

어댑터 환경

어댑터의 설치, 구성 및 사용 전에 환경 요구사항을 이해해야 합니다. 요구사항은 다음 섹션에 나열되어 있습니다.

브로커 호환성

어댑터에 사용되는 어댑터 프레임워크는 어댑터가 통신하고 있는 통합 브로커의 버전과 호환되어야 합니다. Adapter for Manugistics Demand & Fulfillment Management의 버전 1.0.x는 다음 어댑터 프레임워크 및 통합 브로커에서 지원합니다.

- **어댑터 프레임워크:**
  WebSphere Business Integration Adapter Framework 버전 2.3.1 및 2.4.
- **통합 브로커:**
  - WebSphere InterChange Server 버전 4.1.1, 4.2, 4.2.1, 4.2.2
  - WebSphere MQ Integrator 버전 2.1.0
  - WebSphere MQ Integrator Broker 버전 2.1.0

- WebSphere Business Integration Message Broker 버전 5.0
- WebSphere Application Server Enterprise 버전 5.0.2(WebSphere Studio Application Developer Integration Edition 버전 5.0.1 포함)

예외사항은 *릴리스 정보*를 참조하십시오.

주: 통합 브로커 설치 및 전제조건에 대한 지시사항은 다음 안내서를 참조하십시오.

WebSphere ICS(InterChange Server)는 *UNIX용 IBM WebSphere InterChange Server 시스템 설치 안내서* 또는 *Windows용 IBM WebSphere InterChange Server 시스템 설치 안내서*를 참조하십시오.

WebSphere 메시지 브로커는 *Implementing Adapters with WebSphere 메시지 브로커*를 참조하십시오.

WebSphere Application Server는 *Implementing Adapters with WebSphere Application Server*를 참조하십시오.

어댑터 플랫폼

어댑터는 다음 플랫폼에서 지원합니다.

운영 체제:

- AIX 5.1, AIX 5.2
- Solaris 7.0, Solaris 8.0
- HP UX 11i
- Windows 2000

데이터베이스:

- Oracle 9i

제 3 자 소프트웨어:

- Manugistics Demand & Fulfillment Management 버전 7.1.

어댑터 종속성

Oracle 데이터베이스를 사용할 경우 OracleOCI 라이브러리를 설치해야 합니다.

전역화

이 어댑터는 DBCS(2바이트 문자 세트)가 사용 가능하며 변환됩니다.

전제조건

커넥터를 사용하기 전에 다음을 수행하십시오.

- Adapter Development Kit을 설치하십시오.

커넥터가 통합 브로커와 다른 시스템에서 실행되는 경우 통합 브로커의 버전과 호환 가능한 Adapter Development Kit을 설치하십시오.

- 사용할 JDBC 드라이버를 설치하십시오.
- JDBC 드라이버를 포함한 모든 필수 벤더 고유 소프트웨어를 설치했는지 확인하십시오.

  예를 들어 Oracle 데이터베이스용 JDBC 유형 2 드라이버를 사용 중인 경우 OracleOCI 라이브러리를 설치해야 합니다.
- 응용프로그램에 사용자 계정이 있는지 확인하십시오.

  커넥터는 JDBC 스펙에 따르는 드라이버가 지원하는 Manugistics Demand & Fulfillment 데이터베이스의 데이터를 처리합니다. 커넥터가 직접 대화하는 데이터베이스에서 데이터를 처리하려면 응용프로그램에 유효한 사용자 계정과 암호에 대한 액세스가 있어야 합니다. 사용자 계정은 응용프로그램의 데이터베이스에서 데이터를 검색, 삽입, 갱신 및 삭제할 수 있는 특권이 있어야 합니다. 그러한 계정이 아직 없는 경우 작성해야 합니다.
- 연결된 데이터베이스의 문자 코드 세트를 확인하십시오.

  JVM(Java 가상 시스템) 내의 Java 런타임 환경은 Unicode 문자 코드 세트로 데이터를 표현합니다. Unicode에는 가장 잘 알려진 문자 코드 세트(단일 바이트 및 다중 바이트 모두)의 문자에 대한 인코딩이 들어 있습니다. 커넥터는 Java로 쓰여졌으므로 Unicode를 이해합니다.

어댑터 및 관련 파일 설치

WebSphere Business Integration Adapter 제품에 대한 자세한 정보는 다음 사이트의 WebSphere Business Integration Adapters Infocenter에서 *WebSphere Business Integration Adapters 설치 안내서*를 참조하십시오.

http://www.ibm.com/websphere/integration/wbiadapters/infocenter

설치된 파일 구조

다음 섹션에서는 UNIX 또는 Windows 시스템에서 어댑터의 설치된 파일 구조에 대해합니다.

주: 다르게 표시되어 있지 않으면 이 장의 나머지 섹션은 커넥터의 UNIX 및 Windows 설치 모두에 적용됩니다.

UNIX 시스템에 설치

UNIX 시스템에 Adapter for Manugistics Demand & Fulfillment Management 설치하기에 대한 지시사항은 *WebSphere Business Integration Adapters 설치 안내서*를 참조하십시오.

표 2에서는 커넥터가 사용하는 UNIX 파일 구조에 대해 설명합니다.

표 2. 커넥터에 대한 설치된 UNIX 파일 구조

$*ProductDir*의 서브디렉토리	**설명**
connectors/Manugistics	커넥터 BIA_Manugistics.jar 및 start_Manugistics.sh 파일이 포함됩니다. start_Manugistics.sh 파일은 커넥터에 대한 시스템 시작 스크립트입니다. 이는 일반 커넥터 관리자 스크립트에서 호출됩니다. Connector Configurator(WebSphere MQ Integrator를 통합 브로커로 사용하는 경우) 또는 System Manager의 커넥터 구성 화면(ICS를 통합 브로커로 사용하는 경우)에서 설치를 누르면 설치 프로그램이 이 커넥터 관리자 스크립트의 사용자 정의 랩퍼를 작성합니다. 커넥터가 ICS에서 작동할 때 이 사용자 정의 랩퍼를 사용하여 커넥터를 시작하고 중지하십시오. 커넥터가 WebSphere MQ Integrator Broker에서 작동할 때 커넥터를 시작하기 위해서만 이 사용자 정의 랩퍼를 사용하십시오. mqsiremotestopadapter 명령을 사용하여 커넥터를 중지하십시오.
connectors/Manugistics/dependencies	이벤트, 아카이브 및 고유한 ID 테이블을 작성하는 SQL 스크립트를 포함합니다.
connectors/messages	BIA_ManugisticsAdapter.txt 파일을 포함합니다.
repository/Manugistics	BIA_CN_Manugistics.txt 파일을 포함합니다.
/lib	WBIA. jar 파일을 포함합니다.
/bin	CWConnEnv.sh 파일을 포함합니다.

커넥터 구성요소 설치에 대한 자세한 정보는 사용 중인 통합 브로커에 따라 다음 안내서 중 하나를 참조하십시오.

- *UNIX용 시스템 설치 안내서*(통합 브로커로 ICS를 사용할 때)
- *IBM WebSphere Business Integration Adapters Implementation Guide for WebSphere MQ Integrator Broker(통합 브로커로 WebSphere MQ Integrator Broker를 사용할 때)*

Windows 시스템에 설치된 파일 구조

Windows 시스템에 Adapter for Manugistics Demand & Fulfillment Management 설치하기에 대한 지시사항은 *WebSphere Business Integration Adapters 설치 안내서*를 참조하십시오. 표 3에서는 커넥터가 사용하는 Windows 파일 구조에 대해 설명합니다.

표 3. 커넥터에 설치된 Windows 파일 구조

%*ProductDir*%의 서브디렉토리	**설명**
connectors\Manugistics	커넥터 BIA_Manugistics.jar 및 start_Manugistics.bat 파일을 포함합니다.
connectors\Manugistics\dependencies	이벤트, 아카이브 및 고유한 ID 테이블을 작성하는 SQL 스크립트를 포함합니다.
connectors\messages	BIA_ManugisticsAdapter.txt 파일을 포함합니다.
repository\Manugistics	BIA_CN_Manugistics.txt 파일을 포함합니다.

표 3. 커넥터에 설치된 Windows 파일 구조 (계속)

%*ProductDir*%의 서브디렉토리	**설명**
\lib	WBIA. jar 파일을 포함합니다.
\bin	CWConnEnv.bat 파일을 포함합니다.

설치 프로그램은 커넥터 파일에 대한 아이콘을 IBM WebSphere Business Integration Adapters 메뉴에 추가합니다. 커넥터를 빨리 시작하려면 데스크탑에서 이 파일에 대한 바로 가기를 작성하십시오.

커넥터 구성요소 설치에 대한 자세한 정보는 사용 중인 통합 브로커에 따라 다음 안내서 중 하나를 참조하십시오.

- *Windows용 시스템 설치 안내서*(통합 브로커로 ICS를 사용할 때)
- *IBM WebSphere Business Integration Adapters Implementation Guide for WebSphere MQ Integrator Broker*(통합 브로커로 WebSphere MQ Integrator Broker를 사용할 때)

커넥터용으로 Manugistics Demand & Fulfillment Management 사용 가능

이 섹션에서는 다음 주제에 대해 설명합니다.

- 『Manugistics 인터페이스 생성 테이블 구성』
- 14 페이지의 『이벤트 및 아카이브 처리 구성』
- 15 페이지의 『이벤트 및 아카이브 테이블』
- 15 페이지의 『이벤트 및 아카이브 테이블 설치를 위한 SQL 스크립트』
- 16 페이지의 『이벤트 및 아카이브 테이블 스키마』

Manugistics 인터페이스 생성 테이블 구성

Adapter for Manugistics Demand & Fulfillment Management는 IGP(Interface Generation Program)를 사용하여 해당 JDBC 연결 인터페이스 테이블과 데이터베이스에서 데이터를 작성/갱신/삭제 및 검색하는 WebSphere Business Integration Adapters Business Object를 통합합니다. IGP는 다양한 응용프로그램 특정 프로시저의 실행을 통해 실행 중인 데이터베이스 테이블로 데이터를 전달할 때 중간 테이블이 사용하는 프로세스입니다.

데이터베이스의 데이터 값 작성 및 갱신을 요구하는 서비스 호출 요청에 대해서 어댑터는 값의 중간 위치로 IGP의 인터페이스 테이블을 사용합니다. 이 프로세스는 데이터를 인터페이스 테이블에 배치하여 저장 프로시저에 따라 처리한 다음 실행 중인 테이블로 옮기는 방법으로 진행됩니다. 이 저장 프로시저는 데이터 무결성 검사에 제공됩니다.

필요한 통합 테이블 데이터에 대한 삽입, 갱신 및 Upsert 사항을 작성하기 위해서는 먼저 Manugistics 인터페이스 생성 프로그램을 실행하고 구성해야 합니다. 세부사항은 *Manugistics WebWorks Guide*를 참조하십시오.

커넥터가 이벤트 전달을 처리하려면 데이터베이스에 이벤트 공고 메커니즘을 설치해야 합니다. 이를 수행하려면 다음 작업을 완료하십시오.

- 데이터베이스에서 이벤트 및 아카이브 테이블을 작성하십시오.
- 응용프로그램의 테이블에 데이터베이스 트리거를 설치하여 필수 Business Object를 지원하십시오. 사용자가 자신의 데이터베이스 트리거를 개발한다고 가정합니다.
- 선택적으로 카운터 테이블을 설치하십시오. Business Object를 작성할 때 고유 ID를 생성하기 위해 커넥터가 필요한 경우에만 이 단계를 수행하십시오. 고유 ID 생성에 대한 자세한 정보는 `UID=CW.uidcolumnname[=UseIfMissing]` 매개변수를 참조하십시오.

다음 섹션은 이벤트 및 아카이브 테이블 작성과 구성에 대한 정보를 제공합니다.

이벤트 및 아카이브 처리 구성

이벤트 및 아카이브 처리를 구성하려면 구성 등록 정보를 사용하여 다음 정보를 지정해야 합니다.

- 이벤트 테이블의 이름(`EventTableName`). Business Object 요청을 처리하기 위해서만 커넥터를 사용하는 경우 이 등록 정보에 대한 값을 지정할 필요가 없습니다.
- 간격 빈도(113 페이지의 『`PollFrequency`』).
- 각 폴링 간격에 대한 이벤트의 수(`PollQuantity`).
- 아카이브 테이블의 이름(`ArchiveTableName`).
- 커넥터의 미등록 이벤트 및 미처리 이벤트 아카이브 여부(`ArchiveProcessed`). 통합 브로커 고유 등록 정보는 브로커의 구현 안내서를 참조하십시오.
- 복수 커넥터가 동일한 테이블을 폴링할 때 중요한 커넥터의 고유 ID(`ConnectorID`).

또한 `EventOrderBy` 등록 정보 값을 지정하여 처리할 이벤트 순서를 지정할 수도 있습니다. 이 구성 등록 정보 및 다른 구성 등록 정보에 대한 정보는 99 페이지의 부록 A 『커넥터의 표준 구성 등록 정보』 및 19 페이지의 표 6을 참조하십시오.

주: 이벤트 및 아카이브 테이블 작성은 선택적입니다. 그러나 `EventTableName`에 대한 값을 지정하지만 이벤트를 폴링하기 위해 커넥터를 사용하지 않고 이벤트 테이블을 작성하지 않는 경우 커넥터가 시간 종료됩니다. 이러한 제한시간을 차단하려면 `EventTableName`의 값을 `null`(문자열로) 그대로 두십시오.

기본적으로 이벤트 대기열 테이블의 이름은 `xworlds_events`이고 아카이브 대기열 테이블의 이름은 `xworlds_archive_events`입니다.

요청 처리에만 커넥터를 사용하려면 커넥터를 시작할 때 -fno 옵션을 사용하고 EventTableName의 값을 null(문자열로)로 설정하십시오.

사용하고 있는 드라이버가 Java 클래스 DatabaseMetaData를 지원하지 않고 커넥터가 이벤트와 아카이브 테이블의 존재를 검사하지 않도록 하려는 경우 CheckForEventTableInInit의 값을 false로 설정해서 이를 사용 불가능하게 하십시오. 기본값은 true입니다. 값을 false로 설정하지 않는 것이 좋습니다.

주: 사이트가 이벤트를 아카이브 테이블에 아카이브하지 않는 경우 ArchiveProcessed의 값을 false로 설정하십시오.

이벤트 및 아카이브 테이블

커넥터는 이벤트 테이블을 사용하여 선택을 위해 이벤트를 대기열에 넣습니다. ArchiveProcessed 등록 정보를 true 또는 값이 없음으로 설정한 경우 커넥터는 이벤트 테이블에 해당 상태를 갱신한 후 아카이브 테이블을 사용하여 이벤트를 저장합니다.

커넥터는 이벤트 테이블에서 각 이벤트에 대해 Business Object의 이름, Verb 및 키를 가져옵니다. 커넥터는 이 정보를 사용하여 응용프로그램에서 전체 엔티티를 검색합니다. 이벤트를 처음으로 로그한 후 엔티티가 변경된 경우 커넥터는 초기 이벤트 및 모든 후속 변경사항을 가져옵니다. 즉, 이벤트 테이블에서 엔티티를 가져오기 전에 엔티티가 작성 및 갱신된 경우 커넥터는 단일 검색으로 두 가지 데이터 변경사항을 모두 가져옵니다.

커넥터가 처리한 각 이벤트에 대해 다음 세 가지 결과가 나타날 수 있습니다.

- 이벤트가 처리됨
- 이벤트가 처리되지 않음
- 이벤트가 등록되지 않음(통합 브로커 고유 등록 정보는 브로커의 구현 안내서 참조)

커넥터가 이벤트를 픽업한 후 이벤트 테이블에서 이벤트가 삭제되지 않는 경우 불필요한 공간을 차지할 수 있습니다. 그러나 이벤트가 삭제되면, 처리되지 않은 모든 이벤트가 유실되어 이벤트 처리를 감사할 수 없습니다. 따라서 아카이브 테이블도 작성하여 true로 설정된 ArchiveProcessed 등록 정보를 유지하는 것이 좋습니다. 이벤트 테이블에서 이벤트가 삭제될 때마다 커넥터가 아카이브 테이블에 이벤트를 삽입합니다.

주: 응용프로그램 데이터베이스 액세스 문제로 인해 이벤트 테이블에서 이벤트를 삭제하는 중이나 아카이브 테이블에 이벤트를 삽입하는 중 커넥터가 실패하는 경우 커넥터가 APPRESPONSETIMEOUT을 리턴합니다.

이벤트 및 아카이브 테이블 설치를 위한 SQL 스크립트

Oracle 데이터베이스에 대한 이벤트, 아카이브 및 고유 ID 테이블을 설치할 스크립트는 다음과 같습니다.

- event_table_oracle.sql
- event_package_oracle.sql
- archive_table_oracle.sql
- uid_table_oracle.sql

이 파일은 다음 디렉토리에 위치합니다.

UNIX:

connectors/Manugistics/dependencies/

Windows:

connectors\Manugistics\dependencies\

주: 이 스크립트는 커넥터에 대한 필수 테이블을 작성하는 데 도움을 주기 위해 템플리트로서만 제공됩니다. 다른 데이터베이스의 경우 이 스크립트를 지침으로 사용하여 스크립트를 작성하십시오. 테이블 열의 순서 및 데이터 유형이 매우 중요합니다. 올바른 순서 및 유형을 보려면 『이벤트 및 아카이브 테이블 스키마』를 참조하십시오.

특정 설치 및 조회 최적화 요구사항이 충족되도록 DBA 또는 커넥터를 구현하는 개인이 이 스크립트를 수정하는 것이 좋습니다. 예를 들어 이 스크립트는 테이블에서 색인을 작성하지 않습니다. 조회 최적화 프로그램으로 성능을 개선하기 위해 색인을 작성하는 것은 커넥터를 구현하는 개인의 책임입니다.

이벤트 및 아카이브 테이블 스키마

표 4에서는 이벤트 및 아카이브 테이블의 열에 대해 설명합니다.

표 4. 이벤트 및 아카이브 테이블 스키마

이름	설명	유형	제한조건
event_id	이벤트의 내부 ID	NUMBER	1차 키
connector_id	이벤트가 예정된 커넥터의 고유 ID입니다. 이 값은 복수 커넥터가 동일한 테이블을 폴하는 경우에 중요합니다.	VARCHAR	
object_key	Business Object의 1차 키입니다. 키는 name_value 쌍으로 표현되거나 콜론 또는 기타 구성 가능한 분리문자로 분리되는 키 세트(예: 1000065:10056:2333)로 표현이 가능합니다. 자세한 정보는 25 페이지의 『EventKeyDel』 등록 정보를 참조하십시오.	VARCHAR	널(null)이 아님
object_name	Business Object의 이름	VARCHAR	널(null)이 아님
object_verb	이벤트와 연관된 Verb	VARCHAR	널(null)이 아님
event_priority	커넥터가 우선순위에 기초하여 이벤트를 확보하기 위해 사용하는 이벤트 우선순위(0이 최상이고 *n*이 최하임). 커넥터는 우선순위를 낮추거나 올리기 위해 이 값을 사용하지 않습니다.	NUMBER	널(null)이 아님

표 4. 이벤트 및 아카이브 테이블 스키마 (계속)

이름	설명	유형	제한조건
event_time	이벤트 발생 날짜 및 시간	DATETIME	기본 현재 날짜/시간(아카이브 테이블의 경우 실제 이벤트 시간)
archive_time	이벤트가 아카이브된 날짜 및 시간(아카이브 테이블에만 적용됨)	DATETIME	아카이브 날짜/시간
event_status	-2(통합 브로커에 이벤트 전송 중 오류) -1(이벤트 처리 중 오류) 0(폴링 준비) 1(통합 브로커에 전송됨) 2(Business Object에 대한 등록 없음) 3(진행 중). 이 상태는 이벤트 테이블에서만 사용되고 아카이브 테이블에서는 사용되지 않습니다.	NUMBER	널(null)이 아님
event_comment	이벤트 또는 오류 문자열 설명	VARCHAR	

다중 드라이버 지원 사용 가능

다음을 수행하여 드라이버를 지정할 수 있습니다.

1. 시스템에 드라이버를 설치하십시오.
2. 런타임 시 커넥터가 필요로 하는 모든 동적 라이브러리를 제품 디렉토리 아래의 connectors/Manugistics 디렉토리에 두십시오.
3. 커넥터의 시작 파일이 JDBCDRIVERPATH 변수에 있는 모든 관련 클래스 경로 이름을 포함하도록 편집하십시오(필요한 경우 라이센스 정보 포함).

   UNIX에서 시작 파일은 다음과 같습니다.

   ```
   $ProductDir/connectors/Manugistics/start_Manugistics.sh
   ```

   Windows의 시작 파일은 다음과 같습니다.

   ```
   %ProductDir%\connectors\Manugistics\start_Manugistics.bat
   ```

4. JDBCDriverClass 구성 등록 정보에 대한 값을 지정하십시오.

주: 지원하는 모든 기능에 대해 커넥터는 JDBC 2.0 이상 스펙을 따르는 모든 드라이버로 조작할 수 있습니다. 드라이버가 특정 기능을 지원하지 않는 경우에 커넥터는 적절하게 기능하지 않습니다. 예를 들어 드라이버가 ODA for Manugistics에서 사용하는 모든 메소드 호출을 지원하지 않는 경우 ODA 로그는 드라이버가 지원하지 않는 프로세스를 표시합니다. 그런 경우 다른 드라이버를 사용해야 합니다.

사용자 정의 Business Object 핸들러 클래스 사용 가능

커넥터는 사용자 정의 Business Object 핸들러 클래스인 CustomBOH를 지원합니다. 이는 JDBCBOhandlerInterface 인터페이스를 구현합니다. 이 인터페이스의 구문은 다음과 같습니다.

```
public interface JDBCBOhandlerInterface{
   public int doVerbForCustom(CWConnectorBusObj busObj) throws
     VerbProcessingFailedException, ConnectionFailureException;
}
```

doVerbForCustom 메소드를 구현할 때 이 메소드가 두 가지 예외를 발생시키지만 감지하지 못하는지 확인하십시오. 또한 예외가 발생하기 전에 각 예외의 상태 및 메시지를 설정하십시오.

- VerbProcessingFailedException - Verb에 의해 지정된 조작이 실패할 때 발생함.
- ConnectionFailureException - 커넥터가 응용프로그램으로 연결을 설정할 수 없을 때 발생함.

커넥터가 이 Business Object 핸들러를 지원하도록 하려면 다음을 수행하십시오.

- Verb 응용프로그램 특정 정보에 CustomBOH 클래스 이름을 지정하십시오.

  커넥터는 Verb 응용프로그램 특정 정보에서 사용자 정의 Business Object 핸들러 클래스의 이름을 확보합니다. 다음 구문을 사용하십시오.

  ```
  CustomBOH=customBOhandlerClassName
  ```

  예를 들어 Verb 응용프로그램 특정 정보가 다음과 같이 지정되어 있다고 가정하십시오.

  ```
  CustomBOH=JDBCBOhandlerForOverrideSQL
  ```

  이 경우 JDBCBOhandlerForOverrideSQL은 사용자 정의 Business Object 핸들러 클래스의 이름입니다.
- CustomBOH가 com.crossworlds.connectors.JDBC에 속하는지 확인하십시오.

  커넥터가 Verb 응용프로그램 특정 정보에서 "CustomBOH="를 발견하고 com.crossworlds.connectors.JDBC 패키지에서 클래스를 발견하는 경우 사용자 정의 Business Object 핸들러를 실행합니다. 커넥터가 CustomBOH를 찾지 못하는 경우 클래스를 찾을 수 없음을 표시하는 오류가 발생합니다.

커넥터 구성

커넥터의 표준 및 커넥터 특정 구성 등록 정보를 설정해야 커넥터를 실행할 수 있습니다. 다음 도구 중 하나를 사용하여 커넥터의 구성 등록 정보를 설정하십시오.

- Connector Configurator(ICS가 통합 브로커인 경우) - System Manager에서 이 도구에 액세스하십시오.
- Connector Configurator(WebSphere MQ Integrator Broker가 통합 브로커인 경우) - IBM WebSphere Business Integration Adapters 프로그램 폴더에서 이 도구에 액세스하십시오. Connector Configurator에 대한 자세한 정보는 117 페이지의 부록 B『Connector Configurator』를 참조하십시오.

표준 커넥터 등록 정보

표준 구성 등록 정보는 모든 커넥터가 사용하는 정보를 제공합니다. 이 등록 정보에 대한 자세한 정보는 99 페이지의 부록 A『커넥터의 표준 구성 등록 정보』를 참조하십시오.

중요 사항: Connector for JDBC가 ICS 및 WebSphere MQ Integrator Broker 통합 브로커 모두를 지원하기 때문에 이 두 가지 브로커에 대한 구성 등록 정보가 커넥터에 관련됩니다.

또한 IBM WebSphere Business Integration Adapter for JDBC에 특정한 구성 정보는 표 5의 내용을 참조하십시오. 이 테이블의 정보는 부록의 정보를 보충합니다.

표 5. 이 커넥터에 특정한 등록 정보

등록 정보	참고
CharacterEncoding	이 커넥터는 CharacterEncoding 등록 정보를 사용하지 않습니다.
Locale	이 커넥터가 국제화되어 있으므로 로케일 등록 정보의 값을 변경할 수 있습니다. **주:** 브로커로 WebSphere MQ Integrator Broker를 사용하는 경우에는 어댑터, 브로커 및 모든 응용프로그램에 대해 동일한 로케일을 사용해야 합니다.

커넥터를 실행하기 전에 `ApplicationName` 구성 등록 정보에 대한 값을 제공해야 합니다.

커넥터 특정 등록 정보

커넥터 특정 구성 등록 정보는 런타임 시 커넥터에 필요한 정보를 제공합니다. 커넥터별 등록 정보는 커넥터 내의 정적 정보나 논리를 기록하거나 재빌드할 필요 없이 변경하는 방법도 제공합니다.

표 6에서는 커넥터에 대한 커넥터 특정 구성 등록 정보를 나열합니다. 등록 정보 설명은 다음 섹션을 참조하십시오.

표 6. 커넥터 특정 구성 등록 정보

이름	가능한 값	기본값	필수
ApplicationPassword	*커넥터 사용자 계정 암호*		`Yes*`
ApplicationUserName	*커넥터 사용자 계정 이름*		`Yes*`
ArchiveProcessed	`true` 또는 `false`	`true`	`No`
ArchiveTableName	*아카이브 대기열 테이블 이름*	`xworlds_archive_events`	처리된 아카이브가 `true`인 경우 `Yes`
AutoCommit	`true` 또는 `false`	`false`	`No`
CheckforEventTableInInit	`true` 또는 `false`	`true`	`No`
ChildUpdatePhyDelete	`true` 또는 `false`	`false`	`No`
CloseDBConnection	`true` 또는 `false`	`false`	`No`

표 6. 커넥터 특정 구성 등록 정보 (계속)

이름	가능한 값	기본값	필수
ConnectorID	*커넥터의 고유 ID*	null	No
DatabaseURL	*데이터베이스 서버 이름*		Yes
DateFormat	*시간 패턴 문자열*	MM/dd/yyyy HH:mm:ss	No
DriverConnectionProperties	*추가 JDBC 드라이버 연결 등록 정보*		No
EventKeyDel	*이벤트 테이블의 오브젝트 키 열에 대한 분리문자 또는 문자*	semicolon (;)	No
EventOrderBy	none, *ColumnName*, *ColumnName*, ...]		No
EventQueryType	Fixed 또는 Dynamic	Fixed	No
EventTableName	*이벤트 대기열 테이블 이름*	xworlds_events	폴링이 필요한 경우 Yes, 폴링이 필요하지 않은 경우 null(문자열로)
JDBCDriverClass	드라이버 클래스 이름		Yes
MaximumDatabaseConnections	*동시 데이터베이스 연결 수*	5	Yes
PingQuery	SELECT 1 FROM <*tablename*>		No
PollQuantity	값은 1에서 500까지 입니다.	1	No
PreserveUIDSeq	true 또는 false	true	No
RDBMS.initsession	*모든 데이터베이스 세션을 초기화하는 SQL 문*		No
RDBMSVendor	MSSQLServer, Oracle, *기타*		Yes
ReplaceAllStr	true 또는 false	false	No
ReplaceStrList	단일 문자, 문자 분리문자 및 문자의 대체 문자열로 구성된 세트. 또한 이들 사이에 종결 분리문자가 있는 복수 세트.	Q,DSQ 참고: 커넥터 구성 도구에서 이들 문자는 쉼표나 두 개의 작은따옴표 표시 뒤에 오는 작은따옴표 표시를 나타냅니다.	No
RetryCountAndInterval	*계수, 간격(초)*	3,20	No
SchemaName	이벤트가 상주하는 스키마		No
SPBeforePollCall	각 폴 호출에 실행될 저장 프로시저 이름		No
StrDelimiter	ReplaceStrList 등록 정보에 사용되는 문자 및 종결 분리문자	,:	No
TimingStats	0, 1, 2	0	No
UniqueIDTableName	*ID 생성에 사용되는 테이블의 이름*	xworlds_uid	No
UseDefaults	true 또는 false	false	Yes
UseDefaultsForCreatingChildBOs	true 또는 false	false	No
UseDefaultsForRetrieve	true 또는 false	false	No

*신뢰 인증을 사용하는 경우 ApplicationPassword 및 ApplicationUserName이 필요하지 않습니다.

ApplicationPassword
커넥터의 사용자 계정에 대한 암호입니다.

기본값은 없습니다.

ApplicationUserName
커넥터의 사용자 계정 이름입니다.

기본값은 없습니다.

ArchiveProcessed
현재 등록이 없는 이벤트를 커넥터가 아카이브하는지 여부를 지정합니다.

이 등록 정보를 true로 설정하여 이벤트 테이블에서 이벤트가 삭제된 후에 아카이브 테이블에 삽입되지 않도록 하십시오.

이 등록 정보를 false로 설정하여 커넥터가 아카이브 처리를 수행하지 않도록 하십시오. 이 경우 커넥터는 ArchiveTableName 등록 정보의 값을 검사하지 않습니다. ArchiveProcessed가 false로 설정된 경우 커넥터가 다음 작동을 수행합니다.

- 이벤트 처리가 완료되면 커넥터가 이벤트 테이블에서 이벤트를 삭제하고 아카이브하지 않습니다.
- 커넥터가 이벤트의 Business Object에 등록하지 않는 경우 커넥터가 이벤트 테이블에 이벤트를 남겨두고 이벤트 상태를 미등록으로 변경합니다. 통합 브로커 고유 등록 정보는 브로커의 구현 안내서를 참조하십시오.
- Business Object가 처리되는 중 문제가 발생하는 경우 커넥터가 이벤트 테이블에 있는 이벤트를 오류라는 이벤트 상태로 남겨둡니다.

이 등록 정보가 false로 설정되고 폴 수량이 낮은 경우 커넥터가 이벤트 테이블을 폴링하는 것처럼 보이지만 단순히 동일한 이벤트를 반복적으로 선택합니다.

이 등록 정보에 값이 없는 경우 커넥터는 값이 true라고 가정합니다. ArchiveTableName 등록 정보에도 값이 없는 경우 커넥터는 아카이브 테이블의 이름이 xworlds_archive_events라고 가정합니다.

기본값은 true입니다.

ArchiveTableName
아카이브 대기열 테이블의 이름입니다.

ArchiveProcessed 등록 정보를 false로 설정하면 이 등록 정보 값을 설정하지 않아도 됩니다.

기본 이름은 xworlds_archive_events입니다.

AutoCommit

이 등록 정보는 AutoCommit 설정을 구성 가능하게 합니다. true로 설정될 때 모든 트랜잭션이 자동으로 확약됩니다. 일부 데이터베이스(예: Sybase)에서는 AutoCommit를 true로 설정해야 합니다. false로 설정하면 Sybase의 저장 프로시저가 실패합니다.

데이터베이스 연결이 끊어지면 커넥터는 AutoCommit가 false로 설정되어 있는 한 전체 처리를 다시 시작하기 위해 새 연결을 작성하려고 시도합니다. 새 연결이 유효하지 않거나 AutoCommit가 true로 설정된 경우 커넥터는 APPRESPONSETIMEOUT을 리턴하고 결과적으로 커넥터가 종료합니다.

기본값은 false입니다.

CheckforEventTableInInit

이 커넥터 등록 정보를 false로 설정하면 커넥터 초기화 시 커넥터가 이벤트 및 아카이브 테이블 존재 여부를 확인하지 못하게 합니다. 사용 중인 JDBC 드라이버가 JDBC 클래스 DatabaseMetaData를 지원하는 경우 이를 항상 true로 설정하는 것이 좋습니다.

이 등록 정보를 false로 설정하는 경우 커넥터가 이벤트 테이블 및 아카이브 테이블 존재 여부를 확인하지 않더라도 초기화 프로세스 시 커넥터가 이들을 사용하므로 이벤트 및 아카이브 테이블은 언제나 존재해야 합니다. 초기화 시 커넥터가 이벤트 및 아카이브 테이블을 사용하지 못하도록 하려면 EventTableName 등록 정보를 null로 설정하십시오.

기본값은 true입니다.

ChildUpdatePhyDelete

갱신 조작 중 수신 Business Object에서 누락되었지만 데이터베이스에는 존재하는 하위 Business Object에 의해 표시되는 데이터를 커넥터가 처리하는 방법을 지정합니다.

이 등록 정보를 true로 설정하여 커넥터가 데이터베이스에서 데이터 레코드를 실제로 삭제하도록 하십시오.

이 등록 정보를 false로 설정하여 해당 값에 상태 열을 설정해서 커넥터가 데이터베이스에서 데이터 레코드를 논리적으로 삭제하도록 하십시오. 응용프로그램 특정 정보는 Business Object 레벨 응용프로그램 고유 정보에 지정된 SCN(StatusColumnValue) 매개변수에서 해당 값 및 상태 열 이름을 확보합니다. 자세한 정보는 64 페이지의 『Business Object 레벨의 응용프로그램 특정 정보』를 참조하십시오.

기본값은 false입니다.

CloseDBConnection

이 등록 정보는 데이터베이스 연결 닫기를 구성할 수 있게 합니다. 모든 서비스 호출 요청 및 폴 호출에 이 등록 정보를 true로 설정하면 데이터베이스 연결이 종료됩니다. 이 등록 정보를 true로 설정하면 성능에 손상을 줄 수 있으므로 바람직하지 않습니다.

기본값은 false입니다.

ConnectorID

커넥터에 대한 고유 ID입니다. 이 ID는 커넥터의 특정 인스턴스에 대한 이벤트를 검색하는 데 유용합니다.

기본값은 널(null)입니다.

DatabaseURL

커넥터가 연결해야 하는 데이터베이스 서버의 이름입니다.

WebSphere Business Integration System 브랜드의 SQLServer 드라이버를 사용하는 경우 권장 URL은 다음과 같습니다.

jdbc:ibm-crossworlds:sqlserver://*MachineName*:*PortNumber*;DatabaseName=*DBname*

중요 사항

AutoCommit가 false로 설정되어 있는 경우 추가 매개변수인 SelectMethod : jdbc:ibm-crossworlds:sqlserver://*MachineName*:*PortNumber* ;DatabaseName=*DBname*; SelectMethod=cursor를 설정해야 합니다. 기본적으로 SelectMethod 값은 direct로 설정됩니다. 자세한 정보는 22 페이지의 『AutoCommit』를 참조하십시오.

커넥터가 처리를 완료하려면 이 등록 정보에 대한 값을 제공해야 합니다.

DateFormat

커넥터가 수신 및 리턴할 것으로 예상하는 날짜 형식을 지정합니다. 이 등록 정보는 24 페이지의 표 7에 포함된 구문을 기반으로 하는 모든 형식을 지원합니다.

24 페이지의 표 7에서는 시간 패턴 문자열을 사용하여 시간 형식 구문을 정의합니다. 이 패턴에서 모든 ASCII 문자는 패턴 문자로 예약됩니다.

표 7. 시간 형식 구문

기호	의미	표시	예
G	연대 지정자	(Text)	AD
y	연도	(Number)	1996
M	연도의 월	(Text & Number)	July & 07
d	일	(Number)	10
h	am/pm(1 - 12)의 시간	(Number)	12
H	하루의 시간(0 - 23)	(Number)	0
m	시간의 분	(Number)	30
s	분의 초	(Number)	55
S	밀리초	(Number)	978
E	주의 요일	(Text)	Tuesday
D	연도의 일	(Number)	189
F	월 내 주의 일	(Number)	2 (2nd Wed in July)
w	연도의 주	(Number)	27
W	월의 주	(Number)	2
a	am/pm 표시자	(Text)	PM
k	하루의 시간(1 - 24)	(Number)	24
K	am/pm(0 - 11)의 시간	(Number)	0
z	시간대	(Text)	Pacific Standard Time)
'	텍스트 확장	(분리문자)	
''	작은따옴표	(리터럴)	'

표 8. US 로케일을 사용한 예

형식 패턴	결과
"yyyy.MM.dd G 'at' hh:mm:ss z"	1996.07.10 AD at 15:08:56 PDT
"EEE, MMM d, ''yy"	Wed, July 10, ‘96
"h:mm a"	12:08 PM
"hh 'o''clock' a, zzzz"	12 o’clock PM, Pacific Daylight Time
"K:mm a, z"	0:00 PM, PST
"yyyy.MMMMM.dd GGG hh:mm aaa"	1996.July.10 AD 12:08 PM

DriverConnectionProperties

사용자 이름과 암호 외에 JDBC 드라이버에 추가 등록 정보나 정보가 필요할 수 있습니다. DriverConnectionProperties 커넥터 등록 정보는 JDBC 드라이버에 필요한 추가 등록 정보를 이름-값 쌍으로 사용합니다. 등록 정보는 다음과 같이 지정되어야 합니다.

```
property1=value1[;property2=value2...]
```

세미콜론으로 분리된 이름 값 쌍으로 등록 정보를 제공해야 합니다. 등록 정보는 등호 부호(여분의 공백없이)에 의해 값과 분리됩니다.

예를 들어 JDBC 드라이버에 사용권에 관한 정보와 포트 번호가 필요하다고 가정하십시오. 사용권에 관한 정보에 예상되는 등록 정보 이름은 MyLicense이고 값은 ab23jk5입니다. 포트 번호에 예상되는 등록 정보 이름은 PortNumber이고 값은 1200입니다. DriverConnectionProperties는 MyLicense=ab23jk5;PortNumber=1200 값으로 설정되어야 합니다.

EventKeyDel

이벤트 테이블의 object_key 열에 복수 속성 값이 있을 때 분리문자를 지정합니다.

트리거하는 응용프로그램에서 작성, 갱신 또는 삭제된 Business Object를 검색하는 방법에는 두 가지가 있습니다.

- 첫 번째는 Business Object의 키인 속성에 대한 값으로 object_key 열을 채우는 것입니다. EventKeyDel 구성 등록 정보를 키 필드의 일부가 아닌 단일 문자로 설정하십시오. 예를 들어 분리문자가 "**;**"과 같이 지정되어 있는 경우 object_key는 *xxx*;*123*과 같습니다.
- 두 번째 방법은 Business Object의 임의의 속성 값으로 object_key 열을 채우는 것입니다. 이 값은 name_value 쌍으로 표시되어야 합니다. 첫 번째 분리문자는 name_value에 사용되고 두 번째 분리문자는 키에 사용됩니다. 예를 들어 분리문자가 "**=;**"와 같이 지정되어 있는 경우 object_key는 *CustomerName=xxx*;*CustomerId=123*;과 같습니다.

  분리문자가 "**=:**"과 같이 지정되는 경우 object_key는 *CustomerName=xxx*:*CustomerId=123*:과 같습니다.

주: 키 값 정의 순서는 Business Object의 키 속성과 동일한 순서를 따라야 합니다.

중요 사항: 날짜 속성 데이터를 사용하는 경우 속성 데이터에 포함될 수 있으므로 콜론(:) 분리문자를 사용하지 마십시오.

기본값은 세미콜론(;)으로서 name_value 쌍이 아닌 키를 기반으로 합니다.

EventOrderBy

이벤트의 순서화를 끄는지 여부를 지정하거나 기본 순서와 다른 이벤트 처리의 순서를 지정합니다.

기본적으로 각 폴에 있어 커넥터는 해당 PollQuantity 등록 정보에 지정되어 있는 이벤트 수만을 가져와 이벤트 테이블의 event_time 및 event_priority 열에 있는 값에 따라 이벤트 처리 순서를 지정합니다

커넥터가 이벤트의 순서를 정하지 못하게 하려면 이 등록 정보의 값을 none으로 설정하십시오.

커넥터가 이벤트 테이블의 다른 열로 순서를 정하게 하려면 해당 열의 이름을 지정하십시오. 열 이름을 쉼표(,)로 분리하십시오. 이 등록 정보 값을 지정하면 기본값을 겹쳐씁니다.

이 등록 정보의 기본값은 없습니다.

EventQueryType

EventQueryType 등록 정보는 커넥터가 이벤트 테이블에서 이벤트를 검색하기 위해 조회를 동적으로 생성해야 하는지 또는 조회에서 빌드를 사용해야 하는지 여부를 표시하는 데 사용됩니다. 동적으로 생성된 조회의 경우 커넥터가 이벤트 구조를 이벤트 테이블의 열로 맵핑합니다. 테이블 열에서 데이터의 순서는 매우 중요합니다. 올바른 순서를 보려면 16 페이지의 『이벤트 및 아카이브 테이블 스키마』를 참조하십시오.

EventQueryType의 값이 Fixed(문자열로)인 경우 기본 조회가 실행됩니다. Dynamic(문자열로)으로 설정된 경우 『EventTableName』 등록 정보에 지정되는 테이블에서 열 이름을 가져와서 새 조회를 빌드합니다.

이벤트 테이블 열 이름은 변경될 수 있지만 열의 순서와 데이터 유형은 이벤트 테이블 작성 섹션에 지정된 것과 동일합니다. 25 페이지의 『EventOrderBy』는 기본값이나 동적으로 생성된 조회에 추가됩니다.

EventQueryType 등록 정보가 추가되지 않거나 값을 포함하지 않는 경우 기본적으로 Fixed로 설정됩니다.

기본값은 Fixed(문자열로)입니다.

EventTableName

커넥터의 폴링 메커니즘에 사용되는 이벤트 대기열 테이블의 이름입니다.

기본 이름은 xworlds_events입니다.

커넥터에 대한 폴링이 꺼졌을 때 이를 null(문자열로)로 설정하십시오. 이는 이벤트와 아카이브 테이블의 존재에 대한 유효성 검증을 차단합니다.

사용자가 정의한 이벤트 테이블의 경우 event_id가 JDBC 유형인 INTEGER, BIGINT, NUMERIC, VARCHAR 중 하나에 맵핑되는지 확인하십시오.

JDBCDriverClass

드라이버의 클래스 이름을 지정합니다. 특정 JDBC 드라이버를 사용하려면 이 구성 등록 정보에서 드라이버의 클래스 이름을 지정하십시오. 예를 들어 Oracle thin 드라이버를 지정하려면 이 등록 정보의 값을 oracle.jdbc.driver.OracleDriver로 설정하십시오.

자세한 정보는 17 페이지의 『다중 드라이버 지원 사용 가능』 및 32 페이지의 『UseDefaultsForCreatingChildBOs』를 참조하십시오.

기본값은 없습니다.

MaximumDatabaseConnections

허용된 동시 데이터베이스 연결의 최대 수를 지정합니다. 런타임 시 열려 있는 데이터베이스 연결 수는 이 값에 1을 더한 수입니다.

런타임 시 『PreserveUIDSeq』 등록 정보를 false로 설정하는 경우 열려 있는 데이터베이스 연결 수는 이 값에 2를 더한 수입니다.

기본값은 5입니다.

PingQuery

커넥터가 데이터베이스 연결성을 확인하기 위해 실행하는 SQL 문 또는 저장 프로시저를 지정합니다.

다음은 Ping 조회에 사용되는 SQL 문의 예입니다.

```
SELECT 1 FROM <tablename>
```

다음은 Oracle 데이터베이스에 대한 Ping 조회로 사용되는 저장 프로시저 호출(sampleSP)의 예입니다.

```
call sampleSP( )
```

저장 프로시저 호출은 출력 매개변수를 가질 수 없습니다. 데이터베이스가 입력 매개변수를 요구하는 경우 입력값을 Ping 조회의 일부로 지정해야 합니다. 예를 들면 다음과 같습니다.

```
Call checkproc(2)
```

기본값은 없습니다. 자세한 정보는 6 페이지의 『끊어진 데이터베이스 연결 처리』 및 96 페이지의 『응용프로그램에 연결이 끊어짐』을 참조하십시오.

PollQuantity

커넥터가 폴링 간격당 검색하는 데이터베이스 테이블의 행 수입니다. 허용 가능한 값은 1에서 500까지입니다.

기본값은 1입니다.

PreserveUIDSeq

수신 고유 ID 순서가 고유 ID 테이블에서 보존되는지 여부를 지정합니다.

true로 설정되는 경우 목적지 응용프로그램에서 Business Object가 처리될 때까지 고유 ID가 확약되지 않습니다. 고유 ID 테이블에 대한 기타 모든 프로세스의 액세스 시도는 트랜잭션이 확약될 때까지 대기해야 합니다.

false로 설정되는 경우 Business Object가 이를 요청할 때 고유 ID가 확약됩니다. Business Object 처리 및 고유 ID 처리 각각에는 자체 트랜잭션 블록(커넥터에 내부적인)이 있습니다. 고유 ID 테이블에 관련된 트랜잭션에 자체 연결이 있는 경우에만 가능합니다.

주: 이 등록 정보가 커넥터 구성에 추가되지 않는 경우 기본 작동은 이 등록 정보가 추가되어 true로 설정된 경우와 동일합니다. 또한 22 페이지의 『AutoCommit』가 true로 설정된 경우 커넥터는 PreserveUIDSeq가 false로 설정된 경우와 동일한 작동을 실행합니다

런타임 시 27 페이지의 『PreserveUIDSeq』 등록 정보를 false로 설정하는 경우 열려 있는 데이터베이스 연결 수는 이 값에 2를 더한 수입니다.

기본값은 true입니다.

RDBMS.initsession

모든 세션을 데이터베이스로 초기화하는 SQL 문입니다. 커넥터가 조회를 선택하여 시작 시 실행합니다. 이 조회의 리턴값이 없어야 합니다. 등록 정보 이름이 필요하지만 값은 필요하지 않습니다.

기본값은 없습니다.

RDBMSVendor

커넥터가 특수 처리에 사용하는 RDBMS를 지정합니다. Oracle 데이터베이스에 대해서는 이 등록 정보의 값을 Oracle로 설정하십시오.

기본값이 아닌 데이터베이스를 사용 중인 경우 적절한 드라이버가 로드되는지 확인하십시오. 이 등록 정보가 Others로 설정되는 경우 커넥터가 드라이버를 찾아서 사용할 데이터베이스를 판별합니다.

커넥터가 처리를 완료하려면 값이 필요합니다.

기본값은 없습니다.

ReplaceAllStr

커넥터가 ReplaceStrList 등록 정보에서 식별된 각 문자의 모든 인스턴스를 이 등록 정보에 지정된 대체 문자열로 바꾸는지 여부를 지정합니다. 커넥터는 각 속성의 AppSpecificInfo 등록 정보에 있는 ESC=[true|false] 매개변수가 값을 포함하지 않는 경우에만 ReplaceAllStr을 평가합니다. 즉, ESC 매개변수가 지정된 경우 이 값이

ReplaceAllStr 등록 정보에 설정된 값보다 우선합니다. 커넥터가 ReplaceAllStr의 값을 사용하게 하려면 ESC 매개변수를 지정하지 않았는지 확인하십시오.

ReplaceAllStr의 기본값은 false입니다.

주: ESC 매개변수와 ReplaceAllStr 및 ReplaceStrList 등록 정보는 데이터베이스 ESC 문자 기능에 대한 지원을 제공합니다(예: 작은따옴표 ESC). JDBC 드라이버에서 제공한 Prepared 문에서도 동일한 기능이 사용 가능하기 때문에 이들 등록 정보는 커넥터의 이후 릴리스에서 제거됩니다. 현재 커넥터는 JDBC Prepared 문의 사용을 지원합니다.

ReplaceStrList

각각이 대체될 개별 문자, 문자 분리문자 및 대체 문자열로 구성된 하나 이상의 대체 세트를 지정합니다. 속성 AppSpecificInfo 등록 정보의 ESC=[true|false] 매개변수 또는 커넥터의 ReplaceAllStr 등록 정보에 대한 값이 지정된 경우에만 커넥터가 속성의 값에 이 대체를 수행합니다.

주: ESC 매개변수와 ReplaceAllStr 및 ReplaceStrList 등록 정보는 데이터베이스 ESC 문자 기능에 대한 지원을 제공합니다(예: 작은따옴표 ESC). JDBC 드라이버에서 제공한 Prepared 문에서도 동일한 기능이 사용 가능하기 때문에 이들 등록 정보는 커넥터의 이후 릴리스에서 제거됩니다. 현재 커넥터는 JDBC Prepared 문의 사용을 지원합니다.

이 속성에 대한 구문은 다음과 같습니다.

single_char1,*substitution_str1*[:*single_char2*,*substitution_str2*[:...]]

여기서,

single_char	대체할 문자입니다.
single_char	커넥터가 문자를 바꾸기 위해 사용하는 대체 문자열입니다.
single_char	대체하는 문자열에서 대체될 문자를 분리하는 문자 분리문자입니다. 기본적으로 문자 분리문자는 쉼표(,)입니다. StrDelimiter 등록 정보에 첫 번째 분리문자를 설정하여 이 분리문자를 구성할 수 있습니다.
single_char	대체 세트(각각은 대체될 문자, 문자 분리문자 및 대체 문자열로 구성됨)를 분리하는 종결 분리문자입니다. 기본적으로 종결 분리문자는 콜론(:)입니다. StrDelimiter 등록 정보에서 두 번째 분리문자를 설정해서 이 분리문자를 구성할 수 있습니다.

예를 들어 단일 퍼센트 부호(%)를 두 개의 퍼센트 부호(%%)로, 캐럿(^)을 백슬래시와 캐럿(\^)으로 바꾸려 한다고 가정해보십시오. 기본적으로 StrDelimiter는 문자 분리문자로 쉼표(,)를 지정하고 종결 분리문자로 콜론(:)을 지정합니다. 기본 분리문자를 유지하는 경우 다음 문자열을 ReplaceStrList의 값으로 사용하십시오.

```
%,%%:^,\^
```

주: 커넥터 구성 도구의 제한조건은 작은따옴표 표시를 입력하지 못하는 것입니다. 따라서 작은 따옴표를 Q 문자로 표시하고 두 개의 작은 따옴표를 DSQ 문자로 표시해야 합니다. 위의 예에서 작은따옴표 표시(')를 두 개의 작은따옴표 표시('')로 바꾸려면 Q,DSQ:%,%%:^,\^와 같은 표기법을 사용하십시오.

RetryCountAndInterval

갱신 조작 수행 중 데이터를 잠글 수 없을 때 커넥터가 사용해야 하는 초 단위 간격과 시도의 수를 지정합니다.

갱신을 수행하기 전에 커넥터는 갱신에 관련된 행을 잠그고 현재 데이터를 검색하려고 합니다. 커넥터가 행을 잠글 수 없는 경우 이 구성 등록 정보에 지정된 계수와 간격 대로 잠그기를 다시 시도합니다. 여기에 지정된 값 내에서 잠글 수 없는 경우 결국 커넥터는 시간 종료됩니다.

값을 계수, 초 단위 간격 형식으로 지정하십시오. 예를 들어 3,20 값은 20초 간격으로 3번 시도함을 지정합니다.

기본값은 3,20입니다.

SchemaName

이 등록 정보는 해당 특정 스키마 내에서의 이벤트 및 아카이브 테이블 검색을 제한합니다. 이 등록 정보가 추가되지 않는 경우 또는 비어 있는 경우 커넥터는 사용자가 액세스해야 하는 모든 스키마를 검색합니다. 이 SchemaName은 이벤트 및 아카이브 테이블을 액세스하기 위해 조회를 빌드할 때에도 사용됩니다.

Oracle 데이터베이스에서는 스키마 이름을 지원합니다. 특정 정보에 대한 JDBC 드라이버 문서를 참조하십시오.

기본값은 없습니다.

SPBeforePollCall

이 등록 정보는 모든 폴 호출에 실행되는 저장 프로시저의 이름을 지정합니다. SPBeforePollCall 등록 정보에 값(저장 프로시저의 이름)이 있으면 각 폴 호출을 시작할 때 커넥터가 커넥터 등록 정보 ConnectorID 및 PollQuantity의 값을 전달하여 저장 프로시저를 호출합니다. 프로시저는 PollQuantity 열 번호를 갱신하여 커넥터-ID 열을 ConnectorID로 설정합니다. 여기서 status=0이고 커넥터 ID는 null입니다. 이는 커넥터의 로드 균형을 사용 가능하게 합니다.

주: 폴 호출에 너무 빨리 실패한 경우(데이터베이스가 다운되었거나 연결이 끊어짐)에도 커넥터 ID는 계속 설정되어 있습니다. 이로 인해 폴링 중 레코드를 건너뛸 수 있습니다. 따라서 이벤트 테이블의 상태가 0인 모든 레코드에 대해 커넥터 ID를 주기적으로 null로 다시 재설정하는 것이 좋습니다.

StrDelimiter

ReplaceStrList 등록 정보에서 사용할 문자와 종결 분리문자를 지정합니다.

- 문자 분리문자는 대체하는 문자열에서 대체될 문자를 분리합니다. 문자 분리문자는 이 등록 정보 값의 첫 번째(왼쪽) 위치를 차지하며 기본값은 쉼표(,)입니다.
- 종결 분리문자는 대체 세트(각각은 대체될 문자, 문자 분리문자 및 대체 문자열로 구성됨)를 분리합니다. 종결 분리문자는 이 등록 정보 값의 두 번째(오른쪽) 위치를 차지하며 기본값은 콜론(:)입니다.

이 분리문자 중 하나 또는 모두에 대해 사용자 고유값을 지정할 수 있습니다. 이런 경우 분리문자 사이에 공백 또는 다른 문자를 지정하지 마십시오.

기본값은 쉼표 바로 다음의 콜론입니다(,:).

TimingStats

문제를 찾기 위해 커넥터의 각 Verb 조작에 대한 시간을 정하도록 합니다. 사용 가능한 설정은 다음과 같습니다.

0 (시간 통계 없음)

1 (전체 계층 구조 Business Object에 대한 Verb 조작의 입력 및 종료 시에 표시된 시간)

2 (계층 구조 Business Object에서 각각의 개별 Business Object 대한 각 Verb 조작의 입력 및 종료 시에 표시된 시간)

시간 메시지는 추적 메시지가 아닌 로그 메시지입니다. 추적 레벨에 관계 없이 끄고 켤 수 있습니다.

기본값은 0입니다.

UniqueIDTableName

고유 ID 생성에 사용된 최신값이 들어 있는 테이블을 지정합니다. 기본적으로 테이블에는 하나의 열(id)이 있습니다. 테이블을 사용자 정의하여 UID 생성을 필요로 하는 각 속성의 열을 추가할 수 있습니다.

기본값은 xworlds_uid입니다.

UseDefaults

UseDefaults가 true로 설정되거나 설정되지 않는 경우 커넥터는 각 필수 Business Object 속성에 유효한 값이나 기본값이 제공되는지 여부를 검사합니다. 값이 제공되는 경우 작성은 성공하고 그렇지 않으면 실패합니다.

UseDefaults가 false로 설정되는 경우 커넥터는 각 필수 Business Object 속성에 유효한 값이 제공되는지 여부만을 검사합니다. 유효한 값이 제공되지 않는 경우 Create 조작에 실패합니다.

기본값은 false입니다.

UseDefaultsForCreatingChildBOs

UseDefaultsForCreatingChildBOs가 true로 설정되거나 설정되지 않는 경우 커넥터는 각 필수 Business Object 속성에 유효한 값이나 기본값이 제공되는지 여부를 검사합니다. 값이 제공되는 경우 작성은 성공하고 그렇지 않으면 실패합니다.

UseDefaultsForCreatingChildBOs가 false로 설정되는 경우 커넥터는 각 필수 Business Object 속성에 유효한 값이 제공되는지 여부만을 검사합니다. 유효한 값이 제공되지 않는 경우 Create 조작에 실패합니다.

UseDefaultsForRetrieve

폴링의 경우: UseDefaultsForRetrieve가 정의되지 않았는데 true로 설정한 경우 기본값은 데이터베이스에서 검색되어 서버로 전송되기 전에 BO에 설정됩니다.

UseDefaultsForRetrieve가 정의되고 false로 설정한 경우 기본값은 데이터베이스에서 검색되어 서버로 전송되기 전에 BO에 설정되지 않습니다.

요청 처리의 경우: UseDefaultsForRetrieve가 정의되지 않고 false로 설정한 경우 기본값은 데이터베이스에서 검색되어 서버로 전송되기 전에 BO에 설정되지 않습니다.

UseDefaultsForRetrieve가 정의되고 true로 설정한 경우 기본값은 데이터베이스에서 검색되어 서버로 전송되기 전에 BO에 설정됩니다.

다중 커넥터 인스턴스 작성

복수의 커넥터 인스턴스를 작성하는 것은 여러 가지 면에서 사용자 정의 커넥터를 작성하는 방법과 동일합니다. 다음 단계를 수행하여 복수의 커넥터 인스턴스를 작성하여 실행하도록 시스템을 설정할 수 있습니다. 복수의 커넥터 인스턴스를 작성하려면 다음을 수행해야 합니다.

- 커넥터 인스턴스의 새 디렉토리를 작성하십시오.
- 필수 Business Object 정의가 있는지 확인하십시오.
- 새 커넥터 정의 파일을 작성하십시오.

- 새 시작 스크립트를 작성하십시오.

새 디렉토리 작성

각 커넥터 인스턴스의 커넥터 디렉토리를 작성해야 합니다. 다음과 같이 이 커넥터 디렉토리의 이름을 지정해야 합니다.

```
ProductDir\connectors\connectorInstance
```

여기서 connectorInstance는 고유하게 커넥터 인스턴스를 식별합니다.

커넥터에 커넥터 특정 Meta Object가 있는 경우, 커넥터 인스턴스의 Meta Object를 작성해야 합니다. Meta Object를 파일로 저장할 경우, 이 디렉토리를 작성한 후 파일을 다음 디렉토리에 저장하십시오.

```
ProductDir\repository\connectorInstance
```

Business Object 정의 작성

각 커넥터 인스턴스에 대한 Business Object 정의가 아직 프로젝트 내에 존재하지 않으면 이를 정의해야 합니다.

1. 초기 커넥터와 연관된 Business Object 정의를 수정해야 할 경우, 해당 파일을 복사한 다음 Business Object Designer를 사용하여 가져오십시오. 초기 커넥터에 대한 모든 파일을 복사할 수 있습니다. 파일을 변경해야 할 경우 이름만 바꾸십시오.
2. 초기 커넥터용 파일은 다음 디렉토리에 있어야 합니다.

   ```
   ProductDir\repository\initialConnectorInstance
   ```

   작성한 모든 추가 파일은 ProductDir\repository의 적절한 connectorInstance 서브디렉토리에 있어야 합니다.

커넥터 정의 작성

Connector Configurator에서 커넥터 인스턴스의 구성 파일(커넥터 파일)을 작성합니다. 이를 수행하려면 다음 단계를 따르십시오.

1. 초기 커넥터의 구성 파일(커넥터 정의)을 복사한 후 이름을 바꾸십시오.
2. 커넥터 인스턴스에 지원되는 Business Object(및 연관된 Meta Object)가 올바르게 나열되는지 확인하십시오.
3. 커넥터 등록 정보를 적절하게 사용자 정의하십시오.

시작 스크립트 작성

시작 스크립트를 작성하려면 다음을 수행하십시오.

1. 초기 커넥터의 시작 스크립트를 복사한 다음 시작 스크립트 이름에 커넥터 디렉토리의 이름이 포함되도록 이름을 지정하십시오.

   dirname

2. 시작 스크립트를 『새 디렉토리 작성』에서 작성한 커넥터 디렉토리에 배치하십시오.

3. 시작 스크립트 단축 아이콘을 작성하십시오(Windows 전용).
4. 초기 커넥터의 단축 아이콘 텍스트를 복사한 후 새 커넥터 인스턴스의 이름과 일치하도록 초기 커넥터의 이름을 변경하십시오(명령행에서).

이제 통합 서버에서 두 커넥터 인스턴스를 동시에 실행할 수 있습니다.

사용자 정의 사용자 정의 작성에 대한 자세한 정보는 *Connector Development Guide for C++* 또는 *Connector Development for Java*를 참조하십시오.

커넥터 시작

커넥터는 해당 **커넥터 시작 스크립트**를 사용하여 명시적으로 시작해야 합니다. 시작 스크립트는 커넥터의 런타임 디렉토리에 있어야 합니다.

ProductDir\connectors*connName*

여기서 *connName*은 커넥터를 식별합니다. 시작 스크립트의 이름은 표 9에 표시된 대로 운영 체제 플랫폼에 따라 다릅니다.

표 9. 커넥터의 시작 스크립트

운영 체제	시작 스크립트
UNIX 기반 시스템	connector_manager_*connName*
Windows	start_*connName*.bat

다음 방법 중 하나를 사용하여 커넥터 시작 스크립트를 호출할 수 있습니다.

- Windows 시스템의 경우, **시작** 메뉴에서 다음을 수행하십시오.

  프로그램 > IBM WebSphere Business Integration Adapters > 어댑터 > 커넥터를 선택하십시오. 기본적으로 프로그램 이름은 "IBM WebSphere Business Integration Adapters"입니다. 그러나 이 이름을 사용자 정의할 수 있습니다. 대안으로, 커넥터에 대한 데스크탑 단축 아이콘을 작성할 수 있습니다.
- 명령행에서 커넥터를 시작할 경우에는 다음을 수행하십시오.
  - Windows 시스템의 경우:

    start_*connName connName brokerName* [-c*configFile*]
  - UNIX 기반 시스템의 경우:

    connector_manager_*connName* -start

  여기서 *connName*은 커넥터의 이름이며 *brokerName*은 다음과 같이 통합 브로커를 식별합니다.
  - WebSphere InterChange Server의 경우 *brokerName*에 ICS 인스턴스의 이름을 지정하십시오.

– WebSphere 메시지 브로커(WebSphere MQ Integrator, WebSphere MQ Integrator Broker 또는 WebSphere Business Integration Message Broker) 또는 WebSphere Application Server의 경우, *brokerName*에 브로커를 식별하는 문자열을 지정하십시오.

주: Windows 시스템의 WebSphere 메시지 브로커 또는 WebSphere Application Server에서는 `-c` 옵션 다음에 커넥터 구성 파일의 이름이 *포함되어야* 합니다. ICS의 경우 `-c`는 선택적입니다.

- System Manager를 시작할 때 실행되는 Adaptor Monitor(WebSphere Business Integration Adapters 제품 전용)에서 다음을 수행할 수 있습니다.

  이 도구를 사용하여 커넥터를 로드, 활성화, 비활성화, 일시정지, 종료 또는 삭제할 수 있습니다.

- System Monitor(WebSphere InterChange Server 제품 전용)에서 다음을 수행할 수 있습니다.

  이 도구를 사용하여 커넥터를 로드, 활성화, 비활성화, 일시정지, 종료 또는 삭제할 수 있습니다.

- Windows 시스템에서 Windows 서비스로 시작하도록 커넥터를 구성할 수 있습니다. 이 경우, Windows 시스템이 시동하거나(자동 서비스인 경우) Windows 서비스 창을 통해 서비스를 시작할 때(수동 서비스인 경우) 커넥터가 시작됩니다.

명령행 시작 옵션을 포함하여 커넥터를 시작하는 방법에 대한 자세한 정보는 다음 문서 중 하나를 참조하십시오.

- WebSphere InterChange Server에 대한 자세한 정보는 *System Administration Guide*를 참조하십시오.
- WebSphere 메시지 브로커에 대한 자세한 정보는 *Implementing Adapters with WebSphere 메시지 브로커*를 참조하십시오.
- WebSphere Application Server에 대한 자세한 정보는 *Implementing Adapters with WebSphere Application Server*를 참조하십시오.

케넥터 중지

커넥터를 중지하는 방법은 다음과 같이 커넥터를 시작한 방법에 따라 다릅니다.

- 해당 커넥터 시작 스크립트를 사용하여 명령행에서 커넥터를 시작한 경우에는 다음을 수행하십시오.
  - Windows 시스템의 경우, 시작 스크립트를 호출하면 커넥터에 대한 별도의 "콘솔" 창이 작성됩니다. 이 창에서 "Q"를 입력한 다음 Enter를 눌러 커넥터를 중지하십시오.
  - UNIX 기반 시스템에서, 커넥터는 백그라운드에서 실행되므로 별도의 창이 없습니다. 대신 커넥터를 중지하려면 다음 명령을 실행하십시오.

```
connector_manager_connName -stop
```

여기서 *connName*은 커넥터의 이름입니다.

- System Manager를 시작할 때 실행되는 Adapt Monitor(WebSphere Business Integration Adapters 제품 전용)에서 다음을 수행할 수 있습니다.

  이 도구를 사용하여 커넥터를 로드, 활성화, 비활성화, 일시정지, 종료 또는 삭제할 수 있습니다.
- System Monitor(WebSphere InterChange Server 제품 전용)에서 다음을 수행할 수 있습니다.

  이 도구를 사용하여 커넥터를 로드, 활성화, 비활성화, 일시정지, 종료 또는 삭제할 수 있습니다.
- Windows 시스템에서 Windows 서비스로 시작하도록 커넥터를 구성할 수 있습니다. 이 경우, Windows 시스템이 종료되면 커넥터가 중지됩니다.

제 3 장 커넥터의 Business Object 이해

이 장에서는 JDBC의 커넥터가 Business Object를 처리하는 방법과 데이터를 검색하고 수정할 때 커넥터에 의해 이루어지는 가정에 대해 설명합니다. 다음 섹션을 포함합니다.

- 『Business Object 및 속성 이름 지정 규칙』
- 『Business Object 구조』
- 43 페이지의 『Business Object Verb 처리』
- 60 페이지의 『Business Object 속성 등록 정보』
- 62 페이지의 『Business Object 응용프로그램 특정 정보』

이 책을 기존 Business Object를 수정하기 위한 안내서로 사용하거나 새 Business Object를 구현하기 위한 제안사항으로 사용할 수 있습니다. 데이터베이스 테이블에서 WebSphere Business Integration Adapter Business Object 정의 파일의 작성을 자동화하는 유틸리티에 대한 정보는 75 페이지의 제 4 장 『Manugistics용 IBM ODA를 사용한 Business Object 정의 생성』을 참조하십시오.

커넥터에서는 지원되는 Business Object 구조, 상위 Business Object와 하위 Business Object의 관계, 응용프로그램 특정 정보 형식 및 Business Object의 데이터베이스 표시에 대해 가정합니다. 따라서 커넥터가 처리할 Business Object를 작성하거나 수정할 때 수정사항이 커넥터가 따르도록 설계된 규칙을 준수해야 합니다. 그렇지 않으면 커넥터는 새 Business Object 또는 수정된 Business Object를 올바르게 처리할 수 없습니다.

Business Object 및 속성 이름 지정 규칙

커넥터가 사용하는 Business Object의 이름은 영숫자 문자나 밑줄 문자로만 구성될 수 있습니다. Business Object 속성 이름도 영숫자 문자나 밑줄 문자로만 구성될 수 있습니다.

Business Object 구조

대부분의 경우에 커넥터는 하나의 데이터베이스 테이블 또는 보기에 의해 모든 개별 Business Object가 표시되고 오브젝트 내의 각 **단순 속성**(즉, `String`, `Integer` 또는 `Date`와 같이 단순 값을 나타내는 속성)은 이 테이블이나 보기의 열에 의해 표시된다고 가정합니다. 따라서 동일한 개별 Business Object 내의 속성을 다른 데이터베이스 테이블에 저장할 수 없습니다. 그러나 다음 상황은 가능합니다.

- 해당 개별 Business Object에 있는 단순 속성보다 데이터베이스 테이블에 더 많은 열이 있습니다(즉, 데이터베이스의 일부 열이 Business Object에 표시되지 않음). 사용자 설계 시 Business Object 처리에 필요한 열만을 포함하도록 하십시오.
- 해당 데이터베이스 테이블에 있는 열보다 개별 Business Object에 더 많은 단순 속성이 있습니다(즉, Business Object의 일부 속성이 데이터베이스에 표시되지 않음). 데이터베이스에 표시되지 않은 속성은 응용프로그램 특정 정보를 가지고 있지 않거나 기본값으로 설정되거나 또는 저장 프로시저를 지정합니다.
- 개별 Business Object는 복수 데이터베이스 테이블을 확장하는 보기를 표시할 수 있습니다. 커넥터는 응용프로그램에서 트리거되는 Create, Retrieve, Update 및 Delete 이벤트를 처리할 때 이런 Business Object를 사용할 수 있습니다. 그러나 Business Object 요청을 처리할 때 커넥터는 검색 요청에만 이런 Business Object를 사용할 수 있습니다.
- 각 Business Object는 관련 없는 Business Object의 컨테이너로 사용되는 랩퍼 오브젝트를 표시할 수 있습니다. 랩퍼 오브젝트는 데이터베이스 테이블이나 보기로 표시되지 않습니다. 랩퍼 오브젝트는 다른 오브젝트의 하위로 사용되지 않을 수 있습니다.

주: Business Object가 저장 프로시저를 기반으로 하는 경우 각 단순 속성(특수 SP 속성 외의)에는 응용프로그램 특정 정보가 있거나 또는 없을 수 있습니다. 자세한 정보는 52 페이지의 『저장 프로시저』를 참조하십시오.

WebSphere Business Integration Adapters Business Object는 플랫 구조 또는 계층 구조 형식일 수 있습니다. **플랫 구조** Business Object의 모든 속성은 단순하며 단일 값을 표시합니다.

계층 구조 Business Object는 단일 하위 Business Object, 하위 Business Object의 배열 또는 이 둘의 조합을 표시하는 속성이 있습니다. 반대로 각 하위 Business Object에는 하위 Business Object 또는 Business Object의 배열 등이 있을 수 있습니다. 상위 Business Object의 속성이 단일 하위 Business Object를 표시할 때 **단일 카디널리티 관계**가 발생합니다. 이 경우 속성은 하위 Business Object와 동일한 유형입니다.

상위 Business Object의 속성이 하위 Business Object의 배열을 표시할 때 **다중 카디널리티 관계**가 발생합니다. 이 경우 속성은 하위 Business Object와 동일한 유형의 배열입니다.

주: 계층 구조 Business Object는 모든 레벨에 있는 모든 하위 Business Object를 포함하여 완전한 Business Object를 말하는 것입니다. **개별** Business Object는 포함하거나 포함될 수 있는 하위 Business Object와 별개인 단일 Business Object를 말합니다. **최상위 레벨** Business Object는 그 자체에 상위 Business Object가 없는 계층 구조 맨 위의 개별 Business Object를 말합니다.

커넥터는 Business Object 사이의 다음 관계를 지원합니다.

- 『단일 카디낼리티 관계』
- 『단일 카디낼리티 관계 및 소유권이 없는 데이터』
- 41 페이지의 『다중 카디낼리티 관계』
- 41 페이지의 『하위에 관계를 저장하는 단일 카디낼리티 관계』
- 42 페이지의 『랩퍼 오브젝트』

각 유형의 카디낼리티에서 상위와 하위 Business Object 사이의 관계는 관계를 저장하는 Business Object의 키 속성의 응용프로그램 특정 정보에서 설명됩니다. 이 응용프로그램 특정 정보에 대한 자세한 내용은 표 11의 `FK=[fk_object_name.]fk_attribute_name`을 참조하십시오.

단일 카디낼리티 관계

일반적으로 단일 카디낼리티 하위 Business Object를 포함하는 Business Object에는 관계를 표시하는 최소 두 개의 속성이 있습니다. 한 속성의 유형은 하위의 유형과 동일합니다. 다른 속성은 하위의 1차 키를 상위에서 외부 키로 포함하는 단순 속성입니다. 상위에는 하위에 있는 1차 키 속성 만큼의 외부 키 속성이 있습니다.

관계를 설정하는 외부 키가 상위에 저장되기 때문에 각 상위는 제공된 유형의 단일 카디낼리티 하위 하나만을 포함할 수 있습니다.

그림 2는 일반적인 단일 카디낼리티 관계를 보여 줍니다. 예에서 `fk1`은 하위의 1차 키를 포함하는 단순 속성이고 `child[1]`은 하위 Business Object를 표시하는 속성입니다.

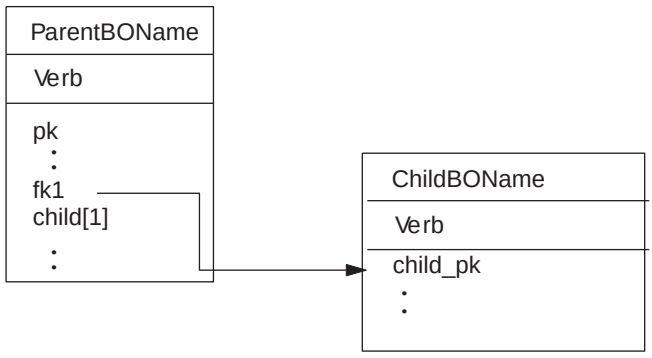


그림 2. 일반적인 단일 카디낼리티 관계

단일 카디낼리티 관계 및 소유권이 없는 데이터

일반적으로 각 상위 Business Object는 포함하는 하위 Business Object 내의 데이터를 **소유**합니다. 예를 들어 각 고객 Business Object가 단일 주소 Business Object를 포함하는 경우 새 고객이 작성될 때 새 행이 고객 및 주소 테이블 모두에 삽입됩니다.

새 주소는 새 고객에 대해 고유합니다. 마찬가지로 고객 테이블에서 고객을 삭제할 때 고객의 주소도 주소 테이블에서 삭제됩니다.

그러나 어느 계층 구조 Business Object도 소유하고 있지 않은 동일한 데이터를 복수 계층 구조 Business Object가 포함하는 경우가 있습니다. 예를 들어 주소 Business Object가 StateProvince lookup table with single cardinality를 표시하는 StateProvince[1] 속성을 갖고 있는 경우를 가정할 수 있습니다. 찾아보기 테이블이 거의 갱신되지 않고 주소 데이터와 독립적으로 유지보수되기 때문에 주소 데이터를 작성하거나 수정해도 찾아보기 테이블의 데이터에 영향을 주지 않습니다. 커넥터가 기존 도 또는 시 이름을 찾거나 실패합니다. 찾아보기 테이블에서 값을 추가하거나 변경하지 않습니다.

복수 Business Object가 동일한 단일 카디널리티 하위 Business Object를 포함할 때 각 상위 Business Object의 외부 키 속성은 관계를 NO_OWNERSHIP으로 지정해야 합니다. 통합 브로커가 작성, 삭제 또는 갱신 요청과 함께 계층 구조 Business Object를 커넥터에게 전송할 때 커넥터는 소유권 없이 포함된 단일 카디널리티 하위를 무시합니다. 커넥터는 이 Business Object에 대해 검색만을 수행합니다. 커넥터가 그러한 단일 카디널리티 Business Object의 검색에 실패하는 경우 오류를 리턴하고 처리를 중지합니다.

소유권 없는 관계를 지정하는 방법에 대한 정보는 71 페이지의 『단일 카디널리티 하위 Business Object를 나타내는 속성』을 참조하십시오. 외부 키 관계 지정에 대한 정보는 67 페이지의 『속성의 외부 키 지정』을 참조하십시오.

비표준화 데이터와 소유권이 없는 데이터

소유권이 없이 포함된 데이터는 정적 찾아보기 테이블의 사용을 쉽게 하는 것 외에 또 다른 성능인 표준화 및 비표준화 데이터의 동기화를 제공합니다.

표준화에서 비표준화 데이터로의 동기화: 관계를 NO_OWNERSHIP으로 지정하면 표준화 응용프로그램에서 비표준화 응용프로그램으로 동기화할 때 데이터를 작성하거나 변경할 수 있습니다. 예를 들어 표준화 소스 응용프로그램이 두 테이블 A 및 B에 데이터를 저장한다고 가정합니다. 또한 비표준화 목적지 응용프로그램이 모든 데이터를 단일 테이블에 저장해서 각 엔티티 A가 B 데이터를 이중으로 저장한다고 가정합니다.

이 예에서 테이블 B 데이터의 변경사항을 소스 응용프로그램에서 목적지 응용프로그램으로 동기화하려면 테이블 B 데이터가 변경될 때마다 테이블 A 이벤트를 트리거해야 합니다. 또한 테이블 B 데이터가 테이블 A에 이중으로 저장되기 때문에 테이블 B의 변경된 데이터를 포함하는 테이블 A의 각 행에 대한 Business Object를 전송해야 합니다.

비표준화에서 표준화 데이터로의 동기화: 비표준화 소스 응용프로그램에서 표준화 목적지 응용프로그램으로 데이터를 동기화할 때 커넥터는 표준화 응용프로그램에 소유권 없이 포함된 데이터를 작성, 삭제 또는 갱신하지 않습니다.

데이터를 표준화 응용프로그램으로 동기화할 때 커넥터는 소유권 없이 포함된 모든 단일 카디낼리티 하위를 무시합니다. 이런 하위 데이터를 작성, 제거 또는 수정하려면 데이터를 수동으로 처리하십시오.

다중 카디낼리티 관계

일반적으로 하위 Business Object의 배열을 포함하는 Business Object에는 관계를 표시하는 유일한 하나의 속성이 있습니다. 속성의 유형은 하위 Business Object와 동일한 유형의 배열입니다. 상위가 둘 이상의 하위를 포함하도록 하기 위해 관계를 설정하는 외부 키가 각 하위에 저장됩니다.

따라서 각 하위에는 상위의 1차 키를 외부 키로 포함하는 최소 하나의 단순 속성이 있습니다. 하위에는 상위에 있는 1차 키 속성 만큼의 외부 키 속성이 있습니다.

관계를 설정하는 외부 키가 하위에 저장되기 때문에 각 상위는 0개 이상의 하위를 가질 수 있습니다.

그림 3은 다중 카디낼리티 관계를 보여 줍니다. 이 예에서 `parentId`는 상위의 1차 키를 포함하는 단순 속성이고 `child[n]`은 하위 Business Object의 배열을 표시하는 속성입니다.

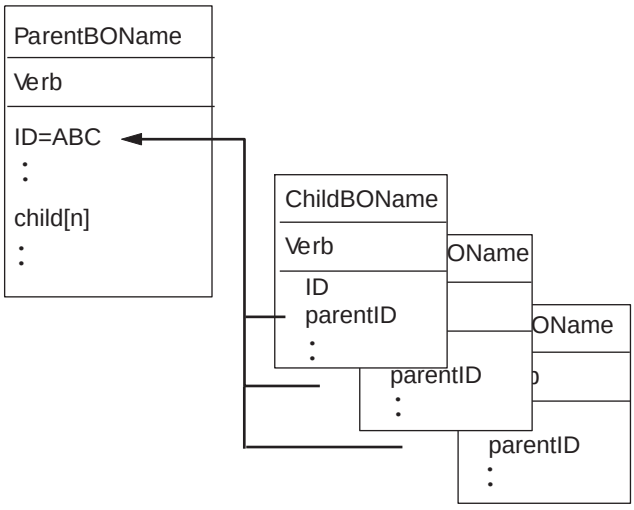


그림 3. 다중 카디낼리티 Business Object 관계

하위에 관계를 저장하는 단일 카디낼리티 관계

단일 하위 엔티티를 저장하고 있는 일부 응용프로그램의 경우 관계가 상위가 아니고 하위에 저장됩니다. 즉, 하위는 값이 상위의 1차 키에 저장된 값과 동일한 외부 키를 포함합니다.

그림 4는 이러한 특수 유형의 단일 카디널리티 관계를 보여 줍니다.

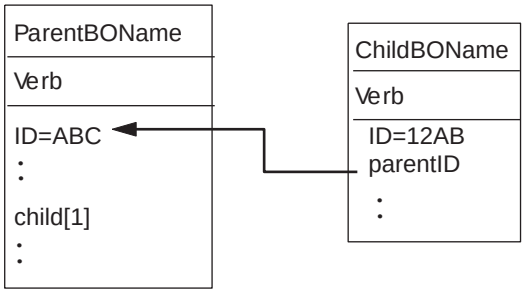


그림 4. 하위에 저장된 관계의 단일 카디널리티 Business Object

하위 데이터가 상위에 독립적으로 존재하지 않고 상위를 통해서만 이 하위에 액세스할 수 있을 때 응용프로그램이 이 유형의 단일 카디널리티 관계를 사용합니다. 그러한 하위 데이터는 둘 이상의 상위에 의해 소유되지 않으며 상위와 1차 키 값이 존재해야만 하위와 외부 키 값을 작성할 수 있습니다.

이러한 응용프로그램을 포함하기 위해 커넥터는 또한 단일 카디널리티의 하위를 포함하지만 상위가 아닌 하위에 관계를 저장하는 계층 구조 Business Object를 지원합니다.

상위 Business Object가 이러한 특수 방식으로 단일 카디널리티 하위를 포함할 수 있도록 지정하려면 하위를 포함하는 속성의 응용프로그램 특정 정보를 지정할 때 CONTAINMENT 매개변수를 포함하지 마십시오. 자세한 정보는 71 페이지의 『단일 카디널리티 하위 Business Object를 나타내는 속성』을 참조하십시오

랩퍼 오브젝트

랩퍼 오브젝트는 데이터베이스 테이블이나 보기로 상응하지 않는 최상위 레벨 Business Object입니다. 랩퍼 오브젝트는 true 값을 가진 WRAPPER의 최상위 레벨 Business Object 등록 정보에 의해 표시됩니다. 랩퍼 오브젝트는 관련 없는 Business Object에 대한 컨테이너로 사용되는 더미 상위입니다. 랩퍼 오브젝트를 처리할 때 커넥터가 최상위 레벨 Business Object를 무시하고 하위만 처리합니다. 랩퍼 오브젝트는 N 카디널리티나 N-1 카디널리티 엔티티 또는 둘 다 가질 수 있습니다.

N 카디널리티 엔티티는 1차 키로 표시되는 적어도 하나의 고유한 속성과 외부 키로 표시되는 적어도 하나의 속성을 가져야 합니다. 그런 후 이 외부 키가 랩퍼 오브젝트에서 1차 키로 추가됩니다. 엔티티의 외부 키는 방금 추가된 랩퍼 오브젝트의 1차 키를 참조합니다.

N-1 카디널리티 엔티티의 경우에는 1차 키가 1차 키와 외부 키 둘다로 표시되어야 하므로 랩퍼에서 N-1 엔티티에 있는 1차 키와 같은 1차 키를 참조합니다.

Business Object Verb 처리

이 섹션에서는 Business Object Verb 처리의 다음 측면에 대해 설명합니다.

- 『Verb 판별』: 각 개별, 소스 Business Object에 사용할 Verb를 커넥터가 판별하는 방법 설명
- 『사후 이미지 및 델타』: 용어를 정의하고 커넥터가 사후 이미지에 대해 작업하는 방법 설명
- 45 페이지의 『Verb 처리』: 커넥터가 Business Object를 작성, 검색, 갱신 또는 삭제할 때 수행하는 단계에 대해 설명
- 52 페이지의 『SQL 문』: 커넥터가 조작을 선택, 갱신, 검색 또는 삭제하기 위한 간단한 SQL 문을 사용하는 방법 설명
- 52 페이지의 『저장 프로시저』: 커넥터가 저장 프로시저를 사용하는 방법에 대해 설명
- 60 페이지의 『트랜잭션 확약 및 롤백』: 커넥터가 트랜잭션 블록을 사용하는 방법에 대해 설명

Verb 판별

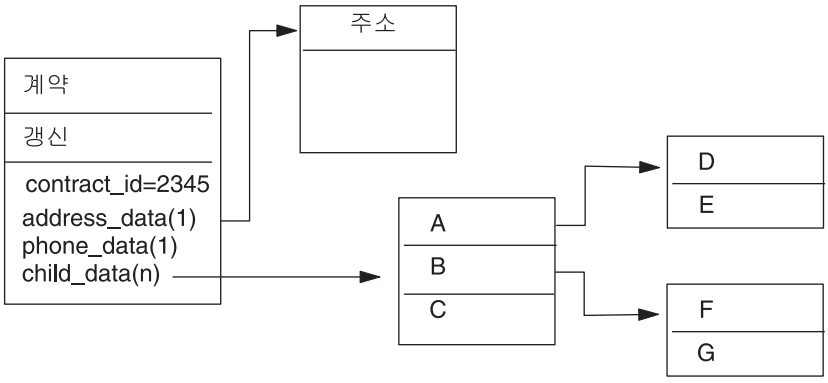
최상위 레벨 Business Object와 각 개별 하위 Business Object는 자체 Verb를 포함할 수 있습니다. 따라서 통합 브로커는 상위 및 하위 Business Object에 다른 Verb를 가지고 있는 Business Object를 커넥터에 전달할 수 있습니다. 이 경우에 커넥터는 최상위 레벨의 상위 Business Object Verb를 사용하여 전체 Business Object를 처리하는 방법을 판별합니다. 자세한 정보는 45 페이지의 『Verb 처리』를 참조하십시오.

사후 이미지 및 델타

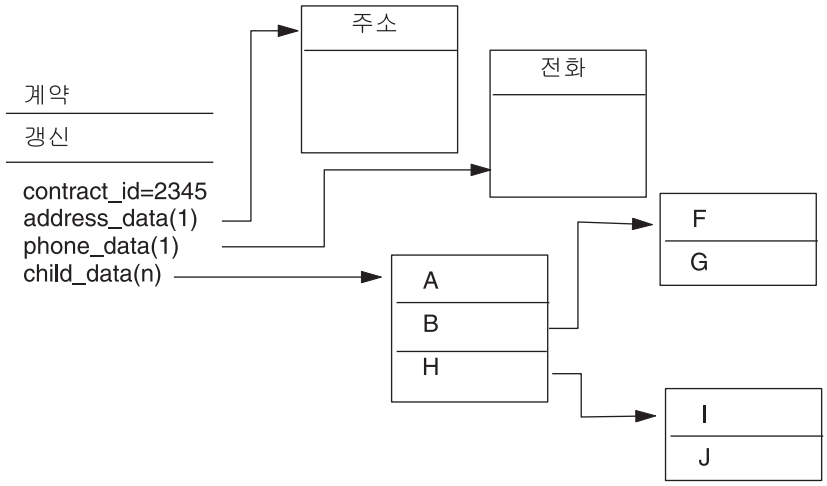
사후 이미지는 모든 변경을 수행한 다음의 Business Object 상태입니다. 델타는 변경될 데이터 및 키 값만을 포함하는 갱신 조작에서 사용되는 Business Object입니다. 커넥터가 사후 이미지만을 지원하기 때문에 갱신을 위한 Business Object를 수신할 때 커넥터는 Business Object가 갱신 후 데이터의 원하는 상태를 표시한다고 가정합니다.

따라서 통합 브로커가 Update Verb가 있는 Business Object를 커넥터에 전송할 때 커넥터는 데이터베이스에 있는 Business Object의 현재 표시를 소스 Business Object와 정확히 일치하도록 변경합니다. 이를 수행하기 위해 커넥터는 단순 속성 값을 변경하고 하위 Business Object를 추가하거나 제거합니다.

예를 들어 데이터베이스에서 Contract 2345의 현재 상태를 다음과 같이 가정합니다.



추가로 통합 브로커가 다음 Business Object를 커넥터에 전달한다고 가정합니다.



갱신을 처리하기 위해 커넥터는 다음 변경사항을 데이터베이스에 적용합니다.

- 최상위 레벨 담당자 및 주소 Business Object의 단순 속성 갱신
- 전화번호 Business Object 작성
- 하위 Business Object A, B, F 및 G의 단순 속성 갱신
- 하위 Business Object C, D 및 E 삭제
- 하위 Business Object H, I 및 J 작성

커넥터는 커넥터가 통합 브로커에서 수신하는 각 Business Object가 사후 이미지를 표시한다고 가정하기 때문에 갱신을 위해 해당 커넥터에 전송된 각 Business Object가 유효한 기존 하위 Business Object를 포함하는지 확인하는 것이 중요합니다. 변경된 하위 Business Object의 단순 속성이 없어도 소스 Business Object에 하위 Business Object를 포함시켜야 합니다.

그러나 일부 커넥터가 갱신 조작 중 누락된 하위 Business Object를 삭제하지 못하도록 차단할 수 있는 방법이 있습니다. 하위나 하위의 배열을 표시하는 속성에 대한 응용

프로그램 특정 정보를 사용하여 소스 Business Object에 포함되지 않은 하위 Business Object를 보존하도록 커넥터에게 지시할 수 있습니다. 이를 수행하려면 KEEP_RELATIONSHIP을 true로 설정하십시오. 자세한 정보는 67 페이지의 『속성의 외부 키 지정』을 참조하십시오.

Verb 처리

이 섹션에서는 커넥터가 통합 브로커에서 수신하는 Business Object를 작성, 검색, 갱신 또는 삭제할 때 수행하는 단계에 대해 설명합니다. 커넥터는 계층 구조 Business Object를 반복적으로 처리합니다. 즉, 모든 개별 Business Object를 처리할 때까지 각 하위 Business Object에 동일한 단계를 수행합니다.

주: 랩퍼인 최상위 레벨 Business Object는 Create, Retrieve, Update 및 Delete Verb를 지원합니다. 랩퍼 오브젝트를 처리할 때의 차이점은 랩퍼 오브젝트는 처리되지 않고 랩퍼 오브젝트가 포함하는 오브젝트만 처리된다는 것입니다.

Business Object 비교

커넥터는 아래에서 설명하는 처리 과정에서 Business Object가 동일한지 확인하기 위해 두 개의 Business Object를 비교합니다. 예를 들어 갱신 조작 중 커넥터는 특정 Business Object가 Business Object의 배열에 존재하는지 여부를 판별합니다. 검사를 수행하기 위해 커넥터는 Business Object를 배열 내의 각 Business Object와 비교합니다. 동일한 두 Business Object의 경우 다음 두 조건이 충족되어야 합니다.

- 비교되는 Business Object의 유형은 동일해야 합니다. 예를 들어 고객 Business Object와 접속 Business Object가 모든 속성이 같아도 동일한 것으로 간주되지 않습니다.
- 두 Business Object의 모든 해당 키 속성이 동일한 값을 포함해야 합니다. 두 Business Object에 키 속성을 CxIgnore로 설정하면 커넥터가 두 Business Object를 동일한 것으로 간주합니다. 그러나 한 Business Object의 키 속성을 CxIgnore로 설정하고 다른 Business Object에는 설정하지 않는 경우 Business Object는 일치하지 않습니다.

Create 조작

Business Object를 작성할 때 커넥터는 조작에 성공한 경우(조작이 Business Object에 대한 변경을 유발했는지 여부에 상관 없이) VALCHANGE 상태를 리턴하거나 조작에 실패한 경우 FAIL 상태를 리턴합니다.

커넥터는 계층 구조 Business Object를 작성할 때 다음 단계를 수행합니다.

1. 소유권을 갖고 포함되어 있는 각 단일 카디널리티 하위 Business Object를 데이터베이스에 반복 삽입합니다. 즉, 커넥터는 하위 및 해당 하위가 포함하는 모든 하위 Business Object 및 하위를 작성합니다.

속성이 단일 카디널리티의 하위 Business Object를 표시하고 속성이 비어 있다고 Business Object 정의가 지정하는 경우 커넥터는 속성을 무시합니다. 그러나 속성이 하위를 표시하도록 Business Object 정의가 요청하였으나 표시하지 않는 경우 커넥터는 오류를 리턴하고 처리를 중지합니다.

2. 소유권 없이 포함된 각 단일 카디널리티 하위 Business Object를 다음과 같이 처리합니다.
   a. 통합 브로커에 의해 전달된 키 값을 사용하여 데이터베이스에서 하위를 검색하려 반복적으로 시도합니다.
   b. 검색에 성공하지 못한 경우 현재 데이터베이스에 하위가 존재하지 않음을 의미하며 커넥터는 오류를 리턴하고 처리를 중지합니다. 검색에 성공하는 경우 커넥터는 하위 Business Object를 반복적으로 갱신합니다.

   주: 하위 Business Object가 응용프로그램 데이터베이스에 이미 존재할 때 이 접근 방법으로 올바르게 작업하려면 하위 Business Object의 1차 키 속성이 Create 조작 시에 올바르게 상호 참조되는지 확인하십시오. 하위 Business Object가 응용프로그램 데이터베이스에 아직 없는 경우 1차 키 속성을 `CxBlank`로 설정하십시오.
3. 최상위 레벨 Business Object를 다음과 같이 데이터베이스에 삽입합니다.
   a. 각각의 외부 키 값을 단일 카디널리티로 표시된 해당 하위 Business Object의 1차 키 값으로 설정합니다. 하위 Business Object의 값은 데이터베이스 순서나 카운터 또는 하위 작성 시 데이터베이스 스스로 설정할 수 있으므로 이 단계에서는 커넥터가 데이터베이스에 상위를 삽입하려면 상위의 외부 키 값이 올바른지 확인해야 합니다.
   b. 데이터베이스에서 자동 설정되는 각 속성의 새로운 고유 ID 값을 생성합니다. 데이터베이스 순서나 카운터의 이름은 속성의 응용프로그램 특정 정보에 저장됩니다. 속성에 연관된 데이터베이스 순서나 카운터가 있는 경우 커넥터에 의해 생성된 값이 통합 브로커가 전달한 모든 값을 겹쳐씁니다. 데이터베이스 순서 또는 카운터의 지정에 관한 자세한 정보는 65 페이지의 『단순 속성에 대한 응용프로그램 특정 정보』의 `UID=AUTO`를 참조하십시오.
   c. 속성 값을 속성 응용프로그램 특정 정보의 `CA`(CopyAttribute) 매개변수가 지정한 대로 다른 속성 값에 복사합니다. `CA` 매개변수의 사용에 관한 자세한 정보는 65 페이지의 『단순 속성에 대한 응용프로그램 특정 정보』의 `CA=set_attr_name`을 참조하십시오.
   d. 최상위 레벨 Business Object를 데이터베이스에 삽입합니다.

   주: 랩퍼인 최상위 레벨 Business Object는 데이터베이스로 삽입되지 않습니다.
4. 하위에 상위/하위 관계를 저장하는 단일 카디널리티 하위 Business Object 각각을 다음과 같이 처리합니다.

a. 상위의 해당 1차 키 속성 값을 참조하도록 하위의 외부 키 값을 설정합니다. 상위의 1차 키 값은 상위 작성 시 생성되었을 수 있으므로 커넥터가 데이터베이스에 하위를 삽입하기 전에 각 하위의 외부 키 값이 올바른지 확인해야 합니다.

b. 하위를 데이터베이스에 삽입합니다.

5. 다음과 같이 해당 다중 카디널리티 하위 Business Object를 각각 처리합니다.

a. 상위의 해당 1차 키 속성 값을 참조하도록 각 하위의 1차 키 값을 설정합니다. 상위의 1차 키 값은 상위 작성 시 생성되었을 수 있으므로 커넥터가 데이터베이스에 하위를 삽입하기 전에 각 하위의 외부 키 값이 올바른지 확인해야 합니다.

b. 해당 다중 카디널리티 하위 Business Object를 데이터베이스에 각각 삽입합니다.

Retrieve 조작

계층 구조 Business Object를 검색 시 커넥터는 다음 두 단계를 수행합니다.

1. 통합 브로커에서 수신한 최상위 레벨 Business Object에서 모든 하위 Business Object를 제거합니다.
2. 데이터베이스에서 최상위 레벨 Business Object를 검색합니다.
   - 검색 결과 한 행을 리턴하는 경우 커넥터는 처리를 계속합니다.
   - 검색 결과 행을 리턴하지 않는 경우 데이터베이스에 최상위 Business Object가 존재하지 않음을 의미하며 커넥터는 `BO_DOES_NOT_EXIST`를 리턴합니다.
   - 검색이 한 행 이상을 리턴하는 경우 커넥터가 `FAIL`을 리턴합니다.

   주: Business Object에는 위치 표시기 속성과 같이 데이터베이스 열에 일치하지 않는 속성이 있을 수 있습니다. 검색 중 커넥터는 최상위 레벨 Business Object에서 해당 속성을 변경하지 않습니다. 이는 여전히 통합 브로커에서 수신된 값으로 설정되어 있습니다. 하위 Business Object의 경우 커넥터는 검색 시 해당 속성을 해당 기본값으로 설정합니다.

   주: 랩퍼인 최상위 레벨 Business Object는 키와 위치 표시기 속성을 포함하여 오브젝트를 검색하는 데 필요한 랩퍼 오브젝트 바로 아래 레벨의 오브젝트 속성 값을 포함해야 합니다. 랩퍼 오브젝트는 모든 키와 위치 표시기 속성으로 채워져야 합니다. 랩퍼 오브젝트보다 한 레벨 아래에 있는 오브젝트에서 외부 키로 사용될 랩퍼 오브젝트의 간단한 속성이 랩퍼 오브젝트의 키로 표시되어야 합니다.

3. 모든 다중 카디널리티 하위 Business Object를 반복적으로 검색합니다.

   주: Business Object 배열을 채우는 경우 커넥터는 고유성을 강요하지 않습니다. 고유성을 확인하는 것은 데이터베이스의 책임입니다. 데이터베이스가 중복 하위 Business Object를 리턴하는 경우 커넥터가 중복 하위를 리턴합니다.

4. 소유권 보유 여부에 따른 하위 Business Object 포함 여부에 관계 없이 각 단일 카디널리티 하위를 반복적으로 검색합니다.

   주: 모든 단일 카디널리티 하위 Business Object는 Business Object에서 발생 시 및 상위 Business Object가 처리되기 전에 처리됩니다. 하위 오브젝트의 소유권 보유 여부는 처리 순서를 판별하지 않으나 처리 유형은 판별합니다.

RetrieveByContent 조작

커넥터는 최상위 레벨 Business Object의 속성만을 기반으로 하는 검색을 수행하므로 RetrieveByContent Verb는 최상위 레벨 Business Object에만 적용됩니다.

최상위 레벨 Business Object가 RetrieveByContent Verb를 사용하는 경우 널값이 아닌 모든 속성(키가 아닌 속성 포함)이 검색 기준으로 사용됩니다.

두 행 이상이 리턴되는 경우 커넥터는 결과 행으로 첫 번째 행을 사용하고 `MULTIPLE_HITS`를 리턴합니다.

주: RetrieveByContent Verb는 랩퍼인 최상위 레벨 Business Object에 적용할 수 없습니다.

Update 조작

Business Object를 갱신할 때 커넥터는 조작에 성공한 경우(조작으로 Business Object에 대한 변경이 발생했는지 여부에 상관 없이) `VALCHANGE` 상태를 리턴하거나 조작에 실패한 경우에는 `FAIL` 상태를 리턴합니다. Oracle 데이터베이스에 대한 작업을 수행하는 경우 커넥터는 검색 시 데이터를 잠금으로써 데이터 무결성을 확인합니다.

커넥터는 계층 구조 Business Object를 갱신할 때 다음 단계를 수행합니다.

1. 소스 Business Object의 1차 키 값을 사용하여 데이터베이스에서 해당 엔티티를 검색합니다. 검색된 Business Object는 데이터베이스에 있는 현재 데이터 상태를 정확히 표시한 것입니다.
   - 검색에 실패하는 경우 데이터베이스에 최상위 레벨 Business Object가 존재하지 않음을 의미하며 커넥터는 `BO_DOES_NOT_EXIST`를 리턴하고 갱신에 실패합니다.

     주: 랩퍼인 최상위 레벨 Business Object는 데이터베이스에 존재할 필요가 없습니다. 그러나 이는 랩퍼 오브젝트 바로 아래 레벨에 있는 오브젝트의 모든 속성 값을 포함해야 하며 이는 키와 위치 표시기 속성을 포함하여 오브젝트를 검색하는 데 필요합니다. 랩퍼 오브젝트는 모든 키와 위치 표시기 속성으로 채워져야 합니다. 랩퍼 오브젝트보다 한 레벨 아래에 있는 오브젝트에서 외부 키로 사용될 랩퍼 오브젝트의 간단한 속성이 랩퍼 오브젝트의 키로 표시되어야 합니다.
   - 검색에 성공하는 경우 커넥터는 검색된 Business Object를 소스 Business Object에 비교하여 데이터베이스에서 변경되어야 하는 하위 Business Object를 판별합

니다. 그러나 커넥터는 소스 Business Object의 단순 속성 값을 검색된 Business Object의 단순 속성 값과 비교하지 않습니다. 커넥터는 키가 아닌 모든 단순 속성의 값을 갱신합니다.

최상위 레벨 Business Object의 모든 단순 속성이 키를 표시하는 경우 커넥터가 최상위 레벨 Business Object에 대한 갱신 조회를 생성할 수 없습니다. 이 경우 커넥터가 경고를 기록하고 2단계로 계속 진행합니다.

2. 최상위 레벨 Business Object의 모든 단일 카디널리티 하위를 반복적으로 갱신합니다.

   Business Object 정의에서 속성이 하위 Business Object를 나타내야 하는 경우 소스 Business Object 및 검색된 Business Object 모두에 하위가 존재해야 합니다. 그렇지 않은 경우 갱신에 실패하고 커넥터가 오류를 리턴합니다.

   커넥터는 다음 중 한 가지 방법으로 소유권을 보유하고 포함되어 있는 단일 카디널리티 하위를 처리합니다.

   - 하위가 소스와 검색된 Business Object에 모두 있는 경우 커넥터는 데이터베이스에 이미 존재하는 하위를 갱신하는 대신에 기존 하위를 삭제하고 새 하위를 작성합니다.
   - 하위가 소스 Business Object에 있지만 검색된 Business Object에는 없는 경우 커넥터는 데이터베이스에서 이를 반복적으로 작성합니다.
   - 하위가 검색된 Business Object에 있지만 소스 Business Object에는 없는 경우 커넥터는 데이터베이스에서 이를 반복적으로 삭제합니다. 삭제, 실제 또는 논리 유형은 해당 `ChildUpdatePhyDelete` 등록 정보 값에 따라 다릅니다.

   소유권 없이 포함되어 있는 단일 카디널리티 하위의 경우 커넥터는 소스 Business Object에 존재하는 데이터베이스의 모든 하위에 대한 검색을 시도합니다. 커넥터가 하위 검색에 성공하는 경우 커넥터는 하위 Business Object를 채우거나 갱신하지 않습니다. 소유권 없이 포함되어 있는 단일 카디널리티는 커넥터가 수정할 수 없기 때문입니다.

3. 상위에 관계를 저장하는 단일 카디널리티 하위 Business Object의 경우 상위의 각 외부 키 값을 해당 단일 카디널리티 하위 Business Object의 1차 키의 값으로 설정합니다. 이 단계는 단일 카디널리티 하위가 이전 단계에서 데이터베이스에 추가되어 새 고유 ID를 생성할 수 있다는 점에서 필요합니다.

4. 소스 Business Object의 해당 속성이 `CxIgnore` 값을 포함하는 속성을 제외하고 검색된 Business Object의 모든 단순 속성을 갱신합니다.

   갱신되는 Business Object는 고유해야 하므로 커넥터는 결과적으로 한 행만이 처리되는지 확인합니다. 두 행 이상이 리턴되는 경우 오류를 리턴합니다.

5. 상위/하위 관계를 하위에 저장하는(다중 카디널리티와 단일 카디널리티 모두) 각 하위의 모든 외부 키 값을 해당 상위 Business Object의 1차 키 값으로 설정합니다. (ICS가 통합 브로커로 사용될 때 이 값은 일반적으로 데이터 맵핑 중 상호 참조됨

니다.) 그러나 이 단계에서는 커넥터가 해당 하위를 갱신하려면 하위에 관계를 저장하는 새 하위의 외부 키 값이 올바른 값이어야 한다는 것이 중요합니다.

6. 검색된 Business Object의 각 다중 카디널리티 하위를 다음 중 한 가지 방식으로 처리합니다.
   - 하위가 소스 및 검색된 Business Object의 배열 모두에 존재하는 경우 커넥터는 데이터베이스에서 이를 반복적으로 갱신합니다.
   - 하위가 소스 배열에 존재하지만 검색된 Business Object의 배열에는 없는 경우 커넥터는 데이터베이스에서 이를 반복적으로 작성합니다.
   - 하위가 검색된 Business Object의 배열에 존재하지만 소스 배열에는 없는 경우 상위에서 하위를 표시하는 속성에 대한 응용프로그램 특정 정보에 `true`로 설정된 `KEEP_RELATIONSHIP`이 없으면 커넥터가 데이터베이스에서 이를 반복적으로 삭제합니다. 이 경우 커넥터는 데이터베이스에서 하위를 삭제하지 않습니다. 자세한 정보는 67 페이지의 『속성의 외부 키 지정』을 참조하십시오. 삭제, 실제 또는 논리 유형은 관련 `ChildUpdatePhyDelete` 등록 정보의 값에 따라 다릅니다.

주: 통합 브로커는 소스 Business Object에 다중 카디널리티와 함께 포함된 Business Object가 고유한지(즉, 배열에 동일한 둘 이상의 Business Object 사본이 있지 않은지) 확인해야 합니다. 커넥터가 소스 배열에서 Business Object의 중복을 수신하는 경우 Business Object를 두 번 처리하는데, 예측할 수 없는 결과가 나타날 수도 있습니다.

DeltaUpdate 조작

DeltaUpdate Verb 처리는 다음과 같이 Update Verb 처리와 다릅니다.

1. DeltaUpdate의 경우 갱신 전에 검색이 이루어지지 않는데 반해 Update Verb 처리에서는 검색이 이루어집니다.
2. 수신 Business Object와 데이터베이스의 Business Object 사이의 비교가 이루어지지 않습니다.
3. 각 하위 오브젝트의 Verb 세트에 따라 모든 하위가 처리됩니다. 하위에 Verb 세트가 없는 경우 커넥터가 오류를 리턴합니다.

Business Object를 델타 갱신하는 경우 커넥터는 조작이 성공하면(조작으로 Business Object가 변경되었는지 여부에 관계 없이) `VALCHANGE` 상태를 리턴하거나 조작이 실패하면 `FAIL` 상태를 리턴합니다.

커넥터는 계층 구조 Business Object를 델타 갱신할 때 다음 단계를 수행합니다.

1. 상위 오브젝트의 모든 단일 카디널리티 하위를 반복적으로 처리합니다. 하위가 Business Object 스펙에서 IsRequired로 표시되면 인바운드 오브젝트에 존재해야 합니다. 그렇지 않은 경우 델타 갱신이 실패하며 커넥터가 오류를 리턴합니다.

2. 단일 카디널리티 하위의 속성을 참조하는 상위의 모든 외부 키 값을 해당 하위 값으로 설정합니다. 이 단계는 단일 카디널리티 하위가 이전 단계에서 데이터베이스에 추가되어 새 순차 값을 생성할 수 있다는 점에서 필요합니다.
3. SQL UPDATE 문 또는 저장 프로시저를 통해 처리 중인 현재 오브젝트를 갱신합니다. 인바운드 Business Object에 IsIgnore로 설정된 속성을 제외한 개별 Business Object의 모든 단순 속성이 갱신됩니다. 커넥터는 갱신 명령문에 추가해야 하는 속성을 판별하기 위해 속성 레벨에서 인바운드 오브젝트와 현재 오브젝트를 비교하지 않습니다. 오브젝트 모두가 갱신됩니다. 갱신되는 Business Object는 고유해야 하므로 커넥터는 결과적으로 한 행만이 처리되는지 확인합니다. 두 행 이상이 처리되는 경우 오류를 리턴합니다.
4. 상위 속성을 참조하는 현재 오브젝트의 모든 카디널리티 N 하위에 있는 모든 외부 키 값을 해당 상위 값으로 설정합니다. 일반적으로 데이터 맵핑 시 이 값이 이미 상호 참조됩니다. 그러나 카디널리티 N 컨테이너의 새 하위에는 적용되지 않습니다. 이로써 모든 카디널리티 N 하위가 갱신되기 전에 하위의 외부 키 값이 올바르게 됩니다.
5. 현재 오브젝트의 모든 카디널리티 N 컨테이너를 갱신합니다.

하위 오브젝트를 처리할 때 각 하위의 Verb를 선택하여 적절한 조작을 완료합니다. DeltaUpdate의 하위에서 허용되는 Verb는 Create, Delete 및 DeltaUpdate입니다.

- Create Verb가 하위에 있으며 하위가 소유권 하위인 경우 하위가 데이터베이스에 작성됩니다. 데이터베이스에 존재하는지 확인하기 위해 비소유권 하위를 검색합니다.
- Delete Verb가 하위에 있으면 해당 하위가 삭제됩니다.
- 하위에서 DeltaUpdate Verb를 찾은 경우 데이터베이스에서 해당 하위가 갱신됩니다.

Delete 조작

Business Object를 삭제할 때 커넥터는 조작에 성공한 경우 SUCCESS 상태를 리턴하거나 조작에 실패한 경우 FAIL 상태를 리턴합니다. 상위 Business Object를 먼저 검색하며, 그리고 나서 어댑터가 상위에 대해 소유권 관계가 있는 모든 단일 카디널리티 하위를, 다음으로 상위 Business Object를, 마지막으로 모든 카디널리티 N 하위를 반복적으로 삭제합니다. 단일 카디널리티 비소유권 하위는 삭제되지 않습니다. Business Object가 존재하지 않는 경우 커넥터는 FAIL을 리턴합니다.

커넥터는 오브젝트의 응용프로그램 특정 정보에 있는 SCN(Status Column Name) 값에 따라 논리 및 실제 삭제를 지원합니다. SCN 값이 정의되어 있는 경우 커넥터는 논리 삭제를 수행합니다. SCN 값이 정의되어 있지 않은 경우 커넥터는 실제 삭제를 수행합니다.

실제 삭제: 커넥터는 계층 구조 Business Object를 실제로 삭제할 때 다음 단계를 수행합니다.

1. 소유권을 포함한 모든 단일 카디널리티 하위 Business Object를 반복적으로 삭제합니다.
2. 최상위 레벨 Business Object를 삭제합니다.
3. 모든 다중 카디널리티 하위 Business Object를 반복적으로 삭제합니다.

주: 랩퍼인 최상위 레벨 Business Object는 해당 데이터베이스 테이블을 갖지 않으므로 데이터베이스에서 삭제되지 않습니다. 랩퍼의 단순한 속성 값은 무시됩니다.

논리 삭제: Business Object를 논리적으로 삭제할 때 커넥터는 다음 단계를 수행합니다.

1. Business Object의 상태 속성을 Business Object의 응용프로그램 특정 정보에서 지정하는 값으로 설정하는 UPDATE를 발행합니다. 커넥터는 결과적으로 데이터베이스 행이 하나만 갱신되는지 확인하고 그렇지 않은 경우 오류를 리턴합니다.
2. 소유권과 함께 포함된 모든 단일 카디널리티 하위와 모든 다중 카디널리티 하위를 논리적으로 반복해서 삭제합니다. 커넥터는 소유권 없이 포함된 단일 카디널리티 하위를 삭제하지 않습니다.

SQL 문

커넥터는 선택, 갱신, 검색 또는 삭제 조작에 단순 SQL 문을 사용할 수 있습니다. SQL 문의 열 이름은 속성의 AppSpecificInfo 등록 정보에서 파생됩니다. 각 조회는 보기에 게시되지 않는 경우 테이블을 하나만 확장합니다.

저장 프로시저

저장 프로시저는 논리 단위를 형성하고 특정 타스크를 수행하는 SQL 문의 그룹입니다. 저장 프로시저는 조작의 세트를 캡슐화하거나 커넥터를 조회하여 데이터베이스 서버에서 오브젝트를 실행합니다.

커넥터가 저장 프로시저를 호출하는 경우는 다음과 같습니다.

- Business Object를 처리하기 전에 준비 조작 프로세스를 수행하는 경우
- Business Object를 처리한 후 사후 조작 프로세스를 수행하는 경우
- 단순 INSERT, RETRIEVE, UPDATE 또는 DELETE 문을 사용하는 대신 Business Object에 조작 세트를 수행하는 경우

커넥터가 계층 구조 Business Object를 처리하는 경우 저장 프로시저를 사용하여 최상위 레벨 Business Object 또는 해당 하위 Business Object를 처리할 수 있습니다. 그러나 각 Business Object 또는 Business Object 배열에는 자체 저장 프로시저가 있어야 합니다.

저장 프로시저 지정

이 섹션에서는 커넥터가 Business Object에 저장 프로시저를 사용하도록 하기 위해 수행해야 하는 단계에 대해 설명합니다. 다음 섹션을 포함합니다.

- 『Business Object에 속성 추가』
- 『저장 프로시저 구문』
- 54 페이지의 『저장 프로시저 예』
- 55 페이지의 『저장 프로시저 지정』

Business Object에 속성 추가: 커넥터가 처리하는 각 저장 프로시저 유형의 Business Object에 특별한 종류의 속성을 추가해야 합니다. 이 속성은 저장 프로시저의 유형 및 속성을 정의하는 응용프로그램 특정 정보를 표시합니다. 이 속성은 표준 단순 속성에 대해 사용 가능한 응용프로그램 특정 정보 매개변수를 사용하지 않습니다.

사용될 저장 프로시저 유형에 따라 속성 이름을 지정하십시오. 예를 들어 커넥터가 AfterUpdate 및 BeforeRetrieve 저장 프로시저를 사용하도록 하려면 `AfterUpdateSP` 및 `BeforeRetrieveSP` 속성을 추가하십시오.

커넥터는 다음 Business Object 속성 이름을 인식합니다.

```
BeforeCreateSP
AfterCreateSP
CreateSP
BeforeUpdateSP
AfterUpdateSP
UpdateSP
BeforeDeleteSP
AfterDeleteSP
DeleteSP
BeforeRetrieveSP
AfterRetrieveSP
RetrieveSP
BeforeRetrieveByContentSP
AfterRetrieveByContentSP
RetrieveByContentSP
BeforeRetrieveUpdateSP
AfterRetrieveUpdateSP
RetrieveUpdateSP
```

주: 커넥터가 실행하도록 하려는 해당 저장 프로시저에 대해서만 속성을 작성하십시오. 응용프로그램 특정 정보나 맵핑을(ICS가 통합 브로커로 사용되는 경우에만) 사용하여 Business Object가 커넥터에 전송되기 전에 이 속성에 값을 지정하십시오. 커넥터가 Business Object에서 후속 호출에 대한 값의 변경사항을 인식하려면 커넥터를 다시 시작해야 합니다.

저장 프로시저 구문: 저장 프로시저를 지정하는 구문은 다음과 같습니다.

```
SPN=StoredProcedureName;RS=true|false[;IP=Attribute_Name1[:Attribute_Name2[:...]]]
[;OP=Attribute_Name1| RS[:Attribute_Name2| RS[:...]]]
[;IO=Attribute_Name1[:Attribute_Name2[:...]]]
```

여기서,

StoredProcedureName 저장 프로시저의 이름입니다.

RS 저장 프로시저가 결과 세트를 리턴하는 경우에는 true이고 그렇지 않은 경우에는 false입니다. 기본값은 false입니다. 값이 true인 경우 속성의 응용프로그램 특정 정보에 있는 ColumnName 등록 정보는 결과 세트에서 해당 열을 나타냅니다. RS가 출력 매개변수 목록의 일부일 경우 특정 매개변수는 결과 세트를 리턴합니다. 하나의 결과 세트인 OUT 매개변수만 지원됩니다. OUT 매개변수로 여러 개의 결과 세트가 리턴되는 경우 첫 번째 결과 세트만 리턴되고 다른 모든 결과 세트는 무시합니다. 현재 이 기능은 Oracle 8i 이상에서 Oracle JDBC Driver를 사용하는 저장 프로시저에 대해 지원됩니다. 데이터베이스에 있는 저장 프로시저의 경우 해당 매개변수는 REFCURSOR 유형을 리턴해야 합니다.

IP 입력 매개변수: 커넥터가 저장 프로시저를 실행할 때 입력 값으로 속성 값을 사용해야 하는 Business Object 속성의 목록.

OP 출력 매개변수: 커넥터가 저장 프로시저를 실행한 후 속성 값을 리턴해야 하는 Business Object 속성의 목록. 결과 세트 설명에 대해서는 RS를 참조하십시오.

IO InputOutput 매개변수: 커넥터가 입력 값으로 속성 값을 사용해야 하고 저장 프로시저를 실행한 후 속성 값을 리턴해야 하는 Business Object 속성의 목록.

주: StoredProcedureName, RS 및 매개변수의 순서가 중요합니다. 하지만 매개변수 간의 순서는 중요하지 않습니다. 즉, 저장 프로시저가 각 유형의 모든 매개변수를 그룹화하거나 매개변수의 유형을 산재시키는 경우 이는 커넥터에 아무런 차이가 없습니다.

동일한 유형의 복수 매개변수를 함께 그룹화하는 경우 콜론 분리문자로 값을 분리하십시오. 각 값에 매개변수의 이름을 반복할 필요는 없습니다. 세미콜론 분리문자로 다른 유형의 매개변수를 분리하십시오. 매개변수 값을 지정하는 경우 등호(=) 양쪽에 모두 공백을 두지 마십시오.

저장 프로시저 예: 다음 예에서는 두 개의 입력 속성에서 값을 가져와 네 개의 출력 속성에 값을 리턴하는 CustomerInsert 및 VendorInsert 저장 프로시저를 사용합니다. 예에서는 저장 프로시저의 다른 구조에 대해 설명합니다.

- 동일한 유형의 매개변수가 함께 그룹화되는 경우(IP, IP, OP, OP, OP, IO):

  ```
  SPN=CustomerInsert;RS=false;IP=LastName:FirstName;OP=CustomerName:
  CustomerID:ErrorStatus:ErrorMessage;IO=VendorID
  ```

- 동일한 유형의 매개변수가 나열되는 경우(IP, OP, OP, OP, IP, IO, OP):

  ```
  SPN=VendorInsert;RS=false;IP=LastName;OP=CustomerName:
  CustomerID:ErrorStatus;IP=FirstName;IO=VendorID;OP=ErrorMessage
  ```

커넥터는 JDBC 드라이버에서 지원하는 단순 데이터 유형만을 지원합니다.

저장 프로시저 지정: 저장 프로시저 이름 및 해당 매개변수 값을 지정하는 두 가지 방법은 다음과 같습니다.

- 속성의 AppSpecificInfo 등록 정보

  저장 프로시저를 지정하는 텍스트의 길이가 4KB 이하인 경우 속성의 AppSpecificInfo 등록 정보에 값을 지정할 수 있습니다. 이 등록 정보를 사용하여 커넥터가 Business Object를 폴했거나(즉, Business Object가 응용프로그램 이벤트를 표시함) Business Object를 통합 브로커 요청으로 수신했는지 여부와 관계 없이 저장 프로시저를 지정할 수 있습니다.

  다음 예는 응용프로그램 특정 정보에서 저장 프로시저의 스펙을 보여 줍니다. 이 경우 MaxLength 등록 정보에 지정된 값은 저장 프로시저에 중요하지 않습니다.

  ```
     [Attribute]
     Name = BeforeCreateSP
     Type = String
     MaxLength = 15
     IsKey = false
     IsRequired = false
     AppSpecificInfo =SPN=ContactInsert;IP=LastName:FirstName;OP=CustomerName:
  CustomerID:ErrorStatus:ErrorMessage

  [End]
  ```

- 속성 값(ICS가 통합 브로커로 사용된 경우에만 해당함)

  저장 프로시저를 지정하는 텍스트 길이가 4KB를 초과하는 경우 맵핑을 사용하여 저장 프로시저를 지정해야 합니다. Business Object가 통합 브로커 요청을 표시하는 경우에만 맵핑을 사용하여 저장 프로시저를 지정할 수 있습니다. 즉, 커넥터가 이벤트를 폴링하는 경우 속성 값을 사용하여 저장 프로시저를 지정할 수 없습니다.

  저장 프로시저의 텍스트가 4KB보다 길고 맵핑을 사용하여 지정하는 경우 MaxLength 등록 정보 값을 확장해 전체 텍스트를 수용할 수 있도록 하십시오.

주: 작성, 갱신 또는 삭제 조작을 처리하는 저장 프로시저가 하위 Business Object 배열을 포함하는 계층 구조 Business Object에서 실행되는 경우 커넥터는 각 하위 Business Object를 개별 처리합니다. 예를 들어 커넥터가 BeforeCreate 저장 프로시저를 실행하는 경우 배열을 단위로 처리하지 않고 배열 내 각 구성원을 처리합니다. BeforeRetrieve 저장 프로시저를 처리할 때 커넥터는 단일 Business Object

에서 조작합니다. AfterRetrieve 저장 프로시저를 처리하는 경우 커넥터는 검색을 통해 리턴된 모든 Business Object에서 작동합니다.

저장 프로시저 또는 단순 SQL 문을 사용한 Business Object 처리

다음 섹션에서는 커넥터가 저장 프로시저를 처리하는 방법에 대해 설명합니다.

- 『Business Object Create 조작』
- 57 페이지의 『Business Object Update 조작』
- 57 페이지의 『Business Object Delete 조작』
- 58 페이지의 『Business Object Retrieve 조작』
- 59 페이지의 『Business Object RetrieveByContent 조작』
- 59 페이지의 『Business Object Retrieve-for-Update 조작』

Business Object Create 조작: Create 저장 프로시저는 일반적으로 커넥터가 최상위 레벨 Business Object에서 단순 속성을 입력하기 위해 사용하는 값을 리턴합니다. 커넥터는 Create 저장 프로시저(BeforeCreate, Create, AfterCreate)를 처리할 때 다음 단계를 수행합니다.

1. Business Object가 BeforeCreateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 BeforeCreate 저장 프로시저를 호출합니다.
2. 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.
3. 단일 카디널리티 하위 Business Object를 작성합니다.
4. 각 최상위 레벨 Business Object의 외부 키 값을 각 단일 카디널리티 하위 Business Object의 1차 키 값으로 설정합니다.
5. Business Object가 CreateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 Create 저장 프로시저를 호출하여 최상위 레벨 Business Object를 작성합니다. 포함하지 않는 경우 INSERT 문을 빌드 및 실행하여 최상위 레벨 Business Object를 작성합니다.
6. Create 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.
7. 각 다중 카디널리티 하위의 외부 키 값을 해당 상위의 1차 키 속성의 값으로 설정합니다.
8. 다중 카디널리티 하위 Business Object를 작성합니다.
9. Business Object가 AfterCreateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 AfterCreate 저장 프로시저를 호출합니다.
10. 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.

커넥터는 10단계에서 리턴된 값을 사용하여 3 또는 5단계에서 작성한 Business Object의 값을 변경할 수 있습니다.

Business Object Update 조작: Update 저장 프로시저는 일반적으로 커넥터가 최상위 레벨 Business Object에 단순 속성을 채우는 데 사용하는 값을 리턴합니다. 커넥터는 Update 저장 프로시저(BeforeUpdate, Update, AfterUpdate)를 처리할 때 다음 단계를 수행합니다.

1. Business Object가 BeforeUpdateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 BeforeUpdate 저장 프로시저를 호출합니다.
2. BeforeUpdate 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.
3. 단일 카디널리티 하위 Business Object를 갱신합니다.
4. 각 최상위 레벨 Business Object의 외부 키 값을 단일 카디널리티로 포함된 각 하위 Business Object의 1차 키 값으로 설정합니다.
5. Business Object가 UpdateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 Update 저장 프로시저를 호출하여 최상위 레벨 Business Object를 갱신합니다. 포함하지 않는 경우 UPDATE 문을 빌드 및 실행하여 최상위 레벨 Business Object를 갱신합니다.
6. Update 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.
7. 다중 카디널리티 하위의 외부 키 값이 상위의 해당 1차 키 속성에 있는 값을 참조하도록 설정합니다.
8. 다중 카디널리티 하위 Business Object를 갱신합니다.
9. Business Object가 AfterUpdateSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 AfterUpdate 저장 프로시저를 호출합니다.
10. 저장 프로시저가 출력 매개변수를 통해 값을 리턴하는 경우 해당 값을 사용하여 Business Object에 단순 속성 값을 설정합니다.

Business Object Delete 조작: Delete 저장 프로시저는 커넥터에 값을 리턴하지 않습니다. 커넥터는 Delete 저장 프로시저(BeforeDelete, Delete, AfterDelete)를 처리할 때 다음 단계를 수행합니다.

1. Business Object가 BeforeDeleteSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 BeforeUpdate 저장 프로시저를 호출합니다.
2. 단일 카디널리티 하위 Business Object를 삭제합니다.
3. 다중 카디널리티 하위 Business Object를 삭제합니다.
4. Business Object가 DeleteSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 Delete 저장 프로시저를 호출하여 최상위 Business Object를 삭제합니다. 포함하지 않는 경우 DELETE 문을 빌드 및 실행합니다.

5. Business Object가 AfterDeleteSP 속성을 포함하는지 여부를 확인합니다. 포함하는 경우 AfterDelete 저장 프로시저를 호출합니다.

Business Object Retrieve 조작: 단순 RETRIEVE 조작의 경우 다중 카디널리티 하위뿐만 아니라 최상위 레벨 Business Object와 단일 카디널리티 하위에 저장 프로시저를 사용할 수 있습니다. 프로시저의 순서는 다음과 같습니다.

- BeforeRetrieve
- Retrieve
- AfterRetrieve

커넥터는 임시 오브젝트를 작성하여 단일 카디널리티 하위 Business Object 또는 다중 카디널리티 하위 Business Object를 검색합니다. 커넥터는 BeforeRetrieve 저장 프로시저를 임시 Business Object에 적용합니다. AfterRetrieve 저장 프로시저는 컨테이너로 검색된 각 하위 오브젝트에 적용됩니다.

커넥터는 Business Object 메타 데이터나 Business Object의 저장 프로시저에서 동적으로 생성된 Retrieve 조회를 실행한 후 AfterRetrieve 저장 프로시저를 실행합니다.

JDBC 스펙에 따른 세 가지 StoredProcedure 호출 유형은 다음과 같습니다.

- {call <spName>(?,?,?)}
- {call <spName>}
- {?= call <spName>(?,?,?)}

커넥터는 처음 두 가지 유형을 지원합니다. StoredProcedure에서 리턴되는 ResultSet를 처리합니다.

저장 프로시저 구문에서 RS=true인 경우 저장 프로시저의 결과 세트가 처리됩니다. RS=false인 경우 결과 세트는 처리되지 않습니다. 기본적으로 RS 값은 false입니다. 결과 세트 값을 처리한 후 저장 프로시저 출력 변수가 처리됩니다. RS=true인 경우 다중 카디널리티 하위는 관련 저장 프로시저에서 출력 변수를 지정할 수 없습니다.

주: Retrieve Verb 조작 및 RetrieveSP의 결과 세트 처리만이 지원됩니다.

RetrieveSP(Retrieve Stored Procedure)에서 리턴된 결과 세트 처리: ResultSetMetaData는 저장 프로시저에서 리턴된 결과 세트에 대해 확보됩니다. 결과 세트에 있는 모든 열의 값은 Business Object의 해당 속성에서 확보되고 설정됩니다. 속성의 응용프로그램 특정 정보의 ColumnName 등록 정보는 속성을 열에 일치시키는 ResultSet 열 이름을 포함해야 합니다.

단일 카디널리티 오브젝트의 경우 해당 결과 세트는 한 행으로만 구성되어야 합니다. 결과 세트에 복수 행이 리턴되면 오류가 보고됩니다.

다중 카디널리티 하위의 경우 결과 세트를 통해 복수 행이 리턴될 수 있습니다. 리턴된 각 행의 경우 새 오브젝트가 작성되어 컨테이너에 추가됩니다. 그런 다음 필수 속성 색인에서 컨테이너가 상위 오브젝트에 추가됩니다.

Business Object RetrieveByContent 조작: 단순 RetrieveByContent 조작의 경우 최상위 레벨 Business Object와 해당 단일 카디널리티 하위에만 저장 프로시저를 사용할 수 있습니다. 즉, 결과 세트나 복수 행을 리턴하는 데 저장 프로시저를 사용할 수 없습니다. 프로시저의 순서는 다음과 같습니다.

- BeforeRetrieveByContent
- RetrieveByContent
- AfterRetrieveByContent

커넥터는 임시 오브젝트를 작성하여 단일 카디널리티 하위 Business Object 또는 다중 카디널리티 하위 Business Object를 검색합니다. 다중 카디널리티 Business Object의 경우 커넥터는 BeforeRetrieveByContent 저장 프로시저를 임시 Business Object에 적용합니다. AfterRetrieveByContent 저장 프로시저는 컨테이너로 검색된 각 하위 오브젝트에 적용됩니다.

커넥터는 Business Object 메타 데이터나 Business Object의 저장 프로시저에서 동적으로 생성된 RetrieveByContent 조회를 실행한 후 AfterRetrieveByContent 저장 프로시저를 실행합니다. 이 경우 계층 구조 Business Object의 검색이 해당 하위 Business Object를 검색해도 커넥터는 배열에 있는 모든 Business Object에서 AfterRetrieveByContent 저장 프로시저를 실행합니다.

Business Object Retrieve-for-Update 조작: 다음 저장 프로시저는 최상위 Business Object에서 호출되며 단순 Retrieve와 동일한 방식으로 모든 하위 Business Object를 검색합니다.

프로시저의 순서는 다음과 같습니다.

- BeforeRetrieveUpdate
- RetrieveUpdate
- AfterRetrieveUpdate

이 저장 프로시저는 BeforeRetrieve 및 AfterRetrieve와 동일한 조작을 수행합니다. 이 프로시저에는 BeforeRetrieve 및 BeforeRetrieveUpdate 조작과 AfterRetrieve 및 AfterRetrieveUpdate 조작 모두를 커넥터가 수행하도록 하기 위해 개별 속성을 작성할 수 있도록 식별 이름이 있습니다.

커넥터는 임시 오브젝트를 작성하여 단일 카디널리티 하위 Business Object 또는 다중 카디널리티 하위 Business Object를 검색합니다. 다중 카디널리티 Business Object의

경우 커넥터는 BeforeRetrieveUpdate 저장 프로시저를 임시 Business Object에 적용합니다. AfterRetrieveUpdate 저장 프로시저는 컨테이너로 검색된 각 하위 오브젝트에 적용됩니다.

커넥터는 Business Object 메타 데이터나 Business Object의 저장 프로시저에서 동적으로 생성된 RETRIEVE 조회를 실행한 후 AfterRetrieveUpdate 저장 프로시저를 실행합니다. 이 경우 계층 구조 Business Object의 검색이 또한 해당 하위 Business Object를 검색해도 커넥터는 배열에 있는 모든 Business Object에서 AfterRetrieveUpdate 저장 프로시저를 실행합니다.

트랜잭션 확약 및 롤백

커넥터가 처리할 Business Object를 수신할 때마다 트랜잭션 블록을 시작합니다. Business Object 처리 중 커넥터가 실행하는 모든 SQL 문은 트랜잭션 블록 내에서 캡슐화됩니다. 커넥터가 Business Object 처리를 완료하면 처리에 성공한 경우 트랜잭션 블록을 확약하고 오류가 발생한 경우 트랜잭션을 롤백합니다.

Business Object 속성 등록 정보

Business Object 아키텍처는 속성에 적용되는 다양한 등록 정보를 정의합니다. 이 섹션에서는 커넥터가 등록 정보를 해석하는 방법을 설명하고 Business Object를 수정할 때 등록 정보를 설정하는 방법을 설명합니다.

Name 등록 정보

각 Business Object 속성에는 고유 이름이 있어야 합니다.

Type 등록 정보

각 Business Object 속성에는 Integer, String 또는 하위 Business Object의 유형과 같은 유형이 있어야 합니다. 커넥터가 Date, Long Text 또는 String 유형의 속성을 발견하면 커넥터는 값을 인용 부호로 묶어 문자 데이터로 처리합니다.

Cardinality 등록 정보

하위 또는 하위 Business Object의 배열을 나타내는 각 Business Object 속성은 이 속성에서 1 또는 n 값을 각각 가집니다. 하위 Business Object를 나타내는 모든 속성에는 ContainedObjectVersion 등록 정보(하위의 버전 번호 지정) 및 Relationship 등록 정보(Containment 값 지정)도 있습니다.

Max length 등록 정보

속성이 String 유형인 경우 이 등록 정보는 속성의 값으로 허용된 최대 길이를 지정합니다.

Key 등록 정보

각 Business Object에서 적어도 하나의 단순 속성을 키로 지정해야 합니다. 속성을 키로 정의하려면 이 등록 정보를 Yes로 설정하십시오. Business Object 속성 유형이 String인 경우 데이터베이스의 데이터 유형은 char 대신 Varchar을 사용하는 것이 좋습니다.

주: 커넥터는 하위 Business Object 또는 하위 Business Object 배열을 키 속성으로 표시하는 속성 지정을 지원하지 않습니다.

단순 속성에 대한 Key 등록 정보를 true로 설정하면 커넥터가 해당 속성을 Business Object 처리 시 생성하는 SELECT, UPDATE, RETRIEVE 및 DELETE SQL 문의 WHERE 절에 추가합니다.

상위/하위 관계를 하위에 저장하는(다중 카디널리티 및 단일 카디널리티 모두) 하위의 속성에 대한 키 등록 정보가 true로 설정되는 경우 커넥터가 SELECT 문의 WHERE 절에 있는 상위의 1차 키를 사용하고 Key 등록 정보는 사용하지 않습니다. 하위의 외부 키 속성을 설정하기 위해 속성 값이 사용되는 Business Object 속성의 이름 지정에 대한 정보는 65 페이지의 『속성 레벨의 응용프로그램 특정 정보』를 참조하십시오

Foreign key 등록 정보

커넥터는 이 등록 정보를 사용하여 속성이 외부 키인지 여부를 판별합니다.

Required 등록 정보

Required 등록 정보는 속성이 값을 포함해야 하는지 여부를 지정합니다.

단일 카디널리티 하위 Business Object를 나타내는 속성에 이 등록 정보를 지정하는 경우 커넥터는 이 속성의 하위 Business Object를 포함할 상위 Business Object를 필요로 합니다.

커넥터가 작성 요청으로 Business Object를 수신하는 경우 다음 중 두 가지 조건이 true이면 커넥터가 Create 조작을 실패하게 만듭니다.

- Business Object가 필수 속성에 대한 기본값 또는 유효한 값이 없습니다.
- 응용프로그램 특정 정보는 커넥터가 고유 ID를 생성하도록 지정하지 않습니다.

커넥터가 검색 요청으로 Business Object를 수신하고 Business Object가 필수 속성의 기본값 또는 올바른 값을 갖고 있지 않은 경우 커넥터는 Retrieve 조작을 실패하게 만듭니다.

커넥터는 하위 Business Object 배열을 포함하는 속성에 이 등록 정보를 사용하지 않습니다.

주: 키 속성이 순서 또는 카운터를 사용하거나 데이터베이스에 의해 채워지는 경우 (UID=AUTO) 필수로 표시되어서는 안됩니다.

AppSpecificInfo

이 등록 정보에 대한 정보는 65 페이지의 『속성 레벨의 응용프로그램 특정 정보』를 참조하십시오.

Default value 등록 정보

이 등록 정보는 데이터베이스 테이블의 값으로 채우지 않는 경우 커넥터가 단순 속성을 채우기 위해 사용하는 기본값을 지정합니다. 커넥터는 하위 Business Object 또는 하위 Business Object의 배열을 나타내는 속성에 대한 이 등록 정보를 평가하지 않습니다.

커넥터는 UseDefaults 구성 등록 정보가 true로 설정된 경우에만 이 등록 정보를 평가합니다. 자세한 정보는 19 페이지의 표 6을 참조하십시오

특수 속성 값

Business Object의 단순 속성에는 특수 값인 CxIgnore가 있을 수 있습니다. 통합 브로커에서 Business Object를 수신할 때 커넥터는 CxIgnore 값이 있는 모든 속성을 무시합니다. 이는 커넥터에서 해당 속성을 볼 수 없는 것과 같습니다.

커넥터가 데이터베이스에서 데이터를 검색하고 SELECT 문이 속성에 대한 null 값을 리턴할 때 커넥터는 이 속성의 값을 기본적으로 CxIgnore로 설정합니다. 속성의 응용프로그램 특정 정보의 UNVL 매개변수에 대한 값이 지정된 경우 커넥터가 이 값을 사용하여 null을 표시합니다.

커넥터는 모든 Business Object가 적어도 하나의 1차 키 속성을 가지도록 요청하기 때문에 개발자는 커넥터에 전달된 WebSphere Business Integration Adapter Business Object에 CxIgnore로 설정되지 않은 1차 키가 적어도 하나는 있는지 확인해야 합니다. 이 요구사항에 대한 유일한 예외는 1차 키가 카운터나 순서를 사용하여 커넥터에 의해 생성되거나 데이터베이스에 의해 생성되는 Business Object입니다.

커넥터가 데이터를 데이터베이스에 삽입하고 Business Object 속성에 지정된 값이 없을 때 속성의 UseNullValue 등록 정보에 의해 지정된 값을 사용합니다. UseNullValue에 대한 자세한 정보는 66 페이지의 표 11의 UNVL=value를 참조하십시오.

Business Object 응용프로그램 특정 정보

Business Object 정의의 응용프로그램 특정 정보는 Business Object 처리 방법에 대한 응용프로그램 종속 지시사항을 커넥터에 제공합니다. 커넥터는 Business Object의 속성이나 Verb에서 또는 Business Object 자체에서 응용프로그램 특정 정보를 구문 분석하여 작성, 갱신, 검색 또는 삭제 조작에 대한 조회를 생성합니다.

커넥터는 Business Object의 응용프로그램 특정 정보 일부를 캐시에 저장하고 이 정보를 사용하여 모든 Verb에 대한 조회를 빌드합니다.

응용프로그램 특정 Business Object를 확장하거나 수정하는 경우 Business Object 정의의 응용프로그램 특정 정보가 커넥터가 예상하는 구문에 일치하는지 확인해야 합니다.

이 섹션에서는 커넥터에서 지원하는 Business Object의 오브젝트 레벨, 속성 및 Verb 응용프로그램 특정 정보 형식에 대한 정보를 제공합니다.

표 10은 Business Object 응용프로그램 특정 정보에 사용 가능한 기능의 개요를 제공합니다.

표 10. 지원되는 Business Object에서 응용프로그램 특정 정보 개요

응용프로그램 특정 정보의 범위	기능
전체 Business Object	다음을 지정합니다. • 해당 데이터베이스 테이블의 이름 • 커넥터가 WHERE 절에서 값을 사용하여 논리(또는 소프트) 삭제를 수행하는 열을 정의합니다. • 최상위 레벨 Business Object가 랩퍼임을 지정합니다.
단순 속성	다음을 지정합니다. • 속성의 데이터베이스 열 이름. • 현재 Business Object의 속성과 상위나 하위 Business Object 간에 외부 키 관계. • 고유 ID 값의 자동 생성. • 커넥터가 값을 사용하여 현재 속성의 값을 설정해야 하는 동일한 Business Object 내 다른 속성의 이름. • 검색 정렬 시 현재 속성의 사용 여부. • 현재 속성의 값이 null일 때 사용할 값. • 문자열 대체 작동. • 문자열 비교 시 LIKE 연산자 또는 = 연산자를 사용할지 여부. • LIKE 연산자를 사용할 때 와일드 카드 위치로 사용할 값.
하위 또는 하위 Business Object의 배열을 포함하는 속성	상위가 단일 카디널리티 하위를 소유하는지 여부를 지정합니다. 소스 Business Object에 데이터가 표시되지 않는 경우 갱신 조작 시 커넥터가 하위 데이터를 삭제하는지 여부를 지정합니다.
Business Object Verb	Retrieve Verb에만 사용되는 경우 이 텍스트는 검색을 위해 WHERE 절에 포함될 속성을 지정합니다. 연산자와 속성 값도 지정할 수 있습니다.

다음 섹션에서는 이 기능에 대해 보다 자세히 설명합니다.

Business Object 레벨의 응용프로그램 특정 정보

Business Object 레벨에서 응용프로그램 특정 정보를 사용하면 다음과 같은 기능을 수행할 수 있습니다.

- 해당 데이터베이스 테이블의 이름을 지정합니다.
- 실제 또는 논리 삭제 수행에 필요한 정보를 제공합니다.
- 최상위 레벨 Business Object가 랩퍼 오브젝트라는 것을 지정합니다.

Business Object 레벨에서 응용프로그램 특정 정보 형식은 다음과 같이 콜론(:) 또는 세미콜론(;) 분리문자로 구분되는 매개변수로 구성됩니다.

```
TN=TableName; SCN=StatusColumnName:StatusValue
```

여기서 TableName은 데이터베이스 테이블을 식별하고 *StatusColumnName*은 논리 삭제를 수행하기 위해 사용되는 데이터베이스 열의 이름이며 *StatusValue*는 Business Object가 비활성화되거나 삭제됨을 의미하는 값입니다.

예를 들어 고객 Business Object가 해당 Business Object 응용프로그램 특정 정보에 다음 값을 지정한 것으로 가정하십시오.

```
TN=CUSTOMER; SCN=CUSTSTATUS:DELETED
```

또한 커넥터가 고객을 삭제하기 위한 요청을 수신한다고 가정합니다. 이 값은 커넥터가 다음 SQL 문을 발행하도록 합니다.

```
UPDATE CUSTOMER SET CUSTSTATUS = 'DELETED' WHERE CUSTOMER_ID = 2345
```

SCN 매개변수가 포함되지 않거나 값이 지정되지 않은 경우 커넥터는 데이터베이스에서 Business Object를 실제로 삭제합니다. 즉, Delete Verb가 있는 Business Object에 해당 응용프로그램 특정 정보의 SCN 매개변수를 포함하는 경우 커넥터는 논리 삭제를 수행합니다. Delete Verb가 있는 Business Object에 해당 응용프로그램 특정 정보의 SCN 매개변수가 포함되지 않는 경우 커넥터는 실제 삭제를 수행합니다.

갱신 및 삭제 조작은 모두 다음과 같이 SCN 등록 정보 값을 사용할 수 있습니다.

- 갱신을 수행할 때 커넥터는 ChildUpdatePhyDelete 등록 정보의 값을 사용하여 누락된 하위 데이터를 실제적으로 또는 논리적으로 삭제하는지 여부를 판별합니다. 하위 데이터를 논리적으로 삭제하는 경우 SCN 매개변수의 값을 사용하여 상태 열의 이름과 상태 값의 텍스트를 확보합니다. 자세한 정보는 48 페이지의 『Update 조작』을 참조하십시오
- 삭제를 수행할 때 커넥터는 SCN 매개변수의 값을 사용하여 전체 Business Object를 실제적으로 또는 논리적으로 삭제하는지 여부를 판별합니다. SCN 매개변수에 값이 포함되는 경우 커넥터는 논리 삭제를 수행합니다. SCN 매개변수에 값이 포함되지 않는 경우 커넥터는 실제 삭제를 수행합니다. 자세한 정보는 51 페이지의 『Delete 조작』을 참조하십시오

다음과 같이 Business Object 레벨에서 응용프로그램 특정 정보가 사용되어서 랩퍼를 지정할 수 있습니다.

```
WRAPPER=true|false
```

랩퍼 매개변수가 true로 설정된 경우 최상위 레벨 Business Object는 랩퍼 오브젝트입니다. 랩퍼 오브젝트는 데이터베이스 테이블이나 보기로 표시되지 않습니다. 랩퍼는 관련 없는 Business Object의 컨테이너로 사용됩니다. 커넥터는 최상위 레벨 오브젝트를 무시하며 하위만을 처리합니다. 랩퍼 오브젝트는 N 카디널리티나 N-1 카디널리티 엔티티 또는 둘 다를 포함할 수 있습니다.

속성 레벨의 응용프로그램 특정 정보

속성에 대한 응용프로그램 특정 정보는 속성이 단순 속성인지 또는 하위나 하위 Business Object의 배열을 나타내는 속성인지에 따라 다릅니다. 또한 하위를 나타내는 속성에 대한 응용프로그램 특정 정보는 상위/하위 관계가 하위에 또는 상위에 저장되는지에 따라 다릅니다. 하위나 하위 Business Object의 배열을 나타내는 속성에 대한 응용프로그램 특정 정보는 67 페이지의 『속성의 외부 키 지정』을 참조하십시오.

단순 속성에 대한 응용프로그램 특정 정보

단순 속성의 경우 응용프로그램 특정 정보 형식은 각각 매개변수 이름 및 해당 값을 포함하는 11가지 이름-값 매개변수로 구성됩니다. 각 매개변수 세트는 콜론(:) 분리문자에 의해 다음 세트와 구분됩니다.

속성 응용프로그램 특정 정보의 형식은 아래에 표시되어 있습니다. 대괄호([ ])는 선택적 매개변수를 묶습니다. 세로 막대(|)는 옵션 세트 구성원을 분리합니다. 콜론은 분리문자로 예약하십시오.

```
CN=col_name:[FK=[fk_object_name.]fk_attribute_name]:
[UID=[AUTO|uid_name| schema_name.uid_name[=UseIfMissing]|CW.uidcolumnname
[=UseIfMissing]]]:
[CA=set_attr_name| ..set_attr_name]:[OB=[ASC|DESC]]:[UNVL=value]:
[ESC=true|false]:[FIXEDCHAR=true|false]:
[BYTEARRAY=true|false]:[USE_LIKE=true|false:
[WILDCARD_POSITION=non-negative number|NONE|BEGIN|END|BOTH]]:
[CLOB=true]
```

커넥터가 처리해야 하는 단순 속성의 유일한 필수 매개변수는 열 이름입니다. 예를 들어 열 이름만을 지정하려면 다음 형식을 사용하십시오.

```
CN=customer_id
```

66 페이지의 표 11은 각 이름-값 매개변수를 설명합니다.

표 11. 속성 응용프로그램 특정 정보의 이름-값 매개변수

매개변수	설명
CN=*col_name*	이 속성에 대한 데이터베이스 열의 이름.
FK=[*fk_object_name*.]*fk_attribute_name*	이 등록 정보 값은 상위/하위 관계가 상위 Business Object에 저장되는지 하위에 저장되는지 여부에 따라 다릅니다. 속성이 외부 키가 아닌 경우 이 매개변수를 응용프로그램 특정 정보에 포함하지 마십시오. 자세한 정보는 67 페이지의 『속성의 외부 키 지정』을 참조하십시오
UID=AUTO	커넥터는 이 매개변수를 사용하여 Business Object의 고유 ID를 생성합니다. 속성이 고유 ID 생성을 필요로 하지 않는 경우 이 매개변수를 응용프로그램 특정 정보에 포함하지 마십시오. Business Object 처리 중 고유 ID의 보존에 대한 자세한 내용은 PreserveUIDSeq 등록 정보 설명을 참조하십시오. 자세한 정보는 70 페이지의 『Business Object의 고유 ID 생성』을 참조하십시오.
UID=*uid_name*| schema_name.uid_name [=UseIfMissing]	
UID=CW.*uidcolumnname*[=UseIfMissing]	**주:** CW는 UID의 유형을 표시하는 데 사용되는 키워드이며 테이블 이름을 표시하지 않습니다.
CA=*set_attr_name*| ..set_attr_name	set_attr_name이 현재 개별 Business Object 내 다른 속성의 이름으로 설정되는 경우 커넥터가 지정된 속성의 값을 사용하여 Create 조작 중 Business Object를 데이터베이스에 추가하기 전에 이 속성의 값을 설정합니다. set_attr_name 값은 하위 Business Object에서 속성을 참조할 수 없지만 set_attr_name 앞에 두 개의 점이 있을 경우 상위 Business Object의 속성을 참조할 수 있습니다. 이 매개변수를 응용프로그램 특정 정보에 포함하지 않는 경우 커넥터는 속성의 값(CA)을 다른 속성에서 복사하지 않고 현재 속성의 값을 사용합니다.
OB=[ASC|DESC]	이 매개변수에 값이 지정되고 하위 Business Object에 속성이 있는 경우 커넥터는 검색 조회의 ORDER BY 절에 있는 속성 값을 사용합니다. 커넥터는 하위 Business Object를 오름차순 또는 내림차순으로 검색할 수 있습니다. ASC를 사용하여 검색을 오름차순으로 지정하십시오. DESC를 사용하여 검색을 내림차순으로 지정하십시오. 이 매개변수를 응용프로그램 특정 정보에 포함시키지 않는 경우 커넥터가 검색 순서를 지정할 때 이 속성을 사용하지 않습니다.
UNVL=*value*	커넥터가 값이 널인 속성의 Business Object를 검색할 때 널을 표시하기 위해 사용하는 값을 지정합니다. 이 매개변수를 응용프로그램 특정 정보에 포함시키지 않는 경우 커넥터가 속성의 값에 CxIgnore를 삽입합니다.
ESC=[true|false]	커넥터가 ReplaceAllStr 등록 정보에서 식별된 각 문자의 모든 인스턴스를 ReplaceStrList 등록 정보에 지정된 대체 문자열로 바꾸는지 여부를 판별합니다. 이 매개변수가 값을 포함하지 않는 경우 커넥터가 ReplaceStrList 등록 정보의 값을 사용하여 이를 판별합니다. **주:** ESC 매개변수와 ReplaceAllStr 및 ReplaceStrList 등록 정보는 데이터베이스 ESC 문자 기능에 대한 지원을 제공합니다(예: 작은따옴표 ESC). JDBC 드라이버에서 제공한 Prepared 문에서도 동일한 기능이 사용 가능하기 때문에 이들 등록 정보는 커넥터의 이후 릴리스에서 제거됩니다. 현재 커넥터는 JDBC Prepared 문의 사용을 지원합니다.
FIXEDCHAR=true|false	테이블의 열이 VARCHAR가 아닌 CHAR 유형일 때 속성이 고정된 길이인지 여부를 지정합니다. 예를 들어 특정 속성이 CHAR 유형의 열에 링크되는 경우 커넥터는 FIXEDCHAR 길이를 예상합니다. 이 속성의 응용프로그램 특정 정보에 대해 FIXEDCHAR=true를 지정하십시오. 속성의 MaxLength 등록 정보가 데이터베이스에 지정되는 CHAR 길이인지 확인하십시오. 기본값은 FIXEDCHAR=false입니다.
BYTEARRAY=true|false	BYTEARRAY=true인 경우 커넥터는 데이터베이스로 2진 데이터를 쓰고 읽으며 ICS 또는 WebSphere MQ Integrator Broker의 문자열로 해당 데이터를 보냅니다. BYTEARRAY=false는 기본값입니다. 자세한 정보는 72 페이지의 『2진 데이터에 대한 작업』을 참조하십시오.

표 11. 속성 응용프로그램 특정 정보의 이름-값 매개변수 (계속)

매개변수	설명
USE_LIKE=true|false	커넥터가 문자열을 비교하는 데 = 연산자를 사용할지 또는 LIKE 연산자를 사용할지를 지정합니다. USE_LIKE가 true로 설정되면 WILDCARD_POSITION을 설정하여 와일드 카드 조회가 수행될 수 있습니다. USE_LIKE가 false로 설정되면 = 연산자가 사용됩니다.
WILDCARD_POSITION=non-negative number|NONE|BEGIN|END|BOTH	USE_LIKE가 true이면 와일드 카드인 위치를 지정하기 위해 WILDCARD_POSITION이 사용됩니다. 이 값은 음수가 아닌 임의의 숫자나 NONE, BEGIN, END 또는 BOTH가 될 수 있습니다. 예를 들면 BEGIN을 사용하면 문자열의 첫 번째 위치(%string)에 와일드 카드 문자를 놓습니다. END를 사용하면 문자열의 마지막 위치(string%)에 와일드 카드 문자를 놓습니다. BOTH는 문자열의 첫 번째와 마지막 위치 모두(%string%)에 와일드 카드 문자를 놓습니다.
CLOB=true	문자열 속성 유형의 경우에만 적용할 수 있습니다. 이 속성에 해당하는 데이터베이스 열이 CLOB 데이터 유형임을 지정합니다. **주:** CLOB 데이터 유형은 다음과 같이 정의됩니다. • CLOB 속성의 유형은 CLOB의 길이를 정의하는 데 사용되는 길이가 있는 문자열 유형입니다. • CLOB 속성은 ASI=CN=xyz; CLOB=true입니다. • ASI에서 CLOB를 참조하는 다른 속성 유형에서는 오류가 발생합니다. • CLOB=false의 경우 오류가 발생합니다. 일반 문자열 유형은 동일하며 AIS에서 CLOB를 참조하지 않습니다. 4KB 이상의 CLOB 데이터 유형을 삽입하거나 갱신할 수 있습니다. 그러나 이 유형은 Oracle에서만 사용할 수 있으며 CLOB가 지원되는 최신의 thin 드라이버가 필요합니다. 다른 드라이버를 사용하면 오류가 발생할 수 있습니다.

주: 커넥터가 조회를 빌드하거나 실행하도록 하는 Business Object 속성의 응용프로그램 특정 정보가 없는 경우 커넥터가 경고를 기록하고 조작을 계속합니다. 예외를 발행하거나 실패를 리턴하지 않습니다.

속성의 외부 키 지정: 이 등록 정보의 값은 상위/하위 관계가 상위 Business Object 또는 하위 Business Object에 저장되는지에 따라 다릅니다.

- 상위에 저장됨 - 하위 Business Object의 유형과 외부 키로 사용될 하위의 속성 이름 모두를 포함하도록 값을 설정하십시오.
- 하위에 저장됨 - 외부 키로 사용될 상위의 속성 이름만 포함하도록 값을 설정하십시오.

fk_object_name의 값이 하위 Business Object의 유형과 일치하지 않고 *fk_attribute_name*의 값이 상위 또는 하위(적용 가능한 대로)의 속성 이름과 일치하지 않는 경우 커넥터가 이 속성을 외부 키로 처리할 수 없습니다. Business Object의 이름과 속성 이름의 대소문자는 구분됩니다.

예를 들어 고객 Business Object가 주소 하위 Business Object를 나타내는 Addr[1] 속성과 하위 Business Object의 1차 키를 외부 키로 저장하는 AID 속성을 포함한다고

가정하십시오. 이 경우 상위 외부 키 속성의 응용프로그램 특정 정보는 하위 Business Object의 유형(주소)과 1차 키 속성의 이름(ID)을 포함해야 합니다. 이 예에서 AID 속성의 응용프로그램 특정 정보는 FK=Address.ID를 포함하게 됩니다.

외부 키 속성 이름 지정: 여러 상위 Business Object는 단일 또는 다중 카디널리티로 하위가 저장되는지 여부에 관계 없이 또한 상위 또는 하위에 상위/하위 관계가 저장되는지 여부에 관계 없이 동일한 하위 Business Object를 포함할 수 있습니다. 그러나 상위/하위 관계를 저장하는 모든 상위 Business Object는 동일한 이름의 속성을 사용하여 하위의 1차 키를 포함해야 합니다. 또한 상위/하위 관계를 저장하는 모든 하위 Business Object는 동일한 이름의 속성을 사용하여 상위의 1차 키를 포함해야 합니다. 그림 5는 이들 관계를 보여 줍니다.

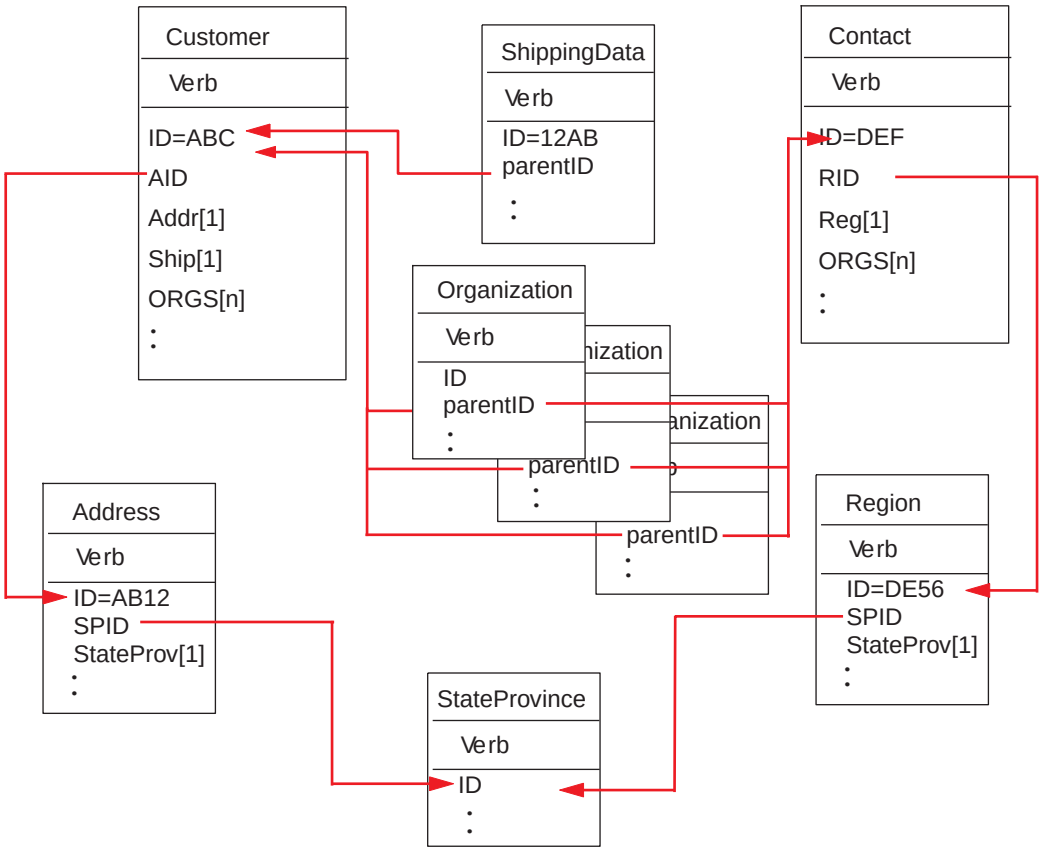


그림 5. Business Object 간의 관계 예

그림 5는 다음 관계에 대해합니다.

- 고객 ABC 및 담당자 DEF의 ORGS[n] 속성은 조직 Business Object 배열을 나타냅니다.조직의 배열에 있는 각 Business Object에 대한 외부 키 값은 고객 및 담당자 Business Object에 있는 ID 속성의 1차 키 값에 해당합니다. 이 경우 배열의 각 Business Object는 여러 상위에 의해 포함됩니다.

  ORGS 속성의 응용프로그램 특정 정보는 다음과 같습니다.

```
KEEP_RELATIONSHIP=true
```

KEEP_RELATIONSHIP 매개변수에 대한 자세한 정보는 71 페이지의 『하위를 나타내는 속성에 대한 응용프로그램 특정 정보』를 참조하십시오.

구성 배열 내 각 하위의 parentID 속성에 대한 응용프로그램 특정 정보는 현재 속성과 일치하는 데이터베이스 내에 열 이름을 포함시키고 상위의 1차 키 속성 이름을 포함시킴으로써 현재 속성의 외부 키를 지정합니다. 예를 들어 다음과 같습니다.

```
CN=ORG_ID:FK=ID
```

주: 여러 Business Object가 동일한 하위를 포함하도록 하려면(상위/하위 관계가 하위에 저장되는 경우) 모든 상위 Business Object가 동일한 이름의 속성을 사용하여 하위의 외부 키를 포함하도록 해야 합니다. 해당 하위의 응용프로그램 특정 정보의 외부 키 매개변수는 상위 Business Object 유형이 아닌 속성 이름만을 식별합니다. 커넥터는 직접 상위가 각 하위의 소유자라고 가정합니다.

- 고객의 Addr[1] 속성은 소유권이 있는 주소 Business Object를 나타냅니다. 고객의 AID 속성은 상위의 외부 키로서 주소 Business Object의 1차 키를 식별합니다. 이 경우 상위의 외부 키 속성은 하위 Business Object의 유형과 1차 키 속성의 이름을 포함해야 합니다. 단일 카디널리티 하위인 주소는 한 상위에만 포함됩니다.

  Addr 속성에 대한 응용프로그램 특정 정보는 다음과 같습니다.

  ```
  CONTAINMENT=OWNERSHIP
  ```

  AID 속성에 대한 응용프로그램 특정 정보는 현재 속성에 일치하는 데이터베이스의 열 이름을 포함하며 하위 Business Object의 유형과 1차 키 속성의 이름을 포함하여 현재 속성의 외부 키를 지정합니다. 예를 들어 다음과 같습니다.

  ```
  CN=FK_AD:FK=Address.ID
  ```

  하위의 1차 키 속성에 대한 응용프로그램 특정 정보는 다음과 같습니다.

  ```
  CN=pk
  ```

- 주소 및 지역 Business Object의 StateProv[1] 속성은 소유권이 없는 StateProvince Business Object를 나타냅니다. 주소 및 지역 Business Object의 SPID 속성에는 상위 외부 키 기능을 수행하는 해당 1차 키 속성 이름 및 하위 Business Object(StateProvince) 유형이 포함됩니다. 동일한 단일 카디널리티 하위인 StateProvince는 복수 상위에 포함됩니다.

  SPID 속성의 응용프로그램 특정 정보는 다음과 같습니다.

  ```
  CONTAINMENT=NO_OWNERSHIP
  ```

  CONTAINMENT 매개변수에 대한 자세한 정보는 71 페이지의 『하위를 나타내는 속성에 대한 응용프로그램 특정 정보』를 참조하십시오.

주소 SPID 속성에 대한 응용프로그램 특정 정보는 현재 속성과 일치하는 데이터베이스 내 열 이름을 포함시키고 하위 Business Object 유형 및 해당 1차 키 속성 이름을 포함시킴으로써 현재 속성의 외부 키를 지정합니다. 예를 들어 다음과 같습니다.

```
CN=FK_SP:FK=StateProvince.ID
```

하위의 1차 키 속성에 대한 응용프로그램 특정 정보는 다음과 같습니다.

```
CN=SP_ID
```

주: 여러 Business Object(상위/하위 관계를 상위에 저장)가 동일한 하위를 포함하도록 하려면 모든 하위 Business Object가 동일한 이름의 속성을 사용하여 상위의 외부 키를 포함하도록 해야 합니다.

- 고객의 Ship[1] 속성은 고객의 선적 정보를 포함하는 ShippingData Business Object를 나타냅니다. 고객의 ID 속성은 선적 데이터의 외부 키로서 기능합니다. 이런 경우 ShippingData가 해당 상위에 관계 없이 존재할 수 없고 해당 상위를 작성해야만 작성되므로 상위/하위 관계는 하위에 저장됩니다.

  하위의 parentID 속성에 대한 응용프로그램 특정 정보는 현재 속성과 일치하는 데이터베이스 내 열 이름을 포함시키고 상위의 1차 키 속성 이름을 포함시킴으로써 현재 속성의 외부 키를 지정합니다. 예를 들어 다음과 같습니다.

  ```
  CN=SD_ID:FK=ID
  ```

Business Object의 고유 ID 생성: 커넥터는 UID 매개변수를 사용하여 Business Object의 고유 ID를 생성합니다. 커넥터는 순서(Oracle의 경우) 또는 카운터(테이블로 구조화됨)를 사용하여 고유 ID를 생성한 후 INSERT 문을 발행합니다.

커넥터는 순서나 카운터를 사용하여 ID 값을 생성한 다음 INSERT 문을 발행합니다.

- UID =*uid_name*인 경우 *uid_name*의 값은 커넥터가 속성의 고유 ID를 생성하는 데 사용하는 Oracle 순서 이름을 제공합니다. 커넥터가 순서값을 페치하고 나면 키 속성을 채우고 INSERT 문을 발행합니다.
- UID =*uid_name*=UseIfMissing이고 속성 값이 CxIgnore가 아닌 경우 커넥터는 고유 ID를 생성하기 보다는 속성 값을 사용합니다. =UseIfMissing 매개변수에는 공백이 포함될 수 없으며 대소문자를 구분하지 않습니다.
- UID=CW.*uidcolumnname*인 경우 커넥터는 카운터 테이블을 사용하여 속성에 대한 고유 ID를 생성합니다. id로 이름 지정된 단일 열이 있고 이름을 구성할 수 있는 테이블이 작성됩니다. 테이블을 사용자 정의하여 UID 생성을 필요로 하는 각 속성의 열을 추가할 수 있습니다. *uidcolumnname* 매개변수를 사용하여 고유 ID를 생성할 때 사용할 커넥터에 대한 열의 이름을 지정하십시오. 커넥터는 UID를 생성해야 하는 열의 숫자 데이터 유형만을 지원함에 유의하십시오.

  테이블 이름 구성에 대한 정보는 UniqueIDTableName을 참조하십시오. 이 테이블 설치를 위한 스크립트는 다음과 같습니다.

\connectors\Manugistics\dependencies\BIA_uid_table_oracle.sql

- UID=CW.*uidcolumnname*=UseIfMissing이고 속성 값이 CxIgnore가 아닌 경우 커넥터는 고유 ID를 생성하기 보다는 속성 값을 사용합니다. =UseIfMissing 매개변수에는 공백이 포함될 수 없으며 대소문자를 구분하지 않습니다.

처리 중 고유 ID 순서 보존에 대해서는 27 페이지의『PreserveUIDSeq』 등록 정보를 참조하십시오

하위를 나타내는 속성에 대한 응용프로그램 특정 정보

단일 카디널리티 하위 Business Object를 나타내는 속성은 하위가 상위에 의해 소유되는지 또는 복수 상위 사이에 공유되는지 여부를 지정할 수 있습니다.

단일 카디널리티 하위나 하위 Business Object의 배열을 나타내는 속성은 상위를 갱신할 때 커넥터의 작동과 하위의 서브세트를 지정할 수 있습니다.

단일 카디널리티 하위 Business Object를 나타내는 속성: 단일 카디널리티 하위 Business Object를 나타내는 속성에 대한 응용프로그램 특정 정보의 형식은 다음과 같습니다.

```
CONTAINMENT= [OWNERSHIP|NO_OWNERSHIP]
```

상위가 하위 Business Object를 소유하는 단일 카디널리티 관계를 나타내려면 CONTAINMENT를 OWNERSHIP으로 설정하십시오. 상위가 하위 Business Object를 공유하는 단일 카디널리티 관계를 나타내려면 CONTAINMENT를 NO_OWNERSHIP으로 설정하십시오. 상위가 아닌 하위에 관계를 저장하는 단일 카디널리티 관계를 나타낼 때 CONTAINMENT 매개변수를 포함하지 마십시오.

자세한 정보는 39 페이지의『단일 카디널리티 관계 및 소유권이 없는 데이터』 및 41 페이지의『하위에 관계를 저장하는 단일 카디널리티 관계』를 참조하십시오.

상위의 키를 저장하는 하위를 나타내는 속성: 상위/하위 관계를 하위에 저장하는 Business Object의 배열에 대한 Update 조작의 경우 하위를 나타내는 속성에 대한 특수 값이 있습니다. KEEP_RELATIONSHIP을 true로 설정하여 커넥터가 소스 Business Object에 표시되지 않는 기존 하위 데이터를 삭제하지 않도록 할 수 있습니다.

예를 들어 기존 계약이 기존 지역(예: New York)과 연관되어 있는 것으로 가정하십시오. 또한 커넥터가 San Francisco를 사이트로 연관시키는 단일 하위 Business Object를 포함하는 계약 Business Object를 갱신하는 요청을 수신한다고 가정합니다. 사이트 데이터를 나타내는 속성에 대한 KEEP_RELATIONSHIP이 true로 평가되는 경우 커넥터가 계약을 갱신하여 San Francisco와의 연관을 추가하고 New York과의 연관을 삭제하지 않습니다.

그러나 KEEP_RELATIONSHIP이 false로 평가되는 경우 커넥터가 소스 Business Object에 포함되지 않은 기존의 모든 하위 데이터를 삭제합니다. 이런 경우 계약은 San Francisco와만 연관됩니다.

이 응용프로그램 특정 정보에 대한 형식은 다음과 같습니다.

```
KEEP_RELATIONSHIP=[true|false]
```

이 응용프로그램 특정 정보에 대해 검사할 때 대소문자는 구별되지 않습니다.

2진 데이터에 대한 작업: BYTEARRAY=true이면 커넥터가 데이터베이스로 2진 데이터를 읽고 씁니다. WebSphere Business Integration System의 현재 버전에서는 2진 데이터를 위한 지원이 없기 때문에 2진 데이터가 String으로 변환된 후에 통합 브로커로 전송됩니다. 이 문자열의 형식은 바이트당 2자를 가지는 16진수입니다. 예를 들면 데이터베이스의 2진 데이터가 (10진수) 값(1, 65, 255)을 가지는 3바이트이면 문자열이 "0141ff"입니다.

Verb에 대한 응용프로그램 특정 정보 형식

커넥터는 Retrieve와 RetrieveByContent Verb를 위해 Verb 응용프로그램 특정 정보를 사용합니다. 이 텍스트를 사용하면 검색을 위해 WHERE 절에 포함될 속성을 지정할 수 있습니다. 연산자와 속성 값도 지정할 수 있습니다.

Retrieve와 RetrieveByContent Verb를 위한 응용프로그램 특정 정보의 구문은 아래에 표시됩니다.

[*condition_variable conditional_operator* @ [...]:[..]*attribute_name* [, ...]]

여기서,

condition_variable	데이터베이스 열 이름입니다.
conditonal_operator	데이터베이스에서 지원하는 연산자는 예를 들어 =, >, OR, AND 및 IN(*value1*,*value2*)입니다.
@	getAttrValue(*attribute_name*)로 검색되는 값으로 대체되는 변수. 대체는 전후 위치에 의존합니다. 즉, 커넥터는 첫 번째 @를 : 분리문자 다음에 지정된 첫 번째 *attribute_name* variable의 값으로 대체합니다.
..	attribute_name 변수에서 지정되는 속성은 바로 다음 상위 Business Object에 속합니다. 이 값이 누락되는 경우 속성은 현재 Business Object에 속합니다.
attribute_name	커넥터가 @을 속성 값으로 대체하는 속성의 이름.

이 등록 정보의 구문을 이해하려면 항목 Business Object에 값이 XY45인 item_id 속성과 값이 RED인 Color 속성이 있다고 가정하십시오. 또한 Retrieve Verb의 AppSpecificInfo 등록 정보를 다음과 같이 지정한다고 가정하십시오.

```
Color='RED'
```

위의 응용프로그램 특정 정보 값은 커넥터가 검색을 위해 다음 WHERE 절을 빌드하도록 합니다.

```
where item_id=XY45 and Color = 'RED'
```

보다 복잡한 예의 경우 고객 Business Object에 값이 1234인 customer_id 속성과 값이 01/01/90인 creation_date 속성이 있다고 가정하십시오. 또한 이 Business Object의 상위에 값이 20인 quantity 속성이 있다고 가정하십시오.

또한 Retrieve Verb의 AppSpecificInfo 등록 정보를 다음과 같이 지정한다고 가정하십시오.

```
creation_date > @ OR quantity = @ AND customer_status IN
('GOLD', 'PLATINUM') : creation_date, ..quantity
```

위의 응용프로그램 특정 정보 값은 커넥터가 검색을 위해 다음 WHERE 절을 빌드하도록 합니다.

```
where customer_id=1234 and creation_date > '01/01/90'
OR quantity = 20 AND customer_status IN ('GOLD', 'PLATINUM')
```

커넥터는 현재 Business Object의 creation_date 속성에서 날짜 값('01/01/90')을 확보합니다. 이는 상위 Business Object의 수량 속성에서 수량 값(20)을 확보합니다(응용프로그램 특정 정보의 ..quantity에 표시된 대로).

커넥터가 Retrieve Verb의 응용프로그램 특정 정보 구문을 분석하고 나면 Business Object의 1차 또는 외부 키에서 구성하는 RETRIEVE 문의 WHERE 절에 텍스트를 추가합니다. 커넥터는 선행 AND를 WHERE 절에 추가합니다. 응용프로그램 특정 정보 값은 올바른 SQL 구문이어야 합니다. RetrieveByContent의 경우에는 응용프로그램 특정 정보가 값이 채워진 Business Object의 속성에서 구성되는 RETRIEVE 문의 WHERE 절로 추가됩니다.

WHERE 절은 상위 Business Object의 실제 속성 대신에 위치 표시기 속성을 참조할 수도 있습니다. 위치 표시기에는 응용프로그램 특정 정보가 없으며 속성의 ASI가 다음 조건 중 하나를 만족할 경우 위치 표시기가 될 수 있습니다.

1. ASI=null 또는 ''인 단순 속성
2. ASI=PH=TRUE인 단순 속성

예를 들어 주문 Business Object가 다중 카디널리티 행 항목 Business Object를 포함하는 경우 특정 행 항목 검색만이 필요합니다. 이 검색은 주문 Business Object의 위치 표시기 속성을 통해서만 처리될 수 있습니다. 하위 오브젝트는 모든 제거(prune)되므로 이 위치 표시기 속성은 상위 오브젝트에서 필요합니다. 위치 표시기 속성은 런타임 시 통합 브로커에 의해 쉼표(,)로 분리되는 특정 행 항목 목록과 함께 채울 수 있습니다.

이 예의 경우 하위 행 항목 Business Object에 있는 Retrieve Verb의 WHERE 절에 다음 정보를 추가하게 됩니다.

```
line_item_id in(@,@,@):..placeholder1,..placeholder2,..placeholder3
```

여기서 `line_item_id`는 하위 Business Object의 ID이며 `placeholder`는 상위 속성입니다. `placeholder`에 `12,13,14` 값이 포함되는 경우 조회를 통해 WHERE 절에서 다음을 선택하게 됩니다.

```
line_item_id in(12,13,14)
```

여기서 (1,2,3)의 `SELECT:..FROM:..WHERE`*x*는 표준 데이터베이스 SQL 구문입니다.

RetrieveByContent Verb에서 WHERE 절의 길이가 0이면 커넥터가 RETRIEVE 문의 WHERE 절에 있는 응용프로그램 특정 정보를 사용합니다. 이 기능으로 사용자가 속성 값이 채워지지 않은 Business Object를 전송하고 RetrieveByContent의 Verb 응용프로그램 특정 정보를 지정하며 커넥터가 Verb 응용프로그램 특정 정보에 지정된 것만을 기본으로 하여 WHERE 절을 빌드할 수 있습니다.

제 4 장 Manugistics용 IBM ODA를 사용한 Business Object 정의 생성

이 장에서는 Connector for JDBC에 대한 Business Object 정의를 생성하는 IBM ODA for Manugistics인 ODA(Object Discovery Agent)에 대해 설명합니다. 커넥터가 테이블 기반 또는 보기 기반의 오브젝트에 대해 작동하기 때문에 ODA for Manugistics는 데이터베이스 테이블과 보기를 사용하여 JDBC 데이터 소스에 고유한 Business Object 요구사항을 전개합니다.

주: 데이터베이스 개념 및 JDBC 드라이버(구성 목적의 경우)에 익숙하면 ODA for Manugistics 작동 원리 이해에 도움이 될 수 있습니다.

이 장은 다음 섹션으로 구성됩니다.

설치 및 사용법

이 섹션은 다음에 대해 논의합니다.

ODA for Manugistics 설치

ODA for Manugistics를 설치하려면 IBM WebSphere Business Integration Adapter 설치 프로그램을 사용하십시오. *UNIX용 시스템 설치 안내서* 또는 *Windows용 시스템 설치 안내서*의 지시사항을 따르십시오. 설치가 완료되면 다음 파일이 제품을 설치한 시스템의 디렉토리에 설치됩니다.

- `ODA\Manugistics\BIA_ManugisticsODA.jar`
- `ODA\messages\BIA_ManugisticsODAAgent.txt`

- ODA\messages\BIA_ManugisticsODAAgent_*ll*_*TT*.txt(언어(*ll*) 및 국가 또는 지역(*TT*) 특정 메시지 파일).
- ODA\Manugistics\start_ManugisticsODA.bat(Windows 전용)
- ODA/Manugistics/start_ManugisticsODA.sh(UNIX 전용)
- bin\CWODAEnv.bat(Windows 전용)
- bin/CWODAEnv.sh(UNIX 전용)

주: 별도로 표시되지 않는 한 이 책에서는 디렉토리 경로 규약으로 백슬래시(\)를 사용합니다. UNIX 설치의 경우 백슬래시를 슬래시(/)로 대체하십시오. 모든 제품 경로 이름은 제품이 사용자 시스템에 설치된 디렉토리와 관련됩니다.

ODA for Manugistics 사용하기 전에

ODA for Manugistics를 실행하려면 다음을 수행해야 합니다.

- 적절한 JDBC 드라이버를 설치하십시오.

  중요 사항: ODA for Manugistics는 JDBC 2.0 이상을 지원하는 JDBC 드라이버를 사용하여 임의 데이터베이스에 연결할 수 있습니다.

- ODA for Manugistics가 대응되는 데이터베이스 테이블과 열의 이름에서 Business Object 이름과 속성 이름을 생성하며 Business Object 이름과 속성 이름이 ISO Latin-1에 있어야 하므로 적합한 데이터베이스 구성요소가 Latin-1 이름을 보유하는지 확인하십시오. 그렇지 않은 경우에는 다음 중에서 선택하십시오.
  - Business Object Designer에서 직접 Business Object 정의를 작성하십시오.
  - 모든 Business Object 이름 및 속성 이름이 Latin-1이 되도록 ODA for Manugistics가 생성한 정의를 편집하십시오.
- 쉘 및 일괄처리 파일을 열어 편집한 후 표 12에 설명되어 있는 대로 값을 구성하십시오.

표 12. 쉘 및 일괄처리 파일 구성 변수

변수	설명	예
AGENTNAME	ODA의 이름	**UNIX:** AGENTNAME=ManugisticsODA **Windows:** set AGENTNAME=ManugisticsODA
AGENT	ODA jar 파일의 이름	**UNIX:** AGENT=$CROSSWORLDS/ODA/Manugistics/BIA_ManugisticsODA.jar **Windows:** set AGENT= %CROSSWORLDS%\ODA\Manugistics\BIA_ManugisticsODA.jar

표 12. 쉘 및 일괄처리 파일 구성 변수 (계속)

변수	설명	예
DRIVERPATH	JDBC 드라이버 라이브러리 경로. ODA for Manugistics는 드라이버 클래스를 사용하여 지정된 데이터베이스 연결을 설정합니다.	**UNIX:** DRIVERPATH=$CROSSWORLDS/lib/ \ xwutil.jar:$CROSSWORLDS/lib/ \ xwbase.jar:$CROSSWORLDS/lib/ \ xwsqlserver.jar:$CROSSWORLDS/lib/ \ spy/lib/spy.jar **Windows:** set DRIVERPATH=%CROSSWORLDS%\ / lib\xwutil.jar;%CROSSWORLDS%\lib\ / xwbase.jar;%CROSSWORLDS%\lib\ / xwsqlserver.jar;%CROSSWORLDS%\lib\ / spy\lib\spy.jar
DRIVERLIB	JDBC 드라이버가 사용하는 기본 라이브러리의 경로	**UNIX:** DRIVERLIB=$CROSSWORLDS/bin/db2jdbc.dll **Windows:** DRIVERLIB=%CROSSWORLDS%\bin\ db2jdbc.dll

JDBC 드라이버를 설치하고 쉘 또는 일괄처리 파일에 구성 값을 설정한 후 다음을 수행하여 Business Object를 생성해야 합니다.

1. ODA를 실행하십시오.
2. Business Object Designer를 실행하십시오.
3. Business Object Designer의 6단계 프로세스를 따라 ODA를 구성하고 실행하십시오.

다음 섹션에서는 이 단계에 대해 자세히 설명합니다.

ODA for Manugistics 실행

사용 중인 운영 체제에 해당하는 시작 스크립트로 ODA for Manugistics를 실행할 수 있습니다.

UNIX:

```
start_ManugisticsODA.sh
```

Windows:

```
start_ManugisticsODA.bat
```

Business Object Designer를 사용하여 ODA for Manugistics를 구성하고 실행합니다. Business Object Designer는 각 스크립트 또는 일괄처리 파일의 AGENTNAME 변수에 지정된 이름으로 각각의 ODA를 찾습니다. 이 커넥터에 대한 기본 ODA 이름은 ManugisticsODA입니다.

ODA for Manugistics의 복수 인스턴스 실행

ODA for Manugistics의 복수 인스턴스를 실행할 때 ODA의 이름을 변경하는 것이 바람직합니다. 추가로 고유하게 이름 지정된 ODA for Manugistics의 인스턴스를 작성하려면 다음을 수행하십시오.

- 각 인스턴스에 개별 스크립트나 일괄처리 파일을 작성하십시오.
- 각 스크립트나 일괄처리 파일의 AGENTNAME 변수에 고유 이름을 지정하십시오.

다른 시스템에서 ODA 인스턴스를 실행할 때에는 각 이름의 접두부를 호스트 시스템의 이름으로 하는 것이 좋습니다.

80 페이지의 그림 6에서는 실행할 ODA를 선택하는 Business Object Designer 창에 대해 설명합니다.

오류 및 추적 메시지 파일에 대한 작업

오류 및 추적 메시지 파일(기본값은 BIA_ManugisticsODAAgent.txt)은 제품 디렉토리 아래에 있는 \ODA\messages\에 있습니다. 이 파일은 다음 이름 지정 규칙을 사용합니다.

*AgentName*Agent.txt

ODA 스크립트나 일괄처리 파일의 복수 인스턴스를 작성하고 표시된 각 ODA에 고유 이름을 제공하는 경우 각 ODA 인스턴스에 대한 메시지 파일을 가질 수 있습니다. 또는 동일한 메시지 파일을 사용하는 다르게 이름 지정된 ODA를 가질 수 있습니다. 유효한 메시지 파일을 지정하는 방법에는 두 가지가 있습니다.

- ODA 이름을 변경하고 해당 메시지 파일을 작성하지 않는 경우 ODA 구성의 일부로서 Business Object Designer에서 메시지 파일 이름을 변경해야 합니다. Business Object Designer가 메시지 파일에 대한 이름을 제공하지만 실제로 파일을 작성하지는 않습니다. ODA 구성의 일부로 표시된 파일이 존재하지 않는 경우 기존 파일을 가리키도록 값을 변경하십시오.
- 특정 ODA의 기존 메시지 파일을 복사하여 필요한 대로 수정할 수 있습니다. Business Object Designer는 각 파일의 이름을 이름 지정 규칙에 따라 지정한다고 가정합니다. 예를 들어 AGENTNAME 변수가 ManugisticsODA1을 지정하는 경우 도구는 연관된 메시지 파일의 이름이 JDBCODA1Agent.txt라고 가정합니다. 따라서 Business Object Designer가 ODA 구성의 일부로 유효성 검증을 위한 파일 이름을 제공하는 경우 파일 이름은 ODA 이름을 기반으로 합니다. 기본 메시지 파일의 이름이 올바르게 지정되었는지 확인하고 필요한 대로 이를 정정하십시오.

중요 사항: ODA를 구성할 때 메시지 파일의 이름을 올바르게 지정하는 데 실패하면 이는 메시지 없이 실행됩니다. 메시지 파일 이름 지정에 대한 자세한 정보는 80 페이지의 『초기화 등록 정보 구성』을 참조하십시오.

구성 프로세스 중 다음을 지정하십시오.

- ODA for Manugistics가 오류 및 추적 정보를 기록하는 파일 이름
- 0 - 5 범위의 추적 레벨

표 13에서는 이 값에 대해 설명합니다.

표 13. 추적 레벨

추적 레벨	설명
0	모든 오류 로그
1	메소드에 대한 모든 입력 및 종료 메시지 추적
2	ODA의 등록 정보와 값 추적
3	모든 Business Object의 이름 추적
4	모든 하위 스레드의 세부사항 추적
5	• 모든 자체 등록 정보에 대해 ODA 초기화 값을 표시 • ODA for Manugistics가 생성한 각 스레드의 세부 상태를 추적 • Business Object 정의 덤프를 추적

이 값 구성 장소에 대한 정보는 80 페이지의 『초기화 등록 정보 구성』을 참조하십시오.

Business Object Designer의 ODA for Manugistics 사용

이 섹션에서는 Business Object Designer에서 ODA for Manugistics를 사용하여 Business Object 정의를 생성하는 방법에 대해 설명합니다. Business Object Designer 실행에 대한 정보는 *Business Object Development Guide*를 참조하십시오.

ODA를 실행한 후 Business Object Designer를 실행하여 이를 구성하고 실행해야 합니다. Business Object Designer에서 ODA를 사용하여 Business Object 정의를 생성하기 위한 여섯 단계가 있습니다. Business Object Designer에서는 각 단계를 안내하는 마법사를 제공합니다.

ODA를 시작한 후 다음을 수행하여 마법사를 시작하십시오.

1. Business Object Designer를 여십시오.
2. 파일 메뉴에서 ODA를 사용하여 새로 작성... 서브메뉴를 선택하십시오.

   Business Object Designer는 마법사에서 에이전트 선택이라는 첫 번째 창을 표시합니다. 80 페이지의 그림 6은 이 창에 대해 설명합니다.

ODA를 선택, 구성 및 실행하려면 다음 단계를 수행하십시오.

1. 80 페이지의 『ODA 선택』
2. 80 페이지의 『초기화 등록 정보 구성』
3. 82 페이지의 『노드 펼치기와 테이블, 보기 및 저장 프로시저 선택』
4. 83 페이지의 『데이터베이스 오브젝트 선택사항 확인』
5. 84 페이지의 『정의 생성』 및 선택적으로 84 페이지의 『추가 정보 제공』
6. 86 페이지의 『정의 저장』

ODA 선택

그림 6은 Business Object Designer의 여섯 단계 마법사의 첫 번째 대화 상자를 보여 줍니다. 이 창에서 실행할 ODA를 선택하십시오.

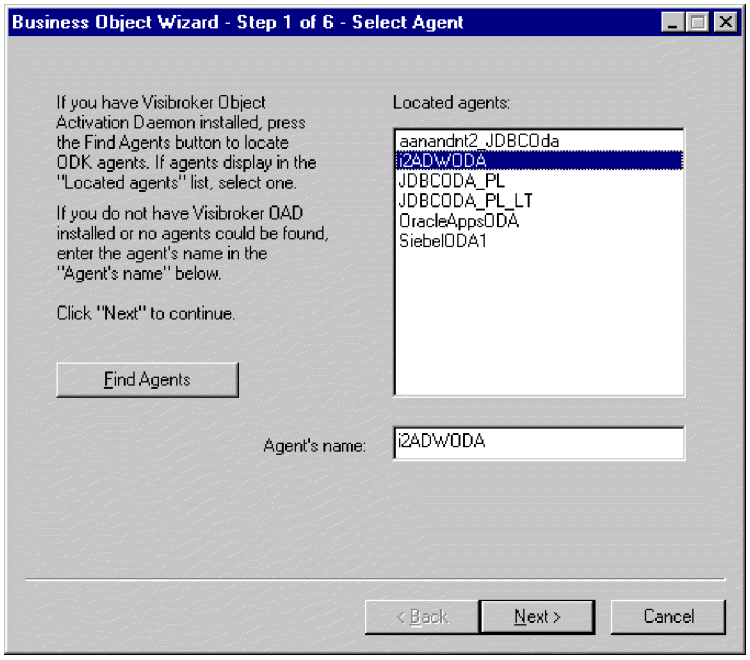


그림 6. ODA 선택

ODA를 선택하려면 다음을 수행하십시오.

1. 에이전트 찾기 단추를 눌러 찾은 에이전트 필드에서 등록된 모든 ODA 또는 현재 실행 중인 ODA를 표시하십시오.

   주: Business Object Designer가 원하는 ODA를 찾지 못하는 경우 ODA의 설정을 검사하십시오.

2. 표시된 목록에서 원하는 ODA를 선택하십시오.

   Business Object Designer가 에이전트의 이름 필드에 선택사항을 표시합니다.

초기화 등록 정보 구성

처음으로 Business Object Designer가 ODA for Manugistics와 통신하게 되면 그림 7에 표시되어 있는 대로 초기화 등록 정보 세트를 입력하도록 프롬프트합니다. ODA for Manugistics를 사용할 때마다 다시 입력할 필요가 없도록 이름 지정된 프로파일에 이러한 등록 정보를 저장할 수 있습니다. ODA 프로파일 지정에 대한 정보는 *Business Object Development Guide*를 참조하십시오.

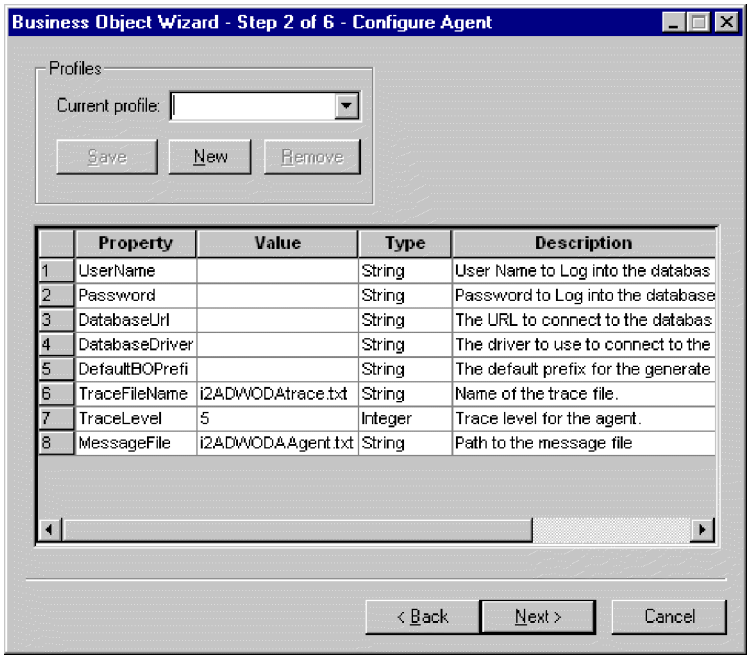


그림 7. 에이전트 초기화 등록 정보 구성

표 14에서 설명한 ODA for Manugistics 등록 정보를 구성하십시오.

표 14. ODA for Manugistics 등록 정보

행 번호	등록 정보 이름	등록 정보 유형	설명
1	UserName	문자열	데이터베이스 연결 권한이 있는 사용자의 이름
2	Password	문자열	데이터베이스 연결 권한이 있는 사용자의 암호
3	DatabaseUrl	문자열	데이터베이스에 대한 연결을 사용 가능하게 하는 URL. 예: jdbc:oracle:thin:@MACHINENAME:1521:SIDNAME
4	DatabaseDriver	문자열	연결 설정에 사용되는 드라이버의 이름. 예: oracle.jdbc.driver.OracleDriver
5	DefaultBOPrefix	문자열	Business Object 이름을 고유하게 만들기 위해 추가되는 텍스트. 필요한 경우 나중에 Business Object Designer가 Business Object 등록 정보에 대해 프롬프트할 때 이를 변경할 수 있습니다. 자세한 정보는 84 페이지의 『추가 정보 제공』을 참조하십시오.
6	TraceFileName	문자열	ODA for Manugistics가 추적 정보를 기록하는 파일. 파일이 존재하지 않는 경우 ODA for Manugistics가 \ODA\Manugistics 디렉토리에 파일을 작성합니다. 파일이 이미 존재하는 경우 ODA for Manugistics가 이를 파일에 추가합니다. ODA for Manugistics는 이름 지정 규칙에 따라 파일 이름을 지정합니다. 예를 들어 에이전트의 이름이 ManugisticsODA로 지정되는 경우 이는 ManugisticsODAtrace.txt로 이름 지정된 추적 파일을 생성합니다. 이 등록 정보를 사용하여 이 파일에 다른 이름을 지정하십시오.
7	TraceLevel	정수	ODA for Manugistics에 사용 가능한 추적 레벨

표 14. ODA for Manugistics 등록 정보 (계속)

행 번호	등록 정보 이름	등록 정보 유형	설명
8	MessageFile	문자열	오류 및 메시지 파일의 이름. ODA for Manugistics는 이름 지정 규칙에 따라 파일 이름을 표시합니다. 예를 들어 에이전트의 이름이 ManugisticsODA로 지정되는 경우 메시지 파일 등록 정보의 값이 ManugisticsODAAgent.txt로 표시됩니다. **중요**: 오류 및 메시지 파일은 \ODA\messages 디렉토리에 위치해야 합니다. 이 등록 정보를 사용하여 기존 파일을 확인하거나 지정하십시오.

중요 사항

Business Object Designer에 표시된 기본값이 존재하지 않는 파일을 나타내는 경우 메시지 파일의 이름을 정정하십시오. 이 대화 상자에서 다음으로 이동할 때 이름이 올바르지 않으면 Business Object Designer가 ODA를 설치한 창에 오류 메시지를 표시합니다. 이 메시지는 Business Object Designer에서 팝업되지 않습니다. 올바른 메시지 파일 지정에 실패하면 ODA가 메시지 없이 실행하게 됩니다.

노드 펼치기와 테이블, 보기 및 저장 프로시저 선택

ODA for Manugistics의 모든 초기화 등록 정보를 구성하고 나면 Business Object Designer가 지정된 데이터베이스에 연결되어 데이터베이스 내 모든 스키마 이름을 갖는 트리를 표시합니다. 트리에서 노드로 나타나는 이 이름들은 펼칠 수 있습니다. 이름을 눌러 각 스키마에 모든 테이블, 보기 및 저장 프로시저를 표시하십시오. 그림 8은 일부 스키마가 펼쳐진 이 대화 상자를 보여 줍니다.

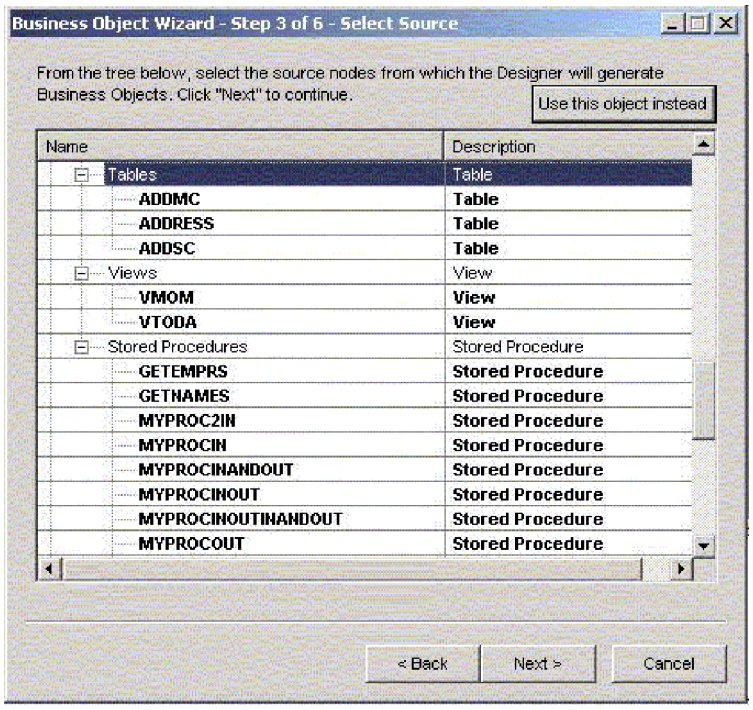


그림 8. 노드가 펼쳐진 스키마의 트리

생성된 Business Object 정의의 데이터를 저장하는 모든 데이터베이스 오브젝트를 식별하려면 모든 필수 테이블, 보기 및 저장 프로시저를 선택한 후 다음을 누르십시오. 리턴된 오브젝트를 필터하는 방법에 대한 정보는 *Business Object Development Guide.* 를 참조하십시오.

데이터베이스 오브젝트 선택사항 확인

생성된 Business Object 정의와 연관될 모든 데이터베이스 오브젝트를 식별한 후 Business Object Designer는 선택된 테이블과 보기만이 있는 대화 상자를 표시합니다. 그림 9는 이 대화 상자를 보여 줍니다.

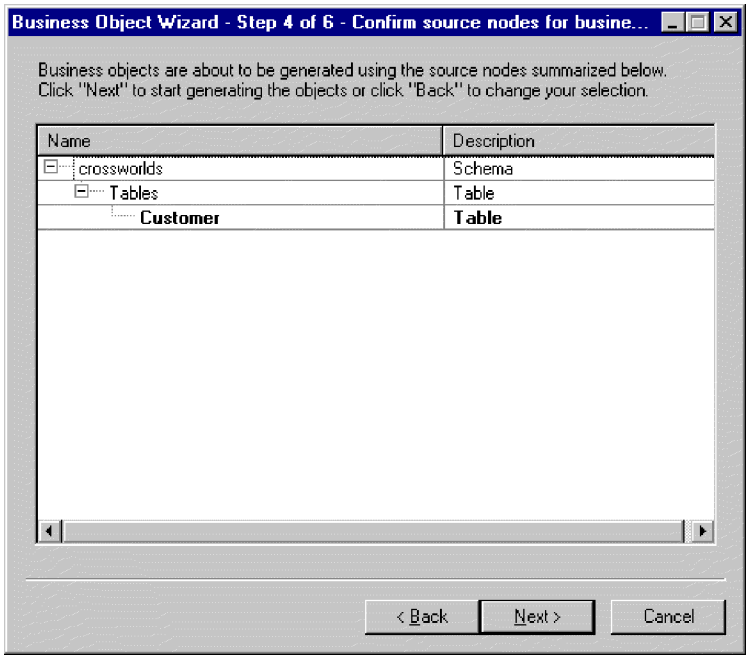


그림 9. 데이터베이스 오브젝트의 선택사항 확인

이 창은 다음 옵션을 제공합니다.

- 선택사항을 확인하려면 다음을 누르십시오.
- 선택사항이 올바르지 않은 경우 이전을 눌러 이전 창으로 리턴하고 필요한 변경을 수행하십시오. 선택사항이 올바르면 다음을 누르십시오.

정의 생성

데이터베이스 오브젝트를 확인한 후 다음 대화 상자에서 Business Object Designer가 정의를 생성 중임을 알 수 있습니다.

추가 정보 제공

ODA for Manugistics에 추가 정보가 필요한 경우 Business Object Designer가 사용자에게 정보를 프롬프트하는 BO 등록 정보 창을 표시합니다.

BO 등록 정보 창에서 다음 정보를 입력하거나 변경하십시오.

- *접두부* - Business Object 이름을 고유하게 만들기 위해 추가되는 텍스트. 에이전트 구성 창(그림 7)에서 *DefaultBOPrefix* 등록 정보에 대해 입력한 값에 만족하는 경우에는 여기서 값을 변경할 필요가 없습니다.
- *Verb* - *값* 필드를 누르고 팝업 메뉴에서 하나 이상의 Verb를 선택하십시오. 이 Verb는 Business Object에서 지원하는 Verb입니다.
- *저장 프로시저 추가* - 값 필드에서 예 또는 아니오를 누르십시오.

– 예를 선택하고 확인을 누르면 ODA for Manugistics가 모든 저장 프로시저 속성 목록을 제공하는 창을 표시합니다. Business Object에 추가하려는 저장 프로시저 속성을 선택하십시오.
– 아니오를 선택하여 저장 프로시저 속성이 생성된 Business Object 정의에 추가되지 않도록 하십시오.

기본값은 예입니다.

주: BO 등록 정보 대화 상자의 필드에 복수 값이 있는 경우 대화 상자가 처음 표시될 때 필드가 비어 있는 것처럼 보입니다. 필드에서 눌러 값의 드롭 다운 목록을 표시하십시오.

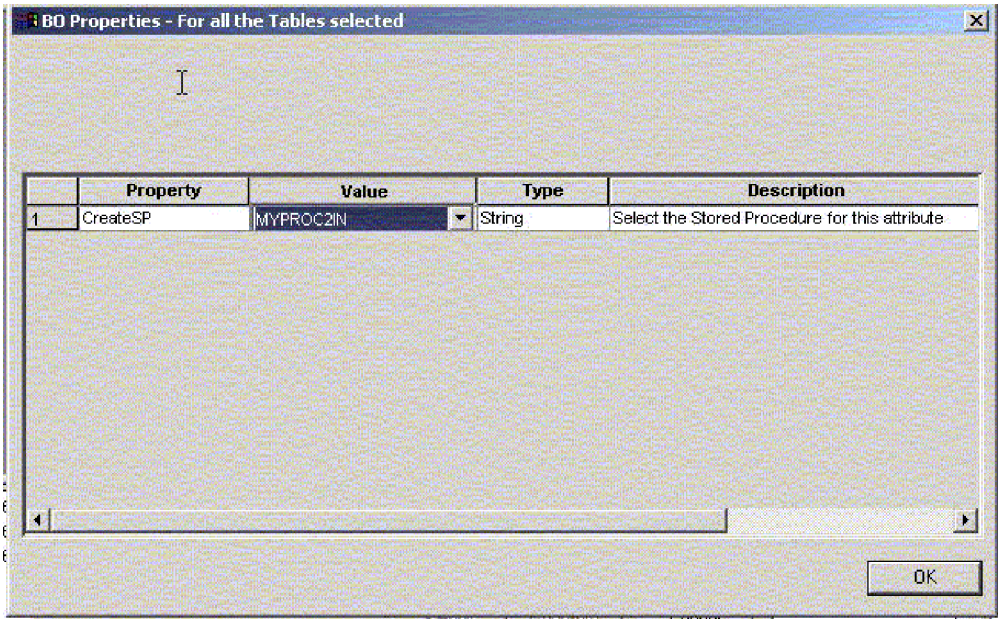


그림 10. 저장 프로시저와 저장 프로시저 속성의 연관

Business Object에 추가된 저장 프로시저 속성은 해당 스키마의 데이터베이스에 있는 저장 프로시저 중 하나와 연관시킬 수 있습니다. 각 저장 프로시저 속성에 대해 해당 스키마의 데이터베이스에 있는 모든 저장 프로시저의 드롭 다운 목록에서 저장 프로시저를 선택할 수 있습니다. 이 정보는 해당 속성에 대해 필요한 ASI 정보를 생성합니다.

오브젝트 레벨의 ASI(응용프로그램 특정 정보)는 `TN=tableName`과 같습니다.

속성 레벨의 ASI는 `CN=ColumnName`과 같습니다.

Business Object가 저장 프로시저에서 생성 중이고 SPForCreate와 같은 ODA for Manugistics Adapter 저장 프로시저 속성이 Business Object와 연관된 경우 저장 프로시저 속성에 대해 ODA는 해당 스키마의 모든 저장 프로시저 이름 목록을 제공하고

사용자는 필수 저장 프로시저를 Business Object에 연관시킬 수 있습니다. 이로써 ODA for Manugistics Adapter 저장 프로시저 속성에 대한 ASI가 다음과 같이 생성됩니다.

```
SPN=stored procedure Name; IN=a1:a2; OUT=b1:b2; IO=c1:c2
```

여기에서 IN은 저장 프로시저의 매개변수가 INPUT 유형, OUT은 OUTPUT 유형, IO는 INPUT/OUTPUT 유형임을 의미합니다. ODA는 ASI에서 RS를 true 또는 false로 설정하지 않기 때문에 사용자가 수동으로 설정해야 합니다.

Business Object에 추가된 Verb는 표준 Verb로서 기본적으로 Retrieve, RetrieveByContent, Create, Update 및 Delete입니다.

저장 프로시저의 리턴 매개변수 유형이 ResultSet인 경우 ODA가 결과 세트를 분석하고 Business Object를 작성하여 Business Object의 결과 세트 속성 열을 만듭니다. 저장 프로시저 열의 ASI는 CN=StoredProcedureColumnName로 설정됩니다. ODA는 드라이버가 리턴하는 JDBC 메타 데이터 정보에 따라 키 속성을 설정합니다. 리턴되는 정보가 없는 경우 기본적으로 ODA는 속성을 키로 표시하지 않습니다. 길이 및 유형과 같은 기타 모든 속성은 테이블에서 생성된 속성으로 설정됩니다.

정의 저장

BO 등록 정보 대화 상자에 모든 필수 정보를 입력하고 확인을 누르고 나면 Business Object Designer가 마법사에 마지막 대화 상자를 표시합니다. 여기서 정의를 서버 또는 파일에 저장하거나 Business Object Designer에서 편집하기 위해 정의를 열 수 있습니다. 자세한 정보 및 추가 수정사항은 *Business Object Development Guide*를 참조하십시오.

생성된 정의 내용

ODA for Manugistics가 생성하는 Business Object 정의는 다음을 포함합니다.

- 지정된 데이터베이스 테이블 및 보기의 각 열에 대한 속성
- BO 등록 정보 창에 지정된 Verb
- 응용프로그램 특정 정보:
  - Business Object 레벨에서
  - 각 속성에 대해
  - 각 Verb에 대해

이 섹션은 다음에 대해 설명합니다.

- 87 페이지의 『Business Object 레벨 등록 정보』
- 87 페이지의 『Attribute 등록 정보』
- 89 페이지의 『Verb』

Business Object 레벨 등록 정보

ODA for Manugistics는 Business Object 레벨에서 다음 정보를 생성합니다.

- Business Object의 이름
- 버전 - 기본값은 1.0.0
- 응용프로그램 특정 정보

Business Object 레벨에서 응용프로그램 특정 정보를 사용하면 다음과 같은 기능을 수행할 수 있습니다.

- 해당 데이터베이스 테이블의 이름 지정
- 실제 또는 논리 삭제 수행에 필요한 정보 제공

Business Object 레벨에서 응용프로그램 특정 정보 형식은 세미콜론(;) 분리문자로 구분되는 매개변수로 구성됩니다. 매개변수 이름 및 해당 값은 콜론(:) 분리문자로 분리됩니다. 구문은 다음과 같습니다.

TN=*TableName*; SCN=*StatusColumnName*:*StatusValue*

여기서 TableName은 데이터베이스 테이블을 식별하고 *StatusColumnName*은 논리 삭제를 수행하기 위해 사용되는 데이터베이스 열의 이름이며 *StatusValue*는 Business Object가 비활성화되거나 삭제됨을 의미하는 값입니다.

이 레벨에서 ODA for Manugistics가 생성하는 AppSpecificInfo는 데이터베이스 테이블이나 보기의 이름에 대한 값만을 포함합니다. 상태 열에 값을 지정하는 것에 대한 정보는 64 페이지의 『Business Object 레벨의 응용프로그램 특정 정보』를 참조하십시오.

Attribute 등록 정보

이 섹션에서는 각 속성에 대해 ODA for Manugistics가 생성하는 등록 정보에 대해 설명합니다. 속성에 대한 자세한 정보는 60 페이지의 『Business Object 속성 등록 정보』를 참조하십시오.

Name 등록 정보

ODA for Manugistics는 데이터베이스 테이블 또는 보기에 있는 열 이름에서 속성 이름 값을 확보합니다.

Data type 등록 정보

속성의 유형을 설정하는 경우 ODA for Manugistics가 테이블 또는 보기에 있는 열의 데이터 유형을 해당 IBM WebSphere Business Integration Adapter Business Object 유형으로 변환합니다. 이 변환은 두 단계로 완료됩니다. 먼저 데이터베이스에 있는 데이터 유형이 JDBC 유형으로 변환됩니다. 그런 다음 JDBC 유형은 IBM WebSphere Business Integration Adapter Business Object 유형으로 변환됩니다. 처음 변환은 사용 중인 JDBC 드라이버로 수행됩니다. JDBC 유형에 맵핑되는 개별 데이터베이스 유

형에 대한 세부사항은 JDBC 스펙(2.0 이상)을 참조하십시오. 표 14는 JDBC 유형이 해당 IBM WebSphere Business Integration Adapter Business Object 유형으로 변환되는 것을 보여줍니다.

표 15. 데이터 유형의 일치

JDBC 유형	WebSphere Business Integration Adapter Business Object 유형
BIT	BOOLEAN
CHAR	STRING
VARCHAR	STRING
LONGVARCHAR	STRING
INTEGER	INTEGER
NUMERIC	INTEGER
SMALLINT	INTEGER
TINYINT	INTEGER
BIGINT	INTEGER
DATE	DATE
TIME	DATE
TIMESTAMP	DATE
DECIMAL	STRING
DOUBLE	DOUBLE
FLOAT	DOUBLE
REAL	FLOAT
BINARY	STRING, 다음 BYTEARRAY=TRUE를 AppSpe ificInfo에 추가
VARBINARY	STRING, 다음 BYTEARRAY=TRUE를 AppSpe ificInfo에 추가

주: 열의 데이터 유형이 표 15에 표시된 유형이 아닌 경우 ODA for Manugistics는 열을 건너뛰고 열을 처리할 수 없다는 내용의 메시지를 표시합니다.

Cardinality 등록 정보

ODA for Manugistics는 모든 단순 속성의 카디낼리티를 1로 설정합니다.

MaxLength 등록 정보

ODA for Manugistics는 문자열의 길이를 varchar, char 또는 text 데이터 유형에 지정된 길이에서 확보합니다.

IsKey 등록 정보

열이 테이블의 1차 키인 경우 ODA for Manugistics는 해당 열을 키 속성으로 표시합니다. 그러나 테이블 대신 보기가 소스 노드로서 선택되어서 Business Object를 생성하면 ODA for Manugistics가 열을 키 속성으로 표시하지 않습니다. 이 경우 키 속성은 수동으로 설정되어야 합니다.

IsForeignKey 등록 정보

ODA for Manugistics는 `IsForeignKey` 등록 정보를 설정하지 않습니다. Business Object Designer에서 설정할 수 있습니다.

IsRequired 등록 정보

필드가 테이블이나 보기에서 널이 아님으로 지정되는 경우 ODA for Manugistics가 이를 특정 필수 속성으로 표시합니다. 그러나 연관 순서가 있거나 ID 열일 수 있으므로 ODA for Manugistics는 키 필드를 필수로 표시하지 않습니다.

AppSpecificInfo 등록 정보

ODA for Manugistics는 속성 레벨에서 `AppSpecificInfo` 등록 정보에 대해 두 개의 매개변수를 포함합니다. 지정된 매개변수의 구문은 다음과 같습니다.

`CN=`*`ColumnName`*

여기서 `ColumnName`은 특정 속성과 연관된 데이터베이스 테이블 또는 보기에 있는 열 이름입니다.

`BYTEARRAY=`*`true`*`|`*`false`*

ODA for Manugistics는 2진 데이터를 가진 열을 인식하며 `BYTEARRAY=true`의 `AppSpecificInfo` 등록 정보로 유형 문자열의 속성을 작성합니다.

주: Business Object Designer에서 추가 `AppSpecificInfo` 매개변수를 설정할 수 있습니다. 이 매개변수에 대한 정보는 65 페이지의 『속성 레벨의 응용프로그램 특정 정보』를 참조하십시오.

Verb

ODA for Manugistics는 BO 등록 정보 창에 지정된 Verb를 생성합니다. ODA for Manugistics는 각 Verb에 `AppSpecificInfo` 등록 정보를 작성하지만 채우지는 않습니다. 자세한 정보는 72 페이지의 『Verb에 대한 응용프로그램 특정 정보 형식』을 참조하십시오.

샘플 Business Object 정의 파일

샘플 Business Object 정의는 다음과 같습니다.

```
[BusinessObjectDefinition]
Name = CUSTOMER
Version = 1.0.0
AppSpecificInfo = TN=ra_customers;SCN=

            [Attribute]
               Name = customer_id
               Type = Integer
               Cardinality = 1
               MaxLength = 0
               IsKey = true
               IsForeignKey = false
               IsRequired = false
               AppSpecificInfo = CN=customer_id
               DefaultValue =
               [End]

               *******Other attributes *********

[Attribute]
               Name = ObjectEventId
               Type = String
               Cardinality = 1
               MaxLength = 0
               IsKey = false
               IsForeignKey = false
               IsRequired = false
               AppSpecificInfo =
               DefaultValue =
               [End]

               [Verb]
               Name = Delete
               AppSpecificInfo =
               [End]

               [Verb]
               Name = Update
               AppSpecificInfo =
               [End]

               [Verb]
               Name = Create
               AppSpecificInfo =
               [End]

               [Verb]
               Name = Retrieve
               AppSpecificInfo =
               [End]
[End]
```

하위 Busines Object를 포함하는 속성 삽입

Business Object Designer를 사용하여 단일 카디낼리티 또는 다중 카디낼리티 하위 Business Object를 나타내는 속성을 삽입하십시오. 자세한 정보는 *Business Object Development Guide*를 참조하십시오.

Business Object 정의에 정보 추가

데이터베이스 테이블 또는 보기가 Business Object 정의가 필요로 하는 모든 정보를 갖고 있지 않을 수 있으므로 ODA for Manugistics가 작성하는 Business Object 정의에 정보를 추가해야 합니다. 자세한 정보는 37 페이지의 제 3 장 『커넥터의 Business Object 이해』를 참조하십시오.

Business Object 정의를 검사하거나 정보를 추가할 때 Business Object Designer나 텍스트 편집기를 사용할 수 있습니다. 수정된 정의를 IBM WebSphere Business Integration Adapter 저장소에 재로드하기 위해 Business Object Designer를 사용하거나 ICS가 통합 브로커인 경우 `repos_copy` 명령을 사용할 수 있습니다.

제 5 장 문제점 해결 및 오류 처리

이 장에서는 Connector for JDBC를 시작하거나 실행할 때 발생할 수 있는 문제점에 대해 설명합니다. 이 섹션에는 다음이 포함되어 있습니다.

- 『시작 문제점』
- 『이벤트 처리』
- 『맵핑(ICS 통합 브로커 전용)』
- 95 페이지의『오류 처리 및 로깅』
- 96 페이지의『응용프로그램에 연결이 끊어짐』
- 97 페이지의『자원 사용 중 오류』
- 98 페이지의『지원되지 않는 JDBC 드라이버로 인한 ODA for Manugistics의 부적절한 작동』

시작 문제점

커넥터를 시작할 때 문제점이 생기는 경우 통합 브로커가 실행 중인지 확인하십시오.

이벤트 처리

이벤트 테이블에 이벤트가 있고 커넥터가 실행 중인 동안에도 처리되지 않는 경우 다음을 확인하십시오.

- 관련 비즈니스 프로세스가 실행 중입니다.
- 이벤트 테이블에 있는 Business Object의 이름이 비즈니스 프로세스 포트에 지정된 Business Object의 이름과 일치합니다.

맵핑(ICS 통합 브로커 전용)

이 섹션은 다음에 대해 논의합니다.

- 『맵핑 문제점』
- 94 페이지의『날짜 변환』

맵핑 문제점

Business Object가 맵핑되고 있지 않고 맵핑이 호출되고 있지 않은 경우 맵이 올바른 디렉토리에 설치되었는지 확인하십시오.

날짜 변환

주: 이 날짜 변환 프로시저는 버전 1.5.0 이전의 커넥터 버전에만 적용됩니다.

맵을 사용하여 데이터베이스에 날짜 형식으로 저장된 데이터를 WebSphere Business Integration Adapter Business Object가 사용하는 문자열 형식으로 변환하십시오.

예를 들어 Oracle 데이터베이스에 저장되는 다음 데이터를

```
Sun Jan 01 00:00:00 CEST 1999
```

JDBC Business Object용 WebSphere Business Integration Adapter에서 처리되는 다음 문자열로 변환한다고 가정하십시오.

```
Jan 01 1999 00:00:00
```

이 변환을 수행하려면 맵핑 시에 데이터 변환에 정의된 `DtpDate()` 및 `DtpSplitString()` 구성자를 사용하십시오. 오브젝트를 구성하는 이들 구성자와 클래스의 구문 및 설명은 *Map Development Guide*를 참조하십시오.

날짜 값을 문자열로 변환할 맵을 사용하려면 다음 단계를 따르십시오.

1. 공백 분리문자와 함께 `DtpSplitString()`을 사용하여 문자열을 여섯 조각으로 분할하고 이를 `DtpDate`가 사용할 수 있는 순서로 재배열하십시오. 예의 날짜를 변환하려면 다음을 사용하십시오.

   ```
   DtpSplitString OurSplitString = new DtpSplitString
      ("Sun Jan 01 00:00:00 CEST 1999"," ");
   ```

   위의 명령문에서 `OurSplitString`은 `DtpSplitString` 유형의 사용자 정의 변수이고 공백이 분리문자로 지정됩니다.

2. `DtpSplitString` 클래스의 `nextElement()` 메소드를 사용하여 각 변수의 여섯 요소를 요소가 문자열 유형인 배열로 정리해서 새로 작성된 `OurSplitString` 변수를 통해 루프하십시오. 다음 예는 `OurStringPieces`를 출력 배열로 지정합니다.

   ```
   String[] OurStringPieces = new String[6];
   for (i=0;i<=5;i=i+1){
      OurStringPieces[i]=OurSplitString.nextElement();
   }
   ```

   이 루핑은 다음 배열 요소를 생성합니다.

   ```
   OurStringPieces[0] = Sun
   OurStringPieces[1] = Jan
   OurStringPieces[2] = 01
   OurStringPieces[3] = 00:00:00
   OurStringPieces[4] = CEST
   OurStringPieces[5] = 1999
   ```

3. DtpDate 입력에 필요한 문자열의 조각을 연결하십시오. 예의 변환은 DtpDate에 대한 입력 형식으로 "Jan 01 1999 00:00:00"과 같은 변환된 문자열을 필요로 하는 "M D Y h:m:s"를 사용합니다. 이 예의 문자열은 OurStringPieces 배열의 요소 1, 2, 5 및 3을 사용합니다.

```
OurConcatenatedString =
OurStringPieces[1]+OurStringPieces[2]+OurStringPieces[5]+OurStringPieces[3];
```

4. 새로 연결된 문자열을 다음과 같이 DtpDate 입력으로 사용하십시오.

```
DtpDate OurDtpDate = new DtpDate(OurConcatenatedString,"M D Y h:m:s");
```

날짜 값을 DtpDate 형식으로 지정하면 맵에서 날짜에 대해 작업할 준비가 된 것입니다.

오류 처리 및 로깅

커넥터는 Business Object 및 Verb의 현재 처리가 실패하도록 하는 조건에 처할 때마다 오류 메시지를 로그합니다. 그러한 오류가 발생하는 경우 커넥터는 실패한 Business Object가 수신될 때 이것의 텍스트 표시를 인쇄하기도 합니다. 커넥터는 해당 구성에 따라 커넥터 로그 파일 또는 표준 출력 스트림에 텍스트를 작성합니다. 오류 소스 판별에 도움이 되도록 텍스트를 사용할 수 있습니다.

오류 유형

표 16에서는 커넥터가 각 추적 레벨에서 출력하는 추적 메시지의 유형에 대해 설명합니다. 이 메시지는 추적 메시지 출력 외에 IBM WebSphere Business Integration Adapter 구조(예: Java 커넥터 실행 랩퍼 및 WebSphere MQ 메시지 인터페이스)에 의해 부가되는 것입니다.

표 16. 커넥터 추적 메시지

추적 레벨	추적 메시지
레벨 0	케넥터 버전을 식별하는 메시지. 이 레벨에서는 그 외의 기타 추적이 수행되지 않습니다. 이는 기본값입니다.
레벨 1	• 상태 메시지 • 처리된 각 Business Object에 대한 식별 (키) 정보를 제공하는 메시지 • pollForEvents 메소드가 실행될 때마다 전달된 메시지
레벨 2	• Business Object 처리 시 커넥터에 발생 또는 검색하는 배열 및 하위 Business Object와 같은 정보가 들어 있는 Business Object 핸들러 메시지 • gotApplEvent() 또는 executeCollaboration()에서 Business Object가 통합 브로커에 게시될 때마다 로그된 메시지 • Business Object가 통합 브로커 요청으로 수신되었음을 표시하는 메시지

표 16. 커넥터 추적 메시지 (계속)

추적 레벨	추적 메시지
레벨 3	• 커넥터가 Business Object에서 외부 키를 찾았거나 설정한 시기와 같은 정보를 포함하는 외부 키 처리 메시지 • Business Object 처리에 대한 정보를 제공하는 메시지. 예를 들어 이 메시지는 커넥터가 Business Object 간의 일치를 발견하거나 또는 하위 Business Object 배열에서 Business Object를 발견할 때 전달됩니다.
레벨 4	• 응용프로그램 특정 정보 메시지(예: Business Object의 응용프로그램 특정 정보 필드 구문을 분석하는 함수에 의해 리턴되는 값을 표시하는 메시지) • 커넥터가 함수를 입력 또는 종료할 때 커넥터 프로세스 플로우 추적을 도와주는 요소를 식별하는 메시지 • 모든 스레드 고유 메시지. 커넥터에 복수 스레드가 있는 경우 각 새 스레드 작성에 대해 메시지가 나타납니다.
레벨 5	• 커넥터 초기화를 표시하는 메시지(예: 통합 브로커에서 검색된 각 구성 등록 정보 값을 표시하는 메시지) • 응용프로그램에서 실행된 명령문을 포함하는 메시지. 이 추적 레벨에서 커넥터 로그 파일은 목적지 응용프로그램에서 실행된 모든 명령문과 대체되는 모든 변수의 값을 포함합니다. • 커넥터가 처리를 시작하기 전과(커넥터가 수신할 때 상태를 표시) 커넥터가 처리를 완료한 후(커넥터가 리턴할 때 상태를 표시) Business Object의 표시를 작성하는 메시지 • Business Object 덤프를 작성하는 메시지 • 커넥터 실행 시 작성하는 각 하위 스레드 상태를 표시하는 메시지

오류 메시지

커넥터 메시지 파일

커넥터가 생성하는 모든 오류 메시지는 JDBCConnector.txt 또는 JDBCConnector_*II_TT*.txt라고 이름 지정된 페이지 파일에 저장됩니다(여기서 *ll*은 언어를 지정하며 *TT*는 국가나 지역을 지정합니다). 각 오류에는 오류 메시지 앞에 오는 오류 번호가 있습니다. 예를 들어 다음과 같습니다.

```
20017
Connector Infrastructure version does not match.

20018
Connection from {1} to the Application is lost! Please enter 'q'
to stop the connector, then restart it after the problem is fixed.

20019
Error: ev_id is NULL in pollForEvent().
```

응용프로그램에 연결이 끊어짐

커넥터가 연결 설정에 실패하는 경우 이는 통합 브로커에 FAIL을 전송하고 종료됩니다.

AutoCommit가 false로 설정되고 PingQuery가 실패하면 커넥터가 데이터베이스로 새로 연결을 작성하려고 시도합니다. 데이터베이스에 새 연결을 작성하는 데 성공하면 처리를 계속하며 그렇지 않으면 커넥터는 APPRESPONSETIMEOUT을 리턴하며 결과적으로 커넥터가 종료합니다.

순서가 잘못된 페치 오류

Sun Solaris에서 Oracle 데이터베이스 8.0과 8.1을 사용하거나 Windows 2000에서 Oracle 8.1을 사용할 때 AutoCommit 등록 정보는 false로 설정되어야 합니다. 그렇지 않으면 ORA-01002(순서가 잘못된 페치) 오류 메시지를 만나게 됩니다. 이전 버전의 Oracle 데이터베이스에서는 이 오류가 발생하지 않습니다. AutoCommit가 false로 설정되면 성능이 향상됩니다.

자원 사용 중 오류

주: 이 커넥터가 Oracle 데이터베이스에서 실행 중일 때에만 이 오류가 발생합니다.

커넥터는 응용프로그램에서 데이터를 검색하거나 변경할 때 종종 다음과 같은 오류를 발견할 수 있습니다.

```
[Time: 2001/05/29 16:30:07.356] [System: ConnectorAgent] [SS: SOVTConnector]
[Type: Trace] [Mesg: Select CLIENT,COUNTRY,STRT_CODE,CITY_CODE,CITYP_CODE,
STRTYPEAB,COMMU_CODE,REGIOGROUP,TAXJURCODE from ADRSTREET where CLIENT='100'
and COUNTRY='DE' and STRT_CODE='000001114136' FOR UPDATE NOWAIT]
[Time: 2001/05/29 16:30:07.526] [System: ConnectorAgent] [SS: SOVTConnector]
[Type: Trace ] [Mesg: :logMsg]
[Time: 2001/05/29 16:30:07.536] [System: ConnectorAgent] [SS: SOVTConnector]
 [Type: Error ] [MsgID: 37002]
[Mesg: Execution of Retrieve statement failed : java.
sql.SQLException: ORA-00054: Versuch, mit NOWAIT eine bereits
belegte Ressourceanzufordern.]
```

이 오류는 커넥터가 현재 잠겨있는 레코드를 갱신하려고 할 때 발생합니다. 다른 프로세스에 의해 레코드가 잠기거나 커넥터가 다중 스레드여서 커넥터 스스로 레코드를 잠궜을 수 있습니다.

갱신 프로세스 중에 레코드는 잠금 상태여야 합니다. 커넥터는 통합 브로커가 수신한 오브젝트의 사후 이미지를 검색하려고 시도하며 프로세스 중에 데이터베이스의 전체 오브젝트 잠금을 수행하여 데이터 무결성을 유지합니다.

이 문제점을 해결하기 위해서는 커넥터가 레코드의 잠금을 확보하지 못하게 하는 프로세스를 중지하거나 커넥터의 RetryCountInterval 구성 등록 정보를 조정할 수 있습니다.

지원되지 않는 JDBC 드라이버로 인한 ODA for Manugistics의 부적절한 작동

JDBC 드라이버가 ODA for Manugistics의 기능을 지원하지 않는 경우 Object Discovery Agent가 적절하게 기능하지 않습니다. 예를 들어 ODA for Manugistics가 사용하는 모든 메소드 호출을 드라이버가 지원하지 않는 경우 ODA 로그가 실패한 프로세스를 표시합니다. 다음은 로그의 예입니다.

```
[Time: 2002/05/15 17:00:55.147] [System: Object Discovery Agent] [SS: null]
[Type: 6] [Mesg: A SQL Error occurred in getting Schema Names from Database.
Reason [ProductName][ODBC ProductName Driver]Optional feature not
implemented]
```

이런 경우 다른 JDBC 드라이버를 사용해야 합니다.

IGP 사용시 오류 처리

인터페이스 테이블에 설정된 저장 프로시저에서 Insert/Update 조작을 수행할 때 해당 오류 테이블에서 오류를 확인할 수 있습니다. 인터페이스 테이블을 통한 삽입 및 갱신 시에 실패에 대한 직접 공고는 가능하지 않습니다.

Integration Engineer는 관련 오류(ERR) 테이블을 확인하여 이 유형의 조작 결과를 파악할 수 있도록 협업 또는 워크플로우를 작성해야 합니다. 또한 ERR 테이블에서 이벤트 공고를 사용 가능하게 하여 이러한 오류 발생에 주의할 수 있습니다.

부록 A. 커넥터의 표준 구성 등록 정보

이 부록에서는 WebSphere Business Integration Adapter의 커넥터 구성요소에 대한 표준 구성 등록 정보에 대해합니다. 다음 통합 브로커에서 실행되는 커넥터에 대한 정보를 다룹니다.

- WebSphere InterChange Server(ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker 및 WebSphere Business Integration 메시지 브로커를 통털어 WMQI(WebSphere 메시지 브로커)라고 합니다.
- WebSphere Application Server(WAS)

모든 커넥터가 이들 표준 등록 정보를 전부 사용하는 것은 아닙니다. Connector Configurator에서 통합 브로커를 선택할 때, 이 브로커에서 실행되는 어댑터에 구성해야 하는 표준 등록 정보의 목록이 표시됩니다.

커넥터 특정 등록 정보에 대한 정보는 관련 어댑터 사용자 안내서를 참조하십시오.

주: 이 책에서 백슬래시(\)는 디렉토리 경로의 규칙으로 사용됩니다. UNIX 설치의 경우, 백슬래시를 슬래시(/)로 대체하고 각 운영 체제의 규칙을 따르십시오.

신규 및 삭제된 등록 정보

다음 표준 등록 정보가 이 릴리스에 추가되었습니다.

새 등록 정보

- XMLNameSpaceFormat

삭제된 등록 정보

- RestartCount
- RHF2MessageDomain

표준 커넥터 등록 정보 구성

어댑터 커넥터에는 두 가지 유형의 구성 등록 정보가 있습니다.

- 표준 구성 등록 정보
- 커넥터 특정 구성 등록 정보

이 절에서는 표준 구성 등록 정보에 대해 설명합니다. 커넥터 특정 구성 등록 정보에 대한 정보는 해당 어댑터 사용자 안내서를 참조하십시오.

Connector Configurator 사용

System Manager에서 액세스하는 Connector Configurator에서 커넥터 등록 정보를 구성합니다. Connector Configurator 사용에 대한 자세한 정보는 Connector Configurator 부록을 참조하십시오.

주: Connector Configurator와 System Manager는 Windows 시스템에서만 실행됩니다. UNIX 시스템에서 커넥터를 실행 중인 경우, Windows 시스템에 이러한 도구가 설치되어 있어야 합니다. UNIX에서 실행하는 커넥터에 커넥터 등록 정보를 설정하려면, Windows 시스템에서 System Manager를 시작하고, UNIX 통합 브로커에 연결한 후 커넥터용 Connector Configurator를 가져와야 합니다.

등록 정보 값 설정 및 갱신

등록 정보 필드의 기본 길이는 255자입니다.

커넥터는 다음 순서를 사용하여 등록 정보의 값을 판별합니다(가장 높은 숫자가 다른 값을 대체함).

1. 기본값
2. 저장소(WebSphere InterChange Server가 통합 브로커일 경우에만)
3. 로컬 구성 파일
4. 명령행

커넥터는 시작할 때 구성값을 확보합니다. 런타임 세션 중에 하나 이상의 커넥터 등록 정보 값을 변경하면, 등록 정보의 **갱신 메소드**가 변경사항의 적용 방법을 판별합니다. 표준 커넥터 등록 정보에 대한 갱신 메소드에는 네 가지가 있습니다.

- **동적**
  System Manager에 저장된 후 변경사항이 즉시 적용됩니다. 커넥터가 WebSphere 메시지 브로커와 같은 독립형 모드에서 작동 중이면(System Manager와 무관하게), 구성 파일을 통해서만 등록 정보를 변경할 수 있습니다. 이 경우에는 동적 갱신이 가능하지 않습니다.
- 구성요소 다시 시작
  커넥터가 중지된 다음 System Manager에서 다시 시작된 후에만 변경사항이 적용됩니다. 응용프로그램 특정 구성요소나 통합 브로커를 중지하고 다시 시작할 필요는 없습니다.
- 서버 다시 시작
  응용프로그램 특정 구성요소와 통합 브로커를 중지하고 다시 시작한 후에만 변경사항이 적용됩니다.
- 에이전트 다시 시작(ICS에만 해당)
  응용프로그램 특정 구성요소를 중지하고 다시 시작한 후에만 변경사항이 적용됩니다.

특정 등록 정보의 갱신 방법을 결정하려면, Connector Configurator 창의 **갱신 메소드** 열이나 아래 등록 정보 요약 테이블의 갱신 메소드 열을 참조하십시오.

표준 등록 정보 요약

표 17은 표준 커넥터 구성 등록 정보에 대한 빠른 참조를 제공합니다. 모든 커넥터가 다음 등록 정보를 모두 사용하는 것은 아니며 표준 등록 정보 종속성은 RepositoryDirectory를 기본으로 하기 때문에 등록 정보 설정은 통합 브로커에 따라 달라질 수 있습니다.

커넥터를 실행하기 전에 이러한 등록 정보 중 일부의 값을 설정해야 합니다. 각 등록 정보에 대한 설명을 보려면 다음 절을 참조하십시오.

표 17. 표준 구성 등록 정보 요약

등록 정보 이름	가능한 값	기본값	갱신 메소드	참고
AdminInQueue	올바른 JMS 대기열 이름	*CONNECTORNAME* /ADMININQUEUE	구성요소 다시 시작	전달 전송이 JMS임
AdminOutQueue	올바른 JMS 대기열 이름	*CONNECTORNAME*/ADMINOUTQUEUE	구성요소 다시 시작	전달 전송이 JMS임
AgentConnections	1 - 4	1	구성요소 다시 시작	전달 전송이 MQ 또는 IDL인 경우: 저장소 디렉토리가 <REMOTE>임
AgentTraceLevel	0 - 5	0	동적	
ApplicationName	응용프로그램 이름	커넥터 응용프로그램 이름에 대해 지정된 값	구성요소 다시 시작	
BrokerType	ICS, WMQI, WAS			
CharacterEncoding	ascii7, ascii8, SJIS, Cp949, GBK, Big5, Cp297, Cp273, Cp280, Cp284, Cp037, Cp437 **주:** 이것은 지원되는 값의 서브세트입니다.	ascii7	구성요소 다시 시작	
ConcurrentEventTriggeredFlows	1 - 32,767	1	구성요소 다시 시작	저장소 디렉토리가 <REMOTE>임
ContainerManagedEvents	값이 없음 또는 JMS	값이 없음	구성요소 다시 시작	전달 전송이 JMS임
ControllerStoreAndForwardMode	true 또는 false	True	동적	저장소 디렉토리가 <REMOTE>임
ControllerTraceLevel	0 - 5	0	동적	저장소 디렉토리가 <REMOTE>임
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	구성요소 다시 시작	JMS 전송만
DeliveryTransport	MQ, IDL 또는 JMS	JMS	구성요소 다시 시작	저장소 디렉토리가 로컬이면 값은 JMS뿐임

표 17. 표준 구성 등록 정보 요약 (계속)

등록 정보 이름	가능한 값	기본값	갱신 메소드	참고
DuplicateEventElimination	True 또는 False	False	구성요소 다시 시작	JMS 전송만: Container Managed Events가 <NONE>이어야 함
FaultQueue		*CONNECTORNAME*/FAULTQUEUE	구성요소 다시 시작	JMS 전송만
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory 또는 CxCommon.Messaging.jms.SonicMQFactory 또는 Java 클래스 이름	CxCommon.Messaging.jms.IBMMQSeriesFactory	구성요소 다시 시작	JMS 전송만
jms.MessageBrokerName	actoryClassName이 IBM일 경우, crossworlds.queue.manager를 사용하십시오. FactoryClassName이 Sonic이면 localhost:2506을 사용하십시오.	crossworlds.queue.manager	구성요소 다시 시작	JMS 전송만
jms.NumConcurrentRequests	양의 정수	10	구성요소 다시 시작	JMS 전송만
jms.Password	올바른 암호		구성요소 다시 시작	JMS 전송만
jms.UserName	올바른 이름		구성요소 다시 시작	JMS 전송만
JvmMaxHeapSize	힙 크기(MB)	128M	구성요소 다시 시작	저장소 디렉토리가 <REMOTE>임
JvmMaxNativeStackSize	스택 크기(KB)	128K	구성요소 다시 시작	저장소 디렉토리가 <REMOTE>임
JvmMinHeapSize	힙 크기(MB)	1M	구성요소 다시 시작	저장소 디렉토리가 <REMOTE>임
ListenerConcurrency	1 - 100	1	구성요소 다시 시작	전달 전송이 MQ여야 함
Locale	en_US, ja_JP, ko_KR, zh_CN, zh_TW, fr_FR, de_DE, it_IT, es_ES, pt_BR **주:** 이것은 지원되는 로케일의 서브세트입니다.	en_US	구성요소 다시 시작	
LogAtInterchangeEnd	True 또는 False	False	구성요소 다시 시작	저장소 디렉토리가 <REMOTE>여야 함
MaxEventCapacity	1 - 2147483647	2147483647	동적	저장소 디렉토리가 <REMOTE>여야 함
MessageFileName	경로 또는 파일 이름	InterchangeSystem.txt	구성요소 다시 시작	
MonitorQueue	올바른 대기열 이름	*CONNECTORNAME*/MONITORQUEUE	구성요소 다시 시작	JMS 전송만: DuplicateEvent Elimination이 True이어야 함

표 17. 표준 구성 등록 정보 요약 (계속)

등록 정보 이름	가능한 값	기본값	갱신 메소드	참고
OADAutoRestartAgent	True 또는 False	False	동적	저장소 디렉토리가 <REMOTE>여야 함
OADMaxNumRetry	양수	1000	동적	저장소 디렉토리가 <REMOTE>여야 함
OADRetryTimeInterval	분 단위의 양수 값	10	동적	저장소 디렉토리가 <REMOTE>여야 함
PollEndTime	HH:MM	HH:MM	구성요소 다시 시작	
PollFrequency	밀리초 단위의 양수 no(폴링을 사용 불가능하게 할 경우), key(문자 p가 커넥터 명령 창에 입력될 때 폴하기 위해)	10000	동적	
PollQuantity	1 - 500	1	에이전트 다시 시작	JMS 전송만: 컨테이너이고 관리 이벤트가 지정되는 경우에만
PollStartTime	HH:MM(HH는 0 - 23, MM은 0 - 59)	HH:MM	구성요소 다시 시작	
RepositoryDirectory	메타 데이터 저장소의 위치		에이전트 다시 시작	ICS의 경우: <REMOTE> WebSphere MQ 메시지 브로커 및 WAS의 경우: C:\crossworlds\repository로 설정됨
RequestQueue	올바른 JMS 대기열 이름	*CONNECTORNAME*/REQUESTQUEUE	구성요소 다시 시작	전달 전송이 JMS임
ResponseQueue	올바른 JMS 대기열 이름	*CONNECTORNAME*/RESPONSEQUEUE	구성요소 다시 시작	전달 전송이 JMS: 저장소 디렉토리가 <REMOTE>인 경우에만 필수
RestartRetryCount	0 - 99	3	동적	
RestartRetryInterval	분별 가능한 값(분): 1 - 2147483547	1	동적	
SourceQueue	올바른 WebSphere MQ 이름	*CONNECTORNAME*/SOURCEQUEUE	에이전트 다시 시작	전달 전송이 JMS 및 컨테이너이고 관리 이벤트가 지정되는 경우에만
SynchronousRequestQueue		*CONNECTORNAME*/SYNCHRONOUSREQUESTQUEUE	구성요소 다시 시작	전달 전송이 JMS임
SynchronousRequestTimeout	0 - 임의의 수(밀리초)	0	구성요소 다시 시작	전달 전송이 JMS임

표 17. 표준 구성 등록 정보 요약 (계속)

등록 정보 이름	가능한 값	기본값	갱신 메소드	참고
SynchronousResponseQueue		*CONNECTORNAME*/SYNCHRONOUSRESPONSEQUEUE	구성요소 다시 시작	전달 전송이 JMS임
WireFormat	CwXML, CwBO	CwXML	에이전트 다시 시작	저장소 디렉토리가 <REMOTE>가 아닐 경우에는 CwXML: 저장소 디렉토리가 <REMOTE>인 경우에는 CwBO
WsifSynchronousRequest 제한시간	0 - 임의의 숫자 (밀리초)	0	구성요소 다시 시작	WAS만 해당됨
XMLNameSpaceFormat	short, long	short	에이전트 다시 시작	WebSphere MQ 메시지 브로커 및 WAS만 해당됨

표준 구성 등록 정보

이 절에서는 표준 커넥터 구성 등록 정보를 나열하고 각각을 정의합니다.

AdminInQueue

관리 메시지를 커넥터에 전송하기 위해 통합 브로커에서 사용하는 대기열입니다.

기본값은 CONNECTORNAME/ADMININQUEUE입니다.

AdminOutQueue

관리 메시지를 통합 브로커에 전송하기 위해 커넥터에서 사용하는 대기열입니다.

기본값은 CONNECTORNAME/ADMINOUTQUEUE입니다.

AgentConnections

RepositoryDirectory가 <REMOTE>인 경우에만 적용 가능합니다.

AgentConnections 등록 정보는 orb.init[]에 의해 열려 있는 ORB 연결 수를 제어합니다.

기본적으로, 이 등록 정보의 값은 1로 설정되어 있으며 이 기본값을 변경할 필요가 없습니다.

AgentTraceLevel

응용프로그램 특정 구성요소의 추적 메시지 레벨입니다. 기본값은 0입니다. 구성요소는 설정된 추적 레벨이나 그 이하의 레벨에서 적용 가능한 모든 추적 메시지를 전달합니다.

ApplicationName

커넥터의 응용프로그램을 고유하게 식별하는 이름입니다. System Manager는 이 이름을 사용하여 WebSphere Business Integration System 환경을 모니터합니다. 이 등록 정보는 사용자가 커넥터를 실행하기 전에 값을 가지고 있어야 합니다.

BrokerType

사용 중인 통합 브로커 유형을 식별합니다. 옵션에는 ICS, WebSphere 메시지 브로커(WMQI, WMQIB 또는 WBIMB) 또는 WAS가 있습니다.

CharacterEncoding

문자(예: 영문자, 숫자 표시 또는 구두점 표시)에서 숫자값으로 맵핑하는 데 사용되는 문자 코드 세트를 지정합니다.

주: Java 기반 커넥터에서는 이 등록 정보를 사용하지 않습니다. C++ 커넥터는 현재 이 등록 정보에 값 `ascii7`을 사용합니다.

기본적으로 지원되는 문자 인코딩의 서브세트만 드롭 목록에 표시됩니다. 기타 지원되는 값을 드롭 목록에 추가하려면, 제품 디렉토리에 있는 `\Data\Std\stdConnProps.xml` 파일을 직접 수정해야 합니다. 자세한 정보는 Connector Configurator의 부록을 참조하십시오.

ConcurrentEventTriggeredFlows

`RepositoryDirectory`가 <REMOTE>인 경우에만 적용 가능합니다.

이벤트 전달을 위해 커넥터에서 동시에 처리할 수 있는 Business Object 수를 판별합니다. 이 속성값을 동시에 맵핑하여 전달할 Business Object 수로 설정하십시오. 예를 들어, 5개의 Business Object가 동시에 처리되도록 하려면 이 등록 정보의 값을 5로 설정하십시오. 기본값은 1입니다.

이 등록 정보를 1보다 큰 값으로 설정하면, 소스 응용프로그램의 커넥터가 동시에 여러 이벤트 Business Object를 맵핑하고 이들을 여러 협업 인스턴스에 동시에 전달할 수 있습니다. 특히 Business Object가 복잡한 맵을 사용하는 경우, 통합 브로커로 Business Object를 전달하는 속도가 빨라집니다. 협업에 대한 Business Object 도착률이 증가하면 시스템에서 전체 성능이 향상될 수 있습니다.

전체 플로우에 대한 동시 처리를 구현하려면(소스 응용프로그램에서 목적지 응용프로그램으로) 다음을 수행해야 합니다.

- 동시 이벤트 최대 수 등록 정보를 다중 스레드를 사용할 만큼 충분히 설정하여, 다중 스레드를 사용하도록 협업을 구성하십시오.
- 목적지 응용프로그램의 응용프로그램 특정 구성요소가 요청을 동시에 처리할 수 있는지 확인하십시오. 즉, 이 구성요소가 다중 스레드 방식이거나 Connector Agent

Parallelism을 사용할 수 있고 다중 프로세스에 맞게 구성되어 있어야 합니다. Parallel Process Degree 구성 등록 정보를 1보다 큰 값으로 설정하십시오.

ConcurrentEventTriggeredFlows 등록 정보는 단일 스레드이고 순차적으로 수행되는 커넥터 폴링에 영향을 주지 않습니다.

ContainerManagedEvents

이 등록 정보는 JMS 이벤트 저장소가 있는 JMS 사용 커넥터에서 보증된 이벤트 전달을 제공하게 하며, 여기에서 이벤트가 소스 대기열에서 제거되고 단일 JMS 트랜잭션으로 목적지 대기열에 위치합니다.

기본값은 값 없음입니다.

ContainerManagedEvents가 JMS로 설정되면, 보증된 이벤트 전달이 가능하도록 다음 등록 정보를 구성해야 합니다.

- PollQuantity = 1 ~ 500
- SourceQueue = CONNECTORNAME/SOURCEQUEUE

또한 MimeType, DHClass 및 DataHandlerConfigMOName(선택적) 등록 정보로 데이터 핸들러를 구성해야 합니다. 이러한 값을 설정하려면 Connector Configurator의 **Data Handler** 탭을 사용하십시오. Data Handler 탭 아래의 값의 필드는 ContainerManagedEvents를 JMS로 설정한 경우에만 표시됩니다.

주: ContainerManagedEvents가 JMS로 설정되면, 커넥터는 pollForEvents() 메소드를 호출하지 않으므로 해당 메소드의 기능을 사용할 수 없습니다.

이 등록 정보는 DeliveryTransport 등록 정보가 JMS 값으로 설정되어 있는 경우에만 나타납니다.

ControllerStoreAndForwardMode

RepositoryDirectory가 <REMOTE>인 경우에만 적용 가능합니다.

목적지 응용프로그램 특정 구성요소가 사용 불가능함을 발견한 후 커넥터 제어기의 작동을 설정합니다.

이 등록 정보가 true로 설정되고 이벤트가 ICS에 도달할 때 목적지 응용프로그램 특정 구성요소가 사용 불가능한 경우, 커넥터 제어기는 응용프로그램 특정 구성요소에 대한 요청을 차단합니다. 응용프로그램 특정 구성요소가 작동하게 되면, 제어기는 요청을 전달합니다.

그러나 커넥터 제어기가 서비스 호출 요청을 전달한 후 목적지 응용프로그램의 응용프로그램 특정 구성요소가 사용 불가능하게 되면, 커넥터 제어기가 요청에 실패합니다.

이 등록 정보가 false로 설정된 경우, 커넥터 제어기는 목적지 응용프로그램 특정 구성요소가 사용 불가능함을 발견하는 즉시 모든 서비스 호출 요청에 실패하기 시작합니다.

기본값은 true입니다.

ControllerTraceLevel

RepositoryDirectory가 <REMOTE>인 경우에만 적용 가능합니다.

커넥터 제어기의 추적 메시지 레벨입니다. 기본값은 0입니다.

DeliveryQueue

DeliveryTransport가 JMS인 경우에만 적용 가능합니다.

Business Object를 통합 브로커에 전송하기 위해 커넥터에서 사용하는 대기열입니다.

기본값은 CONNECTORNAME/DELIVERYQUEUE입니다.

DeliveryTransport

이벤트 전달에 대한 전송 메커니즘을 지정합니다. 가능한 값은 WebSphere MQ의 경우 MQ, CORBA IIOP의 경우 IDL 또는 Java Messaging Service의 경우 JMS입니다.

- 브로커 유형이 ICS인 경우, DeliveryTransport 등록 정보의 값은 MQ, IDL 또는 JMS이며, 기본값은 IDL입니다.
- RepositoryDirectory가 로컬 디렉토리이면 JMS만이 값이 될 수 있습니다.

DeliveryTransport 등록 정보에 구성된 값이 MQ 또는 IDL일 경우, 커넥터는 CORBA IIOP를 통해 서비스 호출 요청과 관리 메시지를 전송합니다.

WebSphere MQ 및 IDL

한 제품만을 사용해야 하는 경우가 아니면, 이벤트 전달 전송에 IDL 대신 WebSphere MQ를 사용하십시오. WebSphere MQ는 IDL을 통해 다음과 같은 장점을 제공합니다.

- 비동기 통신:
  WebSphere MQ는 서버를 사용할 수 없는 경우에도 응용프로그램 특정 구성요소가 이벤트를 폴링하여 지속적으로 저장할 수 있게 합니다.
- 서버측 성능:
  WebSphere MQ는 서버측에서 더 빠른 성능을 제공합니다. 최적화된 모드에서, WebSphere MQ는 실제 이벤트가 WebSphere MQ 대기열에 남아 있는 반면 저장소 데이터베이스에는 이벤트에 대한 포인터만 저장합니다. 이로 인해 잠재적으로 큰 이벤트를 저장소 데이터베이스에 기록하지 않아도 됩니다.
- 에이전트측 성능:
  WebSphere MQ는 응용프로그램 특정 구성요소측에서 더 빠른 성능을 제공합니다.

커넥터의 폴링 스레드가 WebSphere MQ를 사용하여 이벤트를 선택하고, 이를 커넥터의 대기열에 넣은 후 다음 이벤트를 선택합니다. 커넥터의 폴링 스레드가 이벤트를 선택하고 네트워크에서 서버 프로세스로 이동하며 지속적으로 이벤트를 저장소 데이터베이스에 저장한 후 다음 이벤트를 선택해야 하는 IDL보다 더 빠릅니다.

JMS

JMS(Java Messaging Service)를 사용하여 커넥터와 클라이언트 커넥터 프레임워크 간의 통신을 사용 가능하게 합니다.

전달 전송으로 JMS를 선택하면 추가 JMS 등록 정보인 `jms.MessageBrokerName`, `jms.FactoryClassName`, `jms.Password` 및 `jms.UserName`이 Connector Configurator에 표시됩니다. 이들 등록 정보 중 처음 두 가지는 이 전송에 필요합니다.

중요: 다음 환경에서 JMS 전송 메커니즘을 커넥터에 사용하는 경우에는 메모리 제한이 있을 수 있습니다.

- AIX 5.0
- WebSphere MQ 5.3.0.1
- ICS가 통합 브로커일 때

이 환경에서는 WebSphere MQ 클라이언트 내의 메모리 사용으로 인해 커넥터 제어기(서버측) 및 커넥터(클라이언트측) 모두를 시작하는 것이 어려울 수 있습니다. 설치 시 768M 미만의 프로세스 힙 *크기*를 사용하는 경우, IBM은 다음의 설정을 권장합니다.

- `CWSharedEnv.sh` 스크립트의 `LDR_CNTRL` 환경 변수.

  이 스크립트는 제품 디렉토리 아래의 `\bin` 디렉토리에 상주합니다. 텍스트 편집기를 사용하여 다음 행을 `CWSharedEnv.sh` 스크립트의 첫 번째 행으로 추가하십시오.

  ```
  export LDR_CNTRL=MAXDATA=0x30000000
  ```

  이 행은 힙 메모리 사용을 최대 768MB(3세그먼트 * 256MB)로 제한합니다. 프로세스 메모리가 이 한계 이상으로 증가하면 페이지 스와핑이 발생할 수 있습니다. 이는 시스템의 성능을 저하시킬 수 있습니다.
- `IPCCBaseAddress` 등록 정보를 값 `11` 또는 `12`로 설정. 이 등록 정보에 대한 자세한 정보는 *UNIX용 시스템 설치 안내서*를 참조하십시오.

DuplicateEventElimination

이 등록 정보를 `true`로 설정할 때, JMS 사용 커넥터는 중복된 이벤트가 전달 대기열로 전달되지 않게 보장할 수 있습니다. 이 기능을 사용하려면 응용프로그램 특정 코드에서 Business Object의 **ObjectEventId** 속성으로 설정된 커넥터에 고유한 이벤트 ID가 있어야 합니다. 이는 커넥터 개발 중에 수행됩니다.

또한 이 등록 정보는 `false`로 설정될 수 있습니다.

주: DuplicateEventElimination이 true로 설정될 때, 보증된 이벤트 전달이 사용 가능하도록 MonitorQueue 등록 정보를 구성해야 합니다.

FaultQueue

메시지를 처리하는 동안 커넥터에서 오류가 발생한 경우, 커넥터는 상태 표시기 및 문제점 설명과 함께 이 등록 정보에 지정한 대기열로 메시지를 이동시킵니다.

기본값은 CONNECTORNAME/FAULTQUEUE입니다.

JvmMaxHeapSize

에이전트의 최대 힙 *크기*(MB)입니다. 이 등록 정보는 RepositoryDirectory 값이 <REMOTE>인 경우에만 적용 가능합니다.

기본값은 128m입니다.

JvmMaxNativeStackSize

에이전트의 최대 기본 스택 *크기*(KB)입니다. 이 등록 정보는 RepositoryDirectory 값이 <REMOTE>인 경우에만 적용 가능합니다.

기본값은 128k입니다.

JvmMinHeapSize

에이전트의 최소 힙 *크기*(MB)입니다. 이 등록 정보는 RepositoryDirectory 값이 <REMOTE>인 경우에만 적용 가능합니다.

기본값은 1m입니다.

jms.FactoryClassName

JMS 제공자가 인스턴스로 생성할 클래스 이름을 지정합니다. 전달 전송 메커니즘(DeliveryTransport)으로 JMS를 선택할 때 *반드시* 이 커넥터 등록 정보를 설정해야 합니다.

기본값은 CxCommon.Messaging.jms.IBMMQSeriesFactory입니다.

jms.MessageBrokerName

JMS 제공자가 사용할 브로커 이름을 지정합니다. 전달 전송 메커니즘(DeliveryTransport)으로 JMS를 선택할 때 *반드시* 이 커넥터 등록 정보를 설정해야 합니다.

기본값은 crossworlds.queue.manager입니다.

jms.NumConcurrentRequests

동시에 커넥터에 전송할 수 있는 최대 동시 서비스 호출 요청 수를 지정합니다. 최대값에 일단 도달하면, 새 서비스 호출이 차단되고 계속 진행하기 전에 다른 요청이 완료될 때까지 대기합니다.

기본값은 10입니다.

jms.Password

JMS 제공자의 암호를 지정합니다. 이 등록 정보에 대한 값은 선택적입니다.

기본값은 없습니다.

jms.UserName

JMS 제공자의 사용자 이름을 지정합니다. 이 등록 정보에 대한 값은 선택적입니다.

기본값은 없습니다.

ListenerConcurrency

이 등록 정보는 ICS가 통합 브로커일 때 MQ Listener의 멀티스레딩을 지원합니다. 이는 데이터베이스에 대한 복수 이벤트의 일괄처리 쓰기를 가능하게 하여 시스템 성능을 향상시킵니다. 기본값은 1입니다.

이 등록 정보는 MQ 전송을 사용하는 커넥터에만 적용됩니다. `DeliveryTransport` 등록 정보를 `MQ`로 설정해야 합니다.

Locale

언어 코드, 국가 또는 지역, 그리고 선택적으로 연관된 문자 코드 세트를 지정합니다. 이 등록 정보의 값은 데이터의 배열 및 정렬 순서, 날짜 및 시간 형식, 화폐 스펙에서 사용되는 기호와 같은 문화적 규약을 판별합니다.

로케일 이름의 형식은 다음과 같습니다.

ll_TT.codeset

여기서

ll	2문자 언어 코드(보통 소문자)
TT	2문자 국가 또는 지역 코드(보통 대문자)
codeset	연관된 문자 코드 세트의 이름입니다. 이름의 이 부분은 대체로 선택적입니다.

기본적으로, 지원되는 로케일의 서브세트만 드롭 목록에 표시됩니다. 기타 지원되는 값을 드롭 목록에 추가하려면, 제품 디렉토리에 있는 \Data\Std\stdConnProps.xml 파일을 직접 수정해야 합니다. 자세한 정보는 Connector Configurator의 부록을 참조하십시오.

기본값은 en_US입니다. 커넥터가 국제화되지 않은 경우, 이 등록 정보의 유일한 올바른 값은 en_US입니다. 특정 커넥터가 국제화되었는지 여부를 판별하려면 다음 웹사이트의 커넥터 버전 목록을 참조하십시오.

http://www.ibm.com/software/websphere/wbiadapters/infocenter 또는
http://www.ibm.com/websphere/integration/wicserver/infocenter

LogAtInterchangeEnd

RespositoryDirectory가 <REMOTE>인 경우에만 적용 가능합니다.

통합 브로커의 로그 목적지에 오류를 기록할지 여부를 지정합니다. 브로커의 로그 목적지에 기록하면, 전자 우편 공고가 켜져서 오류 또는 심각한 오류가 발생할 때 InterchangeSystem.cfg 파일에 지정된 MESSAGE_RECIPIENT에 대한 전자 우편 공고를 생성합니다.

예를 들어, 커넥터에서 응용프로그램에 대한 연결이 끊어질 때 LogAtInterChangeEnd가 true로 설정되면, 전자 우편 메시지가 지정된 메시지 수신자에게 전송됩니다. 기본값은 false입니다.

MaxEventCapacity

제어기 버퍼의 최대 이벤트 수입니다. 이 등록 정보는 플로우 제어에서 사용되며 RepositoryDirectory 등록 정보의 값이 <REMOTE>인 경우에만 적용 가능합니다.

값은 1 - 2147483647의 양수일 수 있습니다. 기본값은 2147483647입니다.

MessageFileName

커넥터 메시지 파일의 이름입니다. 메시지 파일의 표준 위치는 \connectors\messages입니다. 메시지 파일이 표준 위치에 없으면 절대 경로에 메시지 파일 이름을 지정하십시오.

커넥터 메시지 파일이 존재하지 않을 경우, 커넥터는 InterchangeSystem.txt를 메시지 파일로 사용합니다. 이 파일은 제품 디렉토리에서 발견됩니다.

주: 특정 커넥터에 자체 메시지 파일이 있는지 판별하려면 개별 어댑터 사용자 안내서를 참조하십시오.

MonitorQueue

중복 이벤트를 모니터하기 위해 커넥터에서 사용하는 논리 대기열입니다. 이 등록 정보는 DeliveryTransport 등록 정보 값이 JMS이고 DuplicateEventElimination이 TRUE로 설정되는 경우에만 사용됩니다.

기본값은 CONNECTORNAME/MONITORQUEUE입니다.

OADAutoRestartAgent

RepositoryDirectory가 <REMOTE>인 경우에만 올바릅니다.

커넥터가 자동 및 원격 다시 시작 기능을 사용할지 여부를 지정합니다. 이 기능은 비정상 종료 이후에 커넥터를 다시 시작하거나 System Monitor로부터 원격 커넥터를 시작할때 MQ 트리거 OAD(Object Activation Daemon)를 사용합니다.

자동 및 원격 다시 시작 기능을 사용 가능하게 하려면 이 등록 정보를 true로 설정해야 합니다. MQ 트리거 OAD 기능을 구성하는 방법에 대해서는 *Windows용 설치 안내서* 또는 *UNIX용 설치 안내서*를 참조하십시오.

기본값은 false입니다.

OADMaxNumRetry

RepositoryDirectory가 <REMOTE>인 경우에만 올바릅니다.

비정상 종료 후 MQ 트리거 OAD가 자동으로 커넥터를 다시 시작하려고 시도하는 최대 횟수를 지정합니다. 이 등록 정보를 적용하려면 OADAutoRestartAgent 등록 정보를 true로 설정해야 합니다.

기본값은 1000입니다.

OADRetryTimeInterval

RepositoryDirectory가 <REMOTE>인 경우에만 올바릅니다.

MQ 트리거 OAD에 대한 재시도 시간 간격에 시간(분 단위)을 지정합니다. 이 재시도 시간 간격 이내에 커넥터 에이전트가 다시 시작되지 않으면 커넥터 제어기는 OAD에게 커넥터 에이전트를 다시 시작하도록 요청합니다. OAD는 OADMaxNumRetry 등록 정보에 지정된 횟수만큼 이 재시도 프로세스를 반복합니다. 이 등록 정보를 적용하려면 OADAutoRestartAgent 등록 정보를 true로 설정해야 합니다.

기본값은 10입니다.

PollEndTime

이벤트 대기열 폴링을 중지할 시간입니다. 형식은 HH:MM이며, 여기서 *HH*는 0 - 23의 시간, *MM*은 0 - 59의 분을 나타냅니다.

이 등록 정보에 올바른 값을 제공해야 합니다. 기본값은 HH:MM이지만, 변경해야 합니다.

PollFrequency

각 폴링 조치 사이의 시간값입니다. PollFrequency를 다음 값 중 하나로 설정하십시오.

- 폴링 조치 사이의 밀리초 수.
- 단어 key - 사용자가 커넥터의 명령 프롬프트 창에 영문자 p를 입력할 경우에만 커넥터가 폴링하도록 합니다. 단어는 소문자로 입력하십시오.
- 단어 no - 커넥터가 폴링하지 않게 합니다. 단어는 소문자로 입력하십시오.

기본값은 10000입니다.

중요: 일부 커넥터는 이 등록 정보 사용에 제한을 가지고 있습니다. 특정 커넥터가 이 등록 정보를 사용하는지 여부를 판별하려면 해당 어댑터 안내서의 설치 및 구성 장을 참조하십시오.

PollQuantity

커넥터가 폴링해야 하는 응용프로그램으로부터의 항목 수를 지정합니다. 어댑터에 폴링 수를 설정하기 위한 커넥터 특정 등록 정보가 있으면, 커넥터 특정 등록 정보에 설정된 값이 표준 등록 정보 값을 대체합니다.

PollStartTime

이벤트 대기열 폴링을 시작할 시간입니다. 형식은 *HH:MM*이며, 여기서 *HH*는 0 - 23의 시간, *MM*은 0 - 59의 분을 나타냅니다.

이 등록 정보에 올바른 값을 제공해야 합니다. 기본값은 HH:MM이지만, 변경해야 합니다.

RequestQueue

Business Object를 커넥터에 전송하기 위해 통합 브로커에서 사용하는 대기열입니다.

기본값은 CONNECTOR/REQUESTQUEUE입니다.

RepositoryDirectory

Business Object 정의의 메타 데이터를 저장하는 XML 스키마 문서를 커넥터가 읽는 저장소의 위치입니다.

통합 브로커가 ICS이면, 커넥터가 InterChange Server 저장소에서 정보를 얻으므로 이 값을 <REMOTE>로 설정해야 합니다.

통합 브로커가 WebSphere 메시지 브로커 또는 WAS이면 이 값을 *<로컬 디렉토리>*로 설정해야 합니다.

ResponseQueue

DeliveryTransport가 JMS인 경우에만 적용 가능하며 RepositoryDirectory가 <REMOTE>인 경우에만 필수입니다.

커넥터 프레임워크에서 통합 브로커로 응답 메시지를 전달하는 JMS 응답 대기열을 지정합니다. 통합 브로커가 ICS이면 서버는 요청을 전송한 후 JMS 응답 대기열에서 응답을 기다립니다.

RestartRetryCount

커넥터가 커넥터 자체를 다시 시작하려고 하는 횟수를 지정합니다. 병렬 커넥터에 사용한 경우, 마스터 커넥터 응용프로그램 특정 구성요소가 슬레이브 커넥터 응용프로그램 특정 구성요소를 다시 시작하려고 시도하는 횟수를 지정합니다.

기본값은 3입니다.

RestartRetryInterval

커넥터가 커넥터 자체를 다시 시작하려고 시도하는 간격을 분 수로 지정합니다. 병렬 커넥터에 사용한 경우, 마스터 커넥터 응용프로그램 특정 구성요소가 슬레이브 커넥터 응용프로그램 특정 구성요소를 다시 시작하려고 시도하는 간격을 지정합니다. 가능한 값은 1 - 2147483647입니다.

기본값은 1입니다.

SourceQueue

DeliveryTransport가 JMS이고 ContainerManagedEvents가 지정된 경우에만 적용 가능합니다.

JMS 이벤트 저장소를 사용하는 JMS 사용 커넥터에 대해 보증된 이벤트 전달을 지원하는 커넥터 프레임워크의 JMS 소스 대기열을 지정합니다. 자세한 정보는 106 페이지의 『ContainerManagedEvents』를 참조하십시오.

기본값은 CONNECTOR/SOURCEQUEUE입니다.

SynchronousRequestQueue

DeliveryTransport가 JMS인 경우에만 적용 가능합니다.

커넥터 프레임워크에서 브로커로의 동기 응답을 요구하는 요청 메시지를 전달합니다. 이 대기열은 커넥터가 동기 실행을 사용할 경우에만 필요합니다. 동기 실행을 사용할 때, 커넥터 프레임워크는 메시지를 SynchronousRequestQueue로 전송하고

SynchronousResponseQueue에서 브로커의 응답을 기다립니다. 커넥터에 전송되는 응답 메시지는 원래 메시지 ID와 일치하는 Correlation ID를 유지합니다.

기본값은 CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE입니다.

SynchronousResponseQueue

DeliveryTransport가 JMS인 경우에만 적용 가능합니다.

브로커에서 커넥터 프레임워크에 이르는 동기 요청에 대한 응답으로 전송되는 응답 메시지를 전달합니다. 이 대기열은 커넥터가 동기 실행을 사용할 경우에만 필요합니다.

기본값은 CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE입니다.

SynchronousRequestTimeout

DeliveryTransport가 JMS인 경우에만 적용 가능합니다.

커넥터가 동기 요청에 대한 응답을 기다리는 시간(분)을 지정합니다. 지정된 시간 내에 응답이 수신되지 않을 경우, 커넥터는 원래 동기 요청 메시지를 오류 메시지와 함께 결함 대기열로 이동합니다.

기본값은 0입니다.

WireFormat

전송 시 메시지 형식입니다.

- RepositoryDirectory가 로컬 디렉토리이면 설정은 CwXML입니다.
- RepositoryDirectory의 값이 <REMOTE>이면 설정은 CwBO입니다.

WsifSynchronousRequest 제한시간

WAS 통합 브로커에만 해당됩니다.

커넥터가 동기 요청에 대한 응답을 기다리는 시간(분)을 지정합니다. 지정된 시간 내에 응답이 수신되지 않을 경우, 커넥터는 원래 동기 요청 메시지를 오류 메시지와 함께 결함 대기열로 이동합니다.

기본값은 0입니다.

XMLNameSpaceFormat

WebSphere 메시지 브로커 및 WAS 통합 브로커에만 해당됩니다.

사용자가 XML 형식의 Business Object 정의에 짧고 긴 이름 공간을 지정할 수 있도록 하는 강력한 등록 정보입니다.

기본값은 short입니다.

부록 B. Connector Configurator

이 부록에서는 Connector Configurator를 사용하여 어댑터의 구성 등록 정보 값을 설정하는 방법에 대해 설명합니다.

Connector Configurator를 사용하여 다음을 수행할 수 있습니다.

- 커넥터 구성을 위한 커넥터 특정 등록 정보 템플리트 작성
- 구성 파일 작성
- 구성 파일에 등록 정보 설정

주:

이 책에서 백슬래시(\)는 디렉토리 경로의 규칙으로 사용됩니다. UNIX 설치의 경우, 백슬래시를 슬래시(/)로 대체하고 각 운영 체제의 규칙을 따르십시오.

이 부록에서 다루는 주제는 다음과 같습니다.

Connector Configurator 개요

Connector Configurator를 사용하면 다음 통합 브로커에서 사용하는 어댑터의 커넥터 구성요소를 구성할 수 있습니다.

- WebSphere InterChange Server(ICS)
- WebSphere MQ Integrator, WebSphere MQ Integrator Broker 및 WebSphere Business Integration Message Broker를 통털어 WMQI(WebSphere Message Brokers)라고 합니다.
- WebSphere Application Server(WAS)

Connector Configurator를 사용하여 다음을 수행할 수 있습니다.

- 커넥터 구성을 위한 **커넥터 특정 등록 정보 템플리트** 작성
- **커넥터 구성 파일** 작성: 설치한 각 커넥터당 하나의 구성 파일을 작성해야 합니다.
- 구성 파일에 등록 정보 설정
  커넥터 템플리트의 등록 정보에 설정된 기본값을 수정해야 할 경우가 있습니다. 또한

필요한 경우, 메시징, 로깅 및 추적, Data Handler 매개변수 지정과 더불어 지원되는 Business Object 정의와 ICS와 함께 협업에 사용하기 위한 맵을 지정해야 합니다.

Connector Configurator를 실행하는 모드와, 사용하는 구성 파일 유형은 실행 중인 통합 브로커에 따라 다를 수 있습니다. 예를 들어, WMQI가 브로커인 경우, System Manager 내에서가 아니라 직접 Connector Configurator를 실행합니다(119 페이지의 『독립형 모드로 Configurator 실행』 참조).

커넥터 구성 등록 정보에는 표준 구성 등록 정보(모든 커넥터가 가지고 있는 등록 정보) 및 커넥터 특정 등록 정보(특정 응용프로그램이나 기술에 따라 커넥터에 필요한 등록 정보)가 둘 다 포함됩니다.

표준 등록 정보는 모든 커넥터에서 사용되기 때문에, 이러한 등록 정보를 처음부터 정의할 필요는 없습니다. 파일을 작성하는 즉시 Connector Configurator는 이들을 구성 파일로 통합합니다. 그러나 Connector Configurator에서 각 표준 등록 정보 값을 설정해야 합니다.

모든 브로커와 모든 구성에서 표준 등록 정보의 범위가 동일하지 않을 수도 있습니다. 일부 등록 정보는 다른 등록 정보에 특정 값이 제공되는 경우에만 사용 가능합니다. Connector Configurator의 표준 등록 정보 창에는 특정 구성에 사용 가능한 등록 정보가 표시됩니다.

그러나 **커넥터 특정 등록 정보**의 경우 먼저 등록 정보를 정의한 다음 해당 값을 설정해야 합니다. 특정 어댑터의 커넥터 특정 등록 정보 템플리트를 작성하여 이를 수행합니다. 시스템에 템플리트가 이미 설정되어 있으면 이 템플리트를 사용하면 됩니다. 그렇지 않은 경우, 120 페이지의 『새 템플리트 작성』의 단계에 따라 새 템플리트를 설정하십시오.

주: Connector Configurator는 Windows 환경에서만 실행합니다. UNIX 환경에서 커넥터를 실행하고 있는 경우, Windows에서 Connector Configurator를 사용하여 구성 파일을 수정한 다음 파일을 UNIX 환경으로 복사하십시오.

Connector Configurator 시작

다음 두 가지 모드 중 하나로 Connector Configurator를 시작하고 실행할 수 있습니다.

- 독립형 모드로 독립적으로 시작 및 실행
- System Manager에서 시작 및 실행

독립형 모드로 Configurator 실행

Connector Configurator를 독립적으로 실행하고 브로커와 무관하게 커넥터 구성 파일에 대해 작업할 수 있습니다.

이를 수행하려면 다음 단계를 따르십시오.

- **시작** > **프로그램**에서 **IBM WebSphere InterChange Server** > **IBM WebSphere Business Integration Toolset** > **개발** > **Connector Configurator**를 누르십시오.
- **파일** > **새로 작성** > **구성 파일**을 선택하십시오.
- **시스템 연결 통합 브로커** 옆에 있는 풀다운 메뉴를 누르면 브로커에 따라 ICS, WebSphere 메시지 브로커 또는 WAS를 선택할 수 있습니다.

Connector Configurator를 별도로 실행하도록 선택하여 파일을 생성한 다음 System Manager에 연결하여 파일을 System Manager 프로젝트에 저장할 수 있습니다(124 페이지의 『구성 파일 완료』 참조).

System Manager에서 Configurator 실행

System Manager에서 Connector Configurator를 실행할 수 있습니다.

Connector Configurator를 실행하려면 다음을 수행하십시오.

1. System Manager를 여십시오.
2. System Manager 창에서 **통합 구성요소 라이브러리** 아이콘을 펼치고 **커넥터**를 강조표시하십시오.
3. System Manager 메뉴 표시줄에서 **도구** > **Connector Configurator**를 누르십시오. Connector Configurator 창이 열리고 **새 커넥터** 대화 상자가 표시됩니다.
4. **시스템 연결 통합 브로커** 옆에 있는 풀다운 메뉴를 누르면 브로커에 따라 ICS, WebSphere 메시지 브로커 또는 WAS를 선택할 수 있습니다.

기존 구성 파일을 편집하려면 다음을 수행하십시오.

1. System Manager 창에서 커넥터 폴더에 나열된 구성 파일 중 하나를 선택하고 이를 마우스 오른쪽 단추로 누르십시오. Connector Configurator가 열리고 맨 위에 통합 브로커 유형 및 파일 이름과 함께 구성 파일이 표시됩니다.
2. 표준 등록 정보 탭을 눌러 이 구성 파일에 포함되어 있는 등록 정보를 보십시오.

커넥터 특정 등록 정보 템플리트 작성

커넥터에 맞는 구성 파일을 작성하려면, 시스템 제공 표준 등록 정보 외에 커넥터 특정 등록 정보 템플리트가 필요합니다.

커넥터의 커넥터 특정 등록 정보에 맞는 새 템플리트를 작성하거나, 기존 파일을 템플리트로 사용할 수 있습니다.

- 새 템플리트를 작성하려면 120 페이지의 『새 템플리트 작성』을 참조하십시오.
- 기존 파일을 사용하려면 기존 템플리트를 수정하여 새 이름으로 저장하기만 하면 됩니다.

새 템플리트 작성

이 절에서는 템플리트에서 등록 정보를 작성하고, 이러한 등록 정보의 일반 특성 및 값을 정의하며, 등록 정보 사이에 종속성을 지정하는 방법에 대해 설명합니다. 그런 다음 템플리트를 저장하여 새 커넥터 구성 파일을 작성하기 위한 기본으로 사용합니다.

템플리트를 작성하려면 다음을 수행하십시오.

1. **파일 > 새로 작성 > 커넥터 특정 등록 정보 템플리트**를 누르십시오.
2. 다음 필드가 있는 **커넥터 특정 등록 정보 템플리트** 대화 상자가 나타납니다.
   - **템플리트** 및 **이름**

     커넥터를 식별하는 고유한 이름을 입력하거나, 이 템플리트를 사용할 커넥터 유형을 입력하십시오. 템플리트에서 새 구성 파일을 작성하기 위해 대화 상자를 열 때 이 이름이 다시 표시됩니다.
   - **이전 템플리트** 및 **수정할 기존 템플리트 선택**

     현재 사용 가능한 모든 템플리트 이름이 템플리트 이름 화면에 표시됩니다.
   - 템플리트의 커넥터 특정 등록 정보 정의를 보려면 **템플리트 이름** 화면에서 해당 템플리트의 이름을 선택하십시오. 해당 템플리트에 포함된 등록 정보 정의 목록이 **템플리트 미리보기** 화면에 나타납니다. 등록 정보 정의가 커넥터에 필요한 것과 유사한 기존 템플리트를 사용자 템플리트를 위한 출발점으로 사용할 수 있습니다.
3. **템플리트 이름** 화면에서 템플리트를 선택하고, **이름 찾기** 필드에 해당 템플리트 이름을 입력하고(또는 **템플리트 이름**에서 선택사항을 강조표시하고), **다음**을 누르십시오.

커넥터에서 사용하는 커넥터 특정 등록 정보를 표시하는 템플리트가 보이지 않으면 템플리트를 작성해야 합니다.

일반 특성 지정

다음을 누르면 템플리트를 선택할 수 있는 **등록 정보 - 커넥터 특정 등록 정보 템플리트** 대화 상자가 나타납니다. 대화 상자에는 정의된 등록 정보의 일반 특성과 값 제한사항에 관한 탭이 있습니다. 일반 화면에는 다음 필드가 있습니다.

- **일반**

  등록 정보 유형

  갱신 메소드

  설명

- **플래그**
  표준 플래그
- **사용자 정의 플래그**
  플래그

등록 정보의 일반 특성을 선택한 후, **값** 탭을 누르십시오.

값 지정

값 탭을 사용하여 등록 정보에 대한 최대 길이, 최대 복수 값, 기본값 또는 값 범위를 설정할 수 있습니다. 편집 가능한 값도 허용됩니다. 이를 수행하려면 다음 단계를 따르십시오.

1. **값** 탭을 누르십시오. 일반 표시 패널이 값 표시 패널로 바뀝니다.
2. **등록 정보 편집** 화면에서 등록 정보의 이름을 선택하십시오.
3. **최대 길이** 및 **최대 복수 값**에 대한 필드에서 필요한 변경을 수행하십시오. 다음 단계에 설명된 대로 등록 정보의 **등록 정보 값** 대화 상자를 열 때까지는 변경사항이 승인되지 않습니다.
4. 값 테이블의 왼쪽 맨 위 구석에 있는 상자를 마우스 오른쪽 단추로 누른 다음 **추가**를 누르십시오. **등록 정보 값** 대화 상자가 나타납니다. 등록 정보 유형에 따라, 대화 상자에서 값을 입력하거나 값과 범위를 모두 입력할 수 있습니다. 적절한 값이나 범위를 입력하고 **확인**을 누르십시오.
5. **값** 패널이 **최대 길이** 및 **최대 복수 값**에서 수행한 모든 변경사항을 표시하기 위해 새로 고쳐집니다. 세 개의 열이 있는 다음 테이블이 나타납니다.

   값 열은 사용자가 **등록 정보 값** 대화 상자에 입력한 값과, 사용자가 작성한 모든 이전 값을 표시합니다.

   기본값 열을 사용하면 임의의 값을 기본값으로 지정할 수 있습니다.

   값 범위는 **등록 정보 값** 대화 상자에서 입력한 범위를 표시합니다.

   값을 작성한 후 값이 격자에 나타나면, 테이블 화면에서 이를 편집할 수 있습니다. 테이블의 기존 값을 변경하려면 행 번호를 눌러 전체 행을 선택하십시오. 그런 다음 **값** 필드를 마우스 오른쪽 단추로 누르고 **값 편집**을 누르십시오.

종속성 설정

일반 및 **값** 탭에서 변경을 수행한 후, **다음**을 누르십시오. **종속성 - 커넥터 특정 등록 정보 템플리트** 대화 상자가 나타납니다.

종속 등록 정보는 다른 등록 정보의 값이 특정 조건에 부합하는 *경우에만* 템플리트에 포함되어 구성 파일에 사용되는 등록 정보입니다. 예를 들어, `PollQuantity`는 JMS가 전송 메커니즘이고 `DuplicateEventElimination`이 `True`로 설정되는 경우에만 템플리트에 나타납니다.
등록 정보가 종속되도록 지정하고 종속 조건을 설정하려면, 다음을 수행하십시오.

1. **사용 가능한 등록 정보** 화면에서 종속될 등록 정보를 선택하십시오.
2. **등록 정보 선택** 필드에서 드롭 다운 메뉴를 사용하여 조건 값을 보유할 등록 정보를 선택하십시오.
3. **조건 연산자** 필드에서 다음 중 하나를 선택하십시오.

   == (같음)

   != (같지 않음)

   > (초과)

   < (미만)

   >= (이상)

   <= (이하)
4. **조건 값** 필드에 종속 등록 정보를 템플리트에 포함하기 위해 필요한 값을 입력하십시오.
5. **사용 가능한 등록 정보** 화면에서 종속 등록 정보가 강조표시된 상태에서 화살표를 눌러서 **종속 등록 정보** 화면으로 이동하십시오.
6. **완료**를 누르십시오. Connector Configurator가 Connector Configurator를 설치한 `\bin` 디렉토리의 `\data\app`에 XML 문서로 입력한 정보를 저장합니다.

새 구성 파일 작성

새 구성 파일을 작성하는 첫 단계는 통합 브로커를 선택하는 것입니다. 선택하는 브로커에 따라 구성 파일에 표시될 등록 정보가 달라집니다.

브로커를 선택하려면 다음을 수행하십시오.

- Connector Configurator 홈 메뉴에서 **파일 > 새로 작성 > 커넥터 구성**을 누르십시오. **새 커넥터** 대화 상자가 나타납니다.
- **통합 브로커** 필드에서 ICS, WebSphere 메시지 브로커 또는 WAS 연결을 선택하십시오.
- 이 장의 뒷부분에서 설명된 대로 **새 커넥터** 창에서 나머지 필드를 완료하십시오.

다음 작업도 수행할 수 있습니다.

- System Manager 창에서 **커넥터** 폴더를 마우스 오른쪽 단추로 누르고 **새 커넥터 작성**을 선택하십시오. Connector Configurator가 열리고 **새 커넥터** 대화 상자가 표시됩니다.

커넥터 특정 템플리트에서 구성 파일 작성

커넥터 특정 템플리트를 작성한 후, 이 템플리트를 사용하여 구성 파일을 작성할 수 있습니다.

1. **파일 > 새로 작성 > 커넥터 구성**을 누르십시오.

2. 다음 필드가 있는 **새 커넥터** 대화 상자가 나타납니다.
   - **이름**

     커넥터의 이름을 입력하십시오. 이름은 대소문자를 구분합니다. 입력하는 이름은 고유해야 하며 시스템에 설치된 커넥터의 파일 이름과 일치해야 합니다.

     중요: Connector Configurator는 사용자가 입력한 이름의 철자를 확인하지 않습니다. 사용자가 이름이 정확한지 확인해야 합니다.
   - **시스템 연결**

     ICS, WebSphere 메시지 브로커 또는 WAS를 누르십시오.
   - **커넥터 특정 등록 정보 템플리트**

     커넥터에 맞게 설계된 템플리트의 이름을 입력하십시오. 사용 가능한 템플리트가 **템플리트 이름** 화면에 표시됩니다. 템플리트 이름 화면에서 이름을 선택할 경우 **등록 정보 템플리트 미리보기** 화면이 해당 템플리트에 정의된 커넥터 특정 등록 정보를 표시합니다.

     사용하려는 템플리트를 선택하고 **확인**을 누르십시오.
3. 구성 중인 커넥터의 구성 화면이 표시됩니다. 제목 표시줄이 통합 브로커와 커넥터 이름을 표시합니다. 모든 필드값을 입력하여 지금 정의를 완료하거나, 파일을 저장했다가 나중에 필드를 완성할 수 있습니다.
4. 파일을 저장하려면 **파일 > 저장 > 파일에** 또는 **파일 > 저장 > 프로젝트에**를 누르십시오. 프로젝트를 저장하려면 System Manager가 실행 중이어야 합니다.
   파일로 저장하는 경우 **파일 커넥터 저장** 대화 상자가 나타납니다. 파일 유형으로 `*.cfg`를 선택하고 파일 이름 필드에서 이름의 철자 및 대소문자가 올바른지 확인한 후, 파일을 저장할 디렉토리로 이동하고 **저장**을 누르십시오. Connector Configurator의 메시지 패널에 있는 상태 화면은 구성 파일이 성공적으로 작성되었음을 표시합니다.

   중요: 여기에서 설정하는 디렉토리 경로 및 이름은 커넥터에 대해 시작 파일에 제공한 커넥터 구성 파일 경로 및 이름과 일치해야 합니다.
5. 커넥터 정의를 완료하려면, 이 장의 뒷부분에 설명된 대로 Connector Configurator 창의 각 탭에 있는 필드에 값을 입력하십시오.

기존 파일 사용

다음 중 하나 이상의 형식으로 기존 파일을 사용할 수 있습니다.

- 커넥터 정의 파일.
  이 파일은 특정 커넥터의 등록 정보와 적용 가능한 기본값을 나열한 텍스트 파일입니다. 일부 커넥터는 전달 패키지의 `\repository` 디렉토리에 이러한 파일을 포함합니다. (이 파일의 확장자는 보통 `.txt`입니다. 예를 들어, XML 커넥터의 경우에는 CN_XML.txt입니다.)

- ICS 저장소 파일.
  커넥터의 이전 ICS 구현에서 사용된 정의는 해당 커넥터의 구성에 사용된 저장소 파일에서 사용할 수도 있습니다. 이러한 파일의 확장자는 일반적으로 .in 또는 .out입니다.
- 커넥터의 이전 구성 파일.
  이러한 파일의 확장자는 일반적으로 *.cfg입니다.

이 파일 소스는 커넥터의 커넥터 특정 등록 정보의 전부 또는 대부분을 포함할 수 있지만, 커넥터 구성 파일은 이 장의 뒷부분에 설명된 대로 파일을 열고 등록 정보를 설정할 때까지 완료되지 않습니다.

기존 파일을 사용하여 커넥터를 구성하려면, Connector Configurator에서 파일을 열어 구성을 변경한 다음 파일을 다시 저장해야 합니다.

디렉토리에서 *.txt, *.cfg 또는 *.in 파일을 열려면 다음 단계를 따르십시오.

1. Connector Configurator에서 **파일 > 열기 > 파일에서**를 누르십시오.
2. **파일 커넥터 열기** 대화 상자에서 다음 파일 유형 중 하나를 선택해서 사용 가능한 파일을 보십시오.
   - 구성(*.cfg)
   - ICS 저장소(*.in, *.out)

     저장소 파일이 ICS 환경에서 커넥터를 구성하는 데 사용된 경우 이 옵션을 선택하십시오. 저장소 파일은 여러 개의 커넥터 정의를 포함할 수 있으며, 모든 커넥터 정의는 사용자가 파일을 열 때 표시됩니다.
   - 모든 파일(*.*)

     *.txt 파일이 커넥터용 어댑터 패키지로 전달되거나 확장자가 다른 정의 파일을 사용할 수 있는 경우 이 옵션을 선택하십시오.
3. 디렉토리 화면에서 적당한 커넥터 정의 파일로 이동하고 이를 선택한 후 **열기**를 누르십시오.

System Manager 프로젝트에서 커넥터 구성을 열려면 다음 단계를 따르십시오.

1. System Manager를 시작하십시오. System Manager가 시작된 경우에만 구성을 System Manager에서 열거나 System Manager에 저장할 수 있습니다.
2. Connector Configurator를 시작하십시오.
3. **파일 > 열기 > 프로젝트에서**를 누르십시오.

구성 파일 완료

프로젝트에서 구성 파일이나 커넥터를 열면, Connector Configurator 창에 현재 속성 및 값이 있는 구성 화면이 나타납니다.

구성 화면의 제목에는 파일에 지정된 커넥터 이름과 통합 브로커가 표시됩니다. 브로커가 올바른지 확인하십시오. 그렇지 않을 경우, 커넥터를 구성하기 전에 브로커 값을 변경하십시오. 이를 수행하려면 다음 단계를 따르십시오.

1. **표준 등록 정보** 탭에서 브로커 유형 등록 정보에 대한 값 필드를 선택하십시오. 드롭 다운 메뉴에서 ICS, WMQI 또는 WAS를 선택하십시오.
2. 표준 등록 정보 탭에 선택한 브로커와 연관된 등록 정보가 표시됩니다. 지금 파일을 저장하거나 128 페이지의 『지원되는 Business Object 정의 지정』에 설명된 대로 나머지 구성 필드를 완성할 수 있습니다.
3. 구성을 완료했을 때, **파일 > 저장 > 프로젝트에** 또는 **파일 > 저장 > 파일에**를 누르십시오.

   파일에 저장할 경우, 확장자로 *.cfg를 선택하고 파일의 올바른 위치를 선택한 후 **저장**을 누르십시오.

   여러 커넥터 구성이 열려 있으면 **파일에 모두 저장**을 눌러서 모든 구성을 파일에 저장하거나, **프로젝트에 모두 저장**을 눌러서 모든 커넥터 구성을 System Manager 프로젝트에 저장하십시오.

   Connector Configurator는 파일을 저장하기 전에 모든 필수 표준 등록 정보의 값이 설정되었는지 확인합니다. 필수 표준 등록 정보 값이 누락된 경우, Connector Configurator는 유효성 검증 실패를 알리는 메시지를 표시합니다. 구성 파일을 저장하려면 해당 등록 정보에 값을 제공해야 합니다.

구성 파일 등록 정보 설정

새 커넥터 구성 파일을 작성하여 이름을 지정하거나 기존 커넥터 구성 파일을 열 때, Connector Configurator는 필수 구성값의 카테고리에 대한 탭이 있는 구성 화면을 표시합니다.

Connector Configurator는 다음 범주에서 모든 브로커에서 실행 중인 커넥터에 대한 등록 정보의 값이 필요합니다.

- 표준 등록 정보
- 커넥터 특정 등록 정보
- 지원되는 Business Object
- 추적/로그 파일 값
- Data Handler(이벤트 전달이 보장되는 JMS 메시징을 사용하는 커넥터의 경우 적용 가능)

주: 데이터를 Business Object로 변환하는 Data Handler의 구성에서 JMS 메시징을 사용하는 커넥터의 경우, 추가 카테고리가 표시될 수 있습니다.

ICS에서 실행되는 커넥터의 경우, 다음 등록 정보의 값도 필요합니다.

- 연관된 맵
- 자원
- 메시징(적용 가능할 경우)

중요: Connector Configurator는 영어 또는 영어 이외의 문자 세트의 등록 정보 값을 승인합니다. 그러나 표준 및 커넥터 특정 등록 정보의 이름과 지원되는 Business Object의 이름은 영어 문자 세트만을 사용해야 합니다.

표준 등록 정보는 다음과 같이 커넥터 특정 등록 정보와 다릅니다.

- 커넥터의 표준 등록 정보는 커넥터의 응용프로그램 특정 구성요소와 해당 브로커 구성 요소에서 모두 공유됩니다. 모든 커넥터에는 동일한 표준 등록 정보 세트가 있습니다. 이러한 등록 정보는 각 어댑터 안내서의 부록 A에 설명되어 있습니다. 일부는 변경할 수 있지만 모든 값을 변경할 수는 없습니다.
- 응용프로그램 특정 등록 정보는 커넥터의 응용프로그램 특정 구성요소, 즉 응용프로그램과 직접 상호작용하는 구성요소에만 적용됩니다. 각 커넥터에는 커넥터의 응용프로그램에 고유한 응용프로그램 특정 등록 정보가 있습니다. 이러한 등록 정보 중 일부는 기본값을 제공하며, 일부는 기본값을 제공하지 않습니다. 일부 기본값은 수정할 수 있습니다. 각 어댑터 안내서의 설치 및 구성 장에서는 응용프로그램 특정 등록 정보와 권장값에 대해 설명합니다.

표준 등록 정보 및 **커넥터 특정 등록 정보**에 대한 값은 구성 가능한 필드를 표시하기 위해 색상을 사용하여 코딩됩니다.

- 회색 배경의 필드는 표준 등록 정보를 나타냅니다. 값은 변경할 수 있으나 이름을 변경하거나 등록 정보를 제거할 수는 없습니다.
- 흰색 배경의 필드는 응용프로그램 특정 등록 정보를 나타냅니다. 이들 등록 정보는 응용프로그램 또는 커넥터의 특정 필요에 따라 다릅니다. 값을 변경하고 이 등록 정보를 삭제할 수 있습니다.
- 값 필드는 구성 가능합니다.
- **갱신 메소드** 필드는 정보용이므로 구성할 수 없습니다. 이 필드는 값이 변경된 등록 정보를 활성화하기 위해 필요한 조치를 지정합니다.

표준 커넥터 등록 정보 설정

표준 등록 정보의 값을 변경하려면 다음을 수행하십시오.

1. 값을 설정하려는 필드를 누르십시오.
2. 값을 입력하거나, 값이 나타나는 경우 드롭 다운 메뉴에서 선택하십시오.
3. 표준 등록 정보의 모든 값을 입력한 후, 다음 중 하나를 수행할 수 있습니다.
   - 변경사항을 취소하고 원래 값을 보존한 상태에서 Connector Configurator를 종료하려면, **파일 > 종료**를 누르고(또는 창을 닫고) 변경사항을 저장할 것인지 묻는 메시지가 표시될 때 **아니오**를 누르십시오.

- Connector Configurator에서 다른 카테고리의 값을 입력하려면 카테고리 탭을 선택하십시오. **표준 등록 정보**(또는 다른 카테고리)에 입력하는 값은 사용자가 다음 카테고리로 이동할 때 보존됩니다. 사용자가 창을 닫을 때 모든 카테고리에 입력한 값을 전체적으로 저장할 것인지 버릴 것인지 묻는 메시지가 표시됩니다.
- 변경된 값을 저장하려면, **파일** > **종료**를 누르고(또는 창을 닫고) 변경사항을 저장할지 묻는 메시지가 표시되면 **예**를 누르십시오. 또는 파일 메뉴나 도구 모음에서 **저장** > **파일에**를 누르십시오.

응용프로그램 특정 구성 등록 정보 설정

응용프로그램 특정 구성 등록 정보의 경우 등록 정보 이름을 추가하거나 변경하고, 값을 구성하고, 등록 정보를 삭제하고, 등록 정보를 암호화할 수 있습니다. 기본 등록 정보 길이는 255자입니다.

1. 격자의 왼쪽 맨 윗부분을 마우스 오른쪽 단추로 누르십시오. 팝업 메뉴 표시줄이 표시됩니다. 등록 정보를 추가하려면 **추가**를 누르십시오. 하위 등록 정보를 추가하려면 상위 행 번호를 마우스 오른쪽 단추로 누른 다음 **하위 추가**를 누르십시오.
2. 등록 정보 또는 하위 등록 정보의 값을 입력하십시오.
3. 등록 정보를 암호화하려면 **암호화** 상자를 선택하십시오.
4. 126 페이지의 『표준 커넥터 등록 정보 설정』에 설명한 대로 변경사항을 저장하거나 취소하십시오.

각 등록 정보에 대해 표시되는 갱신 메소드는 변경된 값을 활성화하기 위해 구성요소 또는 에이전트를 다시 시작해야 하는지 여부를 표시합니다.

중요: 사전 설정된 응용프로그램 특정 커넥터 등록 정보 이름을 변경하면 커넥터가 실패할 수 있습니다. 커넥터를 응용프로그램에 연결하거나 제대로 실행하려면 특정 등록 정보 이름이 필요할 수 있습니다.

커넥터 등록 정보 암호화

응용프로그램 특정 등록 정보는 **등록 정보 편집** 창에서 **암호화** 선택란을 선택하여 암호화할 수 있습니다. 값의 암호를 해독하려면 **암호화** 선택란을 선택 해제하고 **검증** 대화 상자에 올바른 값을 입력한 다음 **확인**을 누르십시오. 입력된 값이 맞는 경우, 값은 암호 해독되고 표시됩니다.

각 커넥터에 대한 어댑터 안내서에는 각 등록 정보 및 해당 기본값의 목록과 설명이 들어 있습니다.

등록 정보의 값이 여러 개일 경우, 등록 정보의 첫 번째 값에 대한 **암호화** 선택란이 나타납니다. **암호화**를 선택하면 모든 등록 정보 값이 암호화됩니다. 등록 정보의 여러 값에 대한 암호를 해독하려면, 등록 정보의 첫 번째 값에 대한 **암호화** 선택란을 선택 해제하고 **검증** 대화 상자에 새 값을 입력하십시오. 입력값이 일치하면, 모든 복수 값이 암호 해독됩니다.

갱신 메소드

100 페이지의 『등록 정보 값 설정 및 갱신』의 부록 *커넥터에 대한 표준 구성 등록 정보*에 있는 갱신 메소드에 대한 설명을 참조하십시오.

지원되는 Business Object 정의 지정

Connector Configurator의 **지원되는 Business Object** 탭을 사용하여 커넥터가 사용할 Business Object를 지정하십시오. 일반 Business Object와 응용프로그램 특정 Business Object를 모두 지정하고, Business Object 사이의 맵에 대한 연관을 지정해야 합니다.

주: 일부 커넥터에서는 해당 응용프로그램으로 이벤트 공고 또는 추가 구성(Meta Object 사용)을 수행하기 위해 특정 Business Object가 지원되도록 지정해야 합니다. 자세한 정보는 *Connector Development Guide for C++* 또는 *Connector Development Guide for Java*를 참조하십시오.

ICS가 브로커인 경우

Business Object 정의가 커넥터에서 지원되도록 지정하거나 기존 Business Object 정의의 지원 설정을 변경하려면, **지원되는 Business Objects** 탭을 누르고 다음 필드를 사용하십시오.

Business Object 이름: System Manager가 실행 중인 커넥터에서 Business Object 정의가 지원되도록 지정하려면 다음을 수행하십시오.

1. **Business Object 이름** 목록의 빈 필드를 누르십시오. System Manager 프로젝트에 존재하는 모든 Business Object 정의를 보여주는 드롭 다운 목록이 표시됩니다.
2. Business Object를 눌러 추가하십시오.
3. Business Object에 대한 **에이전트 지원**(아래에 설명됨)을 설정하십시오.
4. Connector Configurator 창의 파일 메뉴에서 **프로젝트에 저장**을 누르십시오. 추가된 Business Object 정의에 지정된 지원을 포함하여 변경된 커넥터 정의가 System Manager의 프로젝트에 저장됩니다.

지원되는 목록에서 Business Object를 삭제하려면 다음을 수행하십시오.

1. Business Object 필드를 선택하려면 Business Object의 왼쪽에 있는 번호를 누르십시오.
2. Connector Configurator 창의 **편집** 메뉴에서 **행 삭제**를 누르십시오. 목록 화면에서 Business Object는 제거됩니다.
3. **파일** 메뉴에서 **프로젝트에 저장**을 누르십시오.

지원되는 목록에서 Business Object를 삭제하면 커넥터 정의가 변경되고 삭제된 Business Object가 이 커넥터의 이 구현에서는 사용할 수 없게 됩니다. 커넥터 코드에 영향을 주거나 System Manager에서 Business Object 정의 자체를 제거하지는 않습니다.

에이전트 지원: Business Object가 에이전트 지원을 갖는 경우, 시스템은 커넥터 에이전트를 통해 응용프로그램으로 데이터를 전달하기 위해 해당 Business Object를 사용하려 시도합니다.

일반적으로 커넥터의 응용프로그램 특정 Business Object는 해당 커넥터의 에이전트에서 지원되지만 일반 Business Object는 지원되지 않습니다.

커넥터 에이전트가 Business Object를 지원한다는 것을 표시하려면 **에이전트 지원** 상자를 선택하십시오. Connector Configurator 창은 에이전트 지원 선택에 대해 유효성을 검증하지 않습니다.

최대 트랜잭션 레벨: 커넥터의 최대 트랜잭션 레벨은 커넥터가 지원하는 최상위 트랜잭션 레벨입니다.

대부분의 커넥터의 경우 최상의 노력만이 가능한 선택사항입니다.

트랜잭션 레벨 변경 사항을 적용하려면 서버를 다시 시작해야 합니다.

WebSphere 메시지 브로커가 브로커인 경우

독립형 모드(System Manager에 연결되지 않음)로 작업 중인 경우 Business Object 이름을 수동으로 입력해야 합니다.

System Manager가 실행 중이면, **지원되는 Business Objects** 탭의 **Business Object 이름** 열에서 빈 상자를 선택할 수 있습니다. 커넥터가 속하는 통합 구성요소 라이브러리 프로젝트에서 사용 가능한 Business Object 목록이 있는 콤보 상자가 나타납니다. 목록에서 원하는 Business Object를 선택하십시오.

WebSphere Business Integration Message Broker 5.0의 경우에는 **메시지 세트 ID**가 선택적 필드이므로 ID를 제공할 경우 고유하지 않아도 됩니다. 그러나 WebSphere MQ Integrator 및 Integrator Broker 2.1의 경우에는 고유한 **ID**를 제공해야 합니다.

WAS가 브로커인 경우

WebSphere Application Server를 브로커 유형으로 선택할 경우, Connector Configurator는 메시지 세트 ID가 필요하지 않습니다. **지원되는 Business Object** 탭이 지원되는 Business Object에 대해서만 **Business Object 이름** 열을 표시합니다.

독립형 모드(System Manager에 연결되지 않은 채)로 작업 중인 경우 Business Object 이름을 수동으로 입력해야 합니다.

System Manager가 실행 중이면, 지원되는 Business Object 탭의 Business Object 이름 열에서 빈 상자를 선택할 수 있습니다. 커넥터가 속하는 통합 구성요소 라이브러리 프로젝트에서 사용 가능한 Business Object의 목록이 포함된 콤보 상자가 나타납니다. 이 목록에서 원하는 Business Object를 선택하십시오.

연관된 맵(ICS에만 해당)

각 커넥터는 현재 WebSphere InterChange Server에서 활성 상태인 Business Object 정의 및 그와 연관된 맵의 목록을 지원합니다. 이 목록은 **연관된 맵** 탭을 선택할 때 나타납니다.

Business Object 목록에는 에이전트가 지원하는 응용프로그램 특정 Business Object와 제어기가 등록 협업으로 전송하는 해당 일반 오브젝트가 있습니다. 맵의 연관은 응용프로그램 특정 Business Object를 일반 Business Object로 변환하거나 일반 Business Object를 응용프로그램 특정 Business Object로 변환하는 데 사용할 맵을 결정합니다.

특정 소스 및 목적지 Business Object에 고유하게 정의된 맵을 사용 중인 경우, 화면을 열면 맵은 이미 해당 Business Object와 연관되어 있으므로 변경할 필요가(또는 변경할 수) 없습니다.

지원되는 Business Object에서 둘 이상의 맵이 사용 가능한 경우, 사용해야 하는 맵에 Business Object를 명시적으로 바인드해야 합니다.

연관된 맵 탭은 다음 필드를 표시합니다.

- **Business Object 이름**

  지원되는 Business Object 탭에서 지정한 대로, 이 커넥터가 지원하는 Business Object입니다. 지원되는 Business Object 탭 아래에서 추가 Business Object를 지정하면, 해당 Business Object는 Connector Configurator 창의 **파일** 메뉴에서 **프로젝트에 저장**을 선택하여 변경사항을 저장한 후 이 목록에 반영됩니다.

- **연관된 맵**

  커넥터가 지원하는 Business Object에서 사용하기 위해 시스템에 설치된 모든 맵이 화면에 표시됩니다. 각 맵에 대한 소스 Business Object가 **Business Object 이름** 화면에서 맵 이름의 왼쪽에 표시됩니다.

- **명시**

  어떤 경우에는 연관된 맵을 명시적으로 바인드해야 할 수도 있습니다.

  명시적 바인딩은 지원되는 특정 Business Object에 둘 이상의 맵이 존재하는 경우에만 필요합니다. ICS는 시동할 때 자동으로 각 커넥터에 대해 각 지원되는 Business Object에 맵을 바인드하려 시도합니다. 여러 개의 맵이 입력으로 같은 Business Object를 사용할 경우, 서버는 다른 맵의 상위 세트인 하나의 맵을 찾아서 바인드하려고 합니다.

  다른 맵의 상위 세트인 맵이 없을 경우, 서버는 Business Object를 단일 맵에 바인드할 수 없으므로 사용자가 명시적으로 바인딩을 설정해야 합니다.

  맵을 명시적으로 바인드하려면 다음을 수행하십시오.

  1. **명시** 열에서, 바인드하려는 맵에 대한 선택란에 체크 표시를 하십시오.
  2. Business Object와 연관시키려는 맵을 선택하십시오.

3. Connector Configurator 창의 **파일** 메뉴에서 **프로젝트에 저장**을 누르십시오.
4. 프로젝트를 ICS에 전개하십시오.
5. 변경사항이 적용되도록 서버를 다시 시동하십시오.

자원(ICS)

자원 탭에서는 커넥터 에이전트가 커넥터 에이전트 병렬화를 사용하여 동시에 다중 프로세스를 처리할 수 있는지 여부와 그 범위를 판별하는 값을 설정할 수 있습니다.
모든 커넥터가 이 기능을 지원하지는 않습니다. 대개 다중 프로세스보다 다중 스레드를 사용하는 것이 더 효율적이므로 다중 스레드되도록 Java로 설계된 커넥터 에이전트를 실행 중인 경우 이 기능을 사용하지 않는 것이 좋습니다.

메시징(ICS)

메시징 등록 정보는 MQ를 `DeliveryTransport` 표준 등록 정보의 값으로 설정하고 ICS를 브로커 유형으로 설정한 경우에만 사용 가능합니다. 이 등록 정보는 커넥터가 대기열을 사용하는 방법에 영향을 미칩니다.

추적/로그 파일 값 설정

커넥터 구성 파일이나 커넥터 정의 파일을 열 때 Connector Configurator는 해당 파일의 로깅 및 추적 값을 기본값으로 사용합니다. Connector Configurator에서 값을 변경할 수 있습니다.

로깅 및 추적 값을 변경하려면 다음을 수행하십시오.

1. **추적/로그 파일** 탭을 누르십시오.
2. 로깅 또는 추적에 대해 다음 중 하나 또는 모두에 메시지를 기록하도록 선택할 수 있습니다.
   - 콘솔에(STDOUT):
     로깅 또는 추적 메시지를 STDOUT 화면에 기록합니다.

     주: Windows 플랫폼에서 실행 중인 커넥터의 경우 **추적/로그 파일** 탭에서 STDOUT 옵션만을 사용할 수 있습니다.
   - 파일에:
     로깅 또는 추적 메시지를 사용자가 지정하는 파일에 기록합니다. 파일을 지정하려면, 디렉토리 단추(말줄임표)를 누르고 원하는 위치로 이동한 후 파일 이름을 제공하고 **저장**을 누르십시오. 로깅 또는 추적 메시지는 사용자가 지정하는 위치 및 파일에 기록됩니다.

     주: 로깅 및 추적 파일은 단순한 텍스트 파일입니다. 파일 이름을 설정할 때 원하는 파일 확장자를 사용할 수 있습니다. 그러나 추적 파일의 경우, 사용자

시스템에 상주할 수 있는 다른 파일과의 혼동을 피하기 위해 .trc보다는 .trace 확장자를 사용하는 것이 좋습니다. 파일 로깅의 경우에는 .log 및 .txt가 일반적인 파일 확장자입니다.

Data Handler

Data Handler 섹션은 DeliveryTransport의 JMS 값과 ContainerManagedEvents의 JMS 값을 지정한 경우에만 구성에 사용할 수 있습니다. 모든 어댑터가 Data Handler를 사용하지는 않습니다.

이들 등록 정보에 사용할 값에 대해서는 부록 A, 표준 등록 정보의 ContainerManagedEvents 설명을 참조하십시오. 자세한 정보는 *Connector Development Guide for C++* 또는 *Connector Development Guide for Java*를 참조하십시오.

구성 파일 저장

커넥터 구성이 완료되면 커넥터 구성 파일을 저장하십시오. Connector Configurator는 이 파일을 구성 중에 선택한 브로커 모드로 저장합니다. Connector Configurator의 제목 표시줄에는 현재 사용하고 있는 브로커 모드(ICS, WMQI 또는 WAS)가 항상 표시됩니다.

파일은 XML 문서로 저장됩니다. XML 문서를 다음과 같은 세 가지 방식으로 저장할 수 있습니다.

- System Manager에서, 통합 구성요소 라이브러리에 *.con 확장자를 갖는 파일로 저장.
- 지정하는 디렉토리에 저장.
- 독립형 모드에서, 디렉토리 폴더에 *.cfg 확장자를 갖는 파일로 저장.

System Manager의 프로젝트 사용 및 전개에 대한 자세한 정보는 다음 구현 안내서를 참조하십시오.

- ICS: *Implementation Guide for WebSphere InterChange Server*
- WebSphere 메시지 브로커의 경우: *Implementing Adapters with WebSphere Message Brokers*
- WAS: *Implementing Adapters with WebSphere Application Server*

구성 파일 변경

기존 구성 파일의 통합 브로커 설정을 변경할 수 있습니다. 그러면 파일을 템플리트로 사용하여 다른 브로커와 함께 사용할 수 있는 새 구성 파일을 작성할 수 있습니다.

주: 통합 브로커를 전환하는 경우에는 브로커 모드 등록 정보 외에 다른 구성 등록 정보를 변경할 필요가 있습니다.

기존 구성 파일 내에서 브로커 선택을 변경하려면 다음을 수행하십시오(선택적).

- Connector Configurator에서 기존 구성 파일을 여십시오.
- **표준 등록 정보** 탭을 선택하십시오.
- 표준 등록 정보 탭의 **브로커 유형** 필드에서 브로커에 적합한 값을 선택하십시오. 현재 값을 변경할 때, 등록 정보 화면의 사용 가능한 탭 및 필드 선택사항은 즉시 변경되어 사용자가 선택한 새 브로커와 관련된 필드와 탭만을 표시합니다.

구성 완료

커넥터에 대한 구성 파일을 작성하고 수정한 후 커넥터가 시작할 때 해당 구성 파일을 찾을 수 있는지 확인하십시오.

이렇게 하려면 커넥터에 사용되는 시작 파일을 열고 커넥터 구성 파일에 사용되는 위치 및 파일 이름이 사용자가 파일에 부여한 이름 및 사용자가 파일을 저장한 디렉토리 또는 경로와 정확하게 일치하는지 확인하십시오.

국제화된 환경에서 Connector Configurator 사용

Connector Configurator는 국제화되었으며 구성 파일과 통합 브로커 사이의 문자 변환을 처리할 수 있습니다. Connector Configurator는 기본 인코딩을 사용합니다. 구성 파일에 기록할 때 UTF-8 인코딩을 사용합니다.

Connector Configurator는 다음 위치에서 영어 이외의 문자를 지원합니다.

- 모든 값 필드
- 로그 파일 및 추적 파일 경로(**추적/로그 파일** 탭에 지정됨)

`CharacterEncoding` 및 `Locale` 표준 구성 등록 정보에 대한 드롭 목록은 지원되는 값의 서브세트만을 표시합니다. 드롭 목록에 다른 값을 추가하려면 제품 디렉토리에 있는 `\Data\Std\stdConnProps.xml` 파일을 수동으로 수정해야 합니다.

예를 들어, 로케일 `en_GB`를 `Locale` 등록 정보에 대한 값 목록에 추가하려면, `stdConnProps.xml` 파일을 열고 아래의 행을 굵은체 유형으로 추가하십시오,

```
<Property name="Locale" isRequired="true" updateMethod="component restart">
                <ValidType>String</ValidType>
            <ValidValues>
                                    <Value>ja_JP</Value>
                                    <Value>ko_KR</Value>
                                    <Value>zh_CN</Value>
                                    <Value>zh_TW</Value>
                                    <Value>fr_FR</Value>
                                    <Value>de_DE</Value>
                                    <Value>it_IT</Value>
                                    <Value>es_ES</Value>
                                    <Value>pt_BR</Value>
                                    <Value>en_US</Value>
                                    <Value>en_GB</Value>
```

```
                <DefaultValue>en_US</DefaultValue>
        </ValidValues>
</Property>
```

부록 C. 널(null) 및 공백 값 지원

이 부록에서는 Business Object에서 키 값이 공백이거나 널(null)인 여러 가지의 성공 및 실패 시나리오에 대해 자세히 설명합니다. 또한 공백 또는 널 Business Object 값에 필요한 기능적 변경사항도 있습니다.

성공 및 실패 시나리오

Business Object의 키 값이 공백이거나 데이터베이스에 널값을 가지고 있으면 ″=″ 연산자 유형 대신 ″is null″ 유형의 where 절을 빌드하십시오.

IBM에서는 Business Object에 공백 값을 가지고 있지 않은 최소한 하나의 키 속성이 있도록 권장합니다.

다음 시나리오는 널값을 가지고 있는 하나의 키가 있는 상위 오브젝트입니다. 이 시나리오는 다음 조건에서 실패합니다.

표 18. 고객

속성	유형
cid	정수(키)
이름	문자열
주석	문자열

다음 시나리오는 두 개의 키가 있고 하나가 널값을 가진 상위 오브젝트의 경우입니다. 이 시나리오는 다음 조건에서 성공합니다.

표 19. 고객

속성	유형
cid	정수(키)
이름	문자열
주석	문자열

시나리오 2에서 cid=1000이고 이름이 널(null)로 설정된 고객으로부터 cid, 이름 및 주석을 선택하여 검색 조회를 빌드하십시오.

다음 시나리오는 외부 키 참조가 있는 컨테이너 오브젝트에 하나의 하위 오브젝트가 있는 상위 오브젝트입니다. 이 시나리오는 다음 조건에서 실패합니다.

표 20. 고객

속성	유형
cid	정수(키)

표 20. 고객 (계속)

속성	유형
이름	정수(키)
주석	문자열
주소	주소
Aid	Integer (Key) ASI:FK=cid
Acity	String
Azip	String

cid에 널값이 있으면 주소에서 Aid, Acity, Azip을 선택하여 검색 조회를 빌드하십시오. Aid 값을 널로 설정하십시오.

다음 시나리오는 두 개의 키 참조가 있는 컨테이너 오브젝트에 하나의 하위 오브젝트가 있는 상위 오브젝트입니다. 이 시나리오는 다음 조건에서 성공합니다.

표 21. 고객

속성	유형
cid	정수(키)
이름	문자열
주석	문자열
주소	주소
Aid	Integer (Key) ASI:FK=cid
Acity	String (Key) ASI:FK=name
Azip	String

이름이 널값이면 주소에서 Aid, Acity 및 Azip을 선택하여(Aid=Cid, Acity=널값) 조회 검색을 빌드하십시오.

기능

커넥터가 키에서 공백 값을 발견하면 그 값을 속성에서 UseNull 값과 비교합니다. 값이 true이면 커넥터는 널값을 조회에 추가합니다. 이는 다음 Verb 조작에 영향을 줍니다.

- Retrieve
- RetrieveBy Content
- Update
- Delete

주의사항

IBM은 다른 국가에서는 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 언급하는 것이 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상 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 특허에 대한 라이센스까지 부여하는 것은 아닙니다. 라이센스에 대한 의문사항은 다음으로 문의하십시오.

135-270
서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
한국 아이.비.엠 주식회사
고객만족센터
전화번호: 080-023-8080

다음 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 “현상태대로” 제공합니다. 일부 국가에서는 특정 거래에서 명시적 또는 묵시적 보증의 면책사항을 허용하지 않으므로, 이 사항이 적용되지 않을 수도 있습니다.

이 정보에는 기술적으로 부정확한 내용이나 인쇄상의 오류가 있을 수 있습니다. 이 정보는 주기적으로 변경되며, 변경된 사항은 최신판에 통합됩니다. IBM은 이 책에서 설명한 제품 및(또는) 프로그램을 사전 통지없이 언제든지 개선 및(또는) 변경할 수 있습니다.

이 정보에서 언급되는 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 사이트 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 프로그램과 기타 프로그램(본 프로그램 포함) 간의 정보 교환 및 (2) 교환된 정보의 상호 이용을 목적으로 정보를 원하는 프로그램 라이센스 사용자는 다음 주소로 문의하십시오.

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

이러한 정보는 해당 조항 및 조건(예를 들어, 사용료 지불 등)에 따라 사용할 수 있습니다.

이 정보에 기술된 라이센스가 있는 프로그램 및 이 프로그램에 대해 사용 가능한 모든 라이센스가 있는 자료는 IBM이 IBM 기본 계약, IBM 프로그램 라이센스 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

본 문서에 포함된 모든 성능 데이터는 제한된 환경에서 산출된 것입니다. 따라서 다른 운영 환경에서 얻어진 결과는 상당히 다를 수 있습니다. 일부 성능은 개발 레벨 상태의 시스템에서 측정되었을 수 있으므로 이러한 측정치가 일반적으로 사용되고 있는 시스템에서도 동일하게 나타날 것이라고는 보증할 수 없습니다. 또한, 일부 성능은 추정치일 수도 있으므로 실제 결과는 다를 수 있습니다. 이 문서의 사용자는 해당 데이터를 사용자의 특정 환경에서 검증해야 합니다.

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

이 정보에는 일상의 비즈니스 운영에서 사용되는 자료 및 보고서에 대한 예제가 들어 있습니다. 이들 예제에는 개념을 가능한 완벽하게 설명하기 위해 개인, 회사, 상표 및 제품의 이름이 사용될 수 있습니다. 이들 이름은 모두 가공의 것이며 실제 기업의 이름 및 주소와 유사하더라도 이는 전적으로 우연입니다.

IBM의 향후 방향 또는 의도에 관한 모든 언급은 별도의 통지없이 변경될 수 있습니다.

프로그래밍 인터페이스 정보

프로그래밍 인터페이스 정보(제공될 경우)는 이 프로그램을 사용하여 응용프로그램 소프트웨어를 작성하는 것을 돕기 위한 것입니다.

범용 프로그래밍 인터페이스를 사용하면 이 프로그램 도구 서비스를 확보하는 응용프로그램 소프트웨어를 작성할 수 있습니다.

그러나 이 정보는 도한 진단, 수정 및 성능 조정에 대한 정보를 포함할 수 있습니다. 진단, 수정 및 성능 조정에 대한 정보는 사용자의 응용프로그램 소프트웨어를 디버그하는 데 도움을 주기 위해 제공됩니다.

경고: 진단, 수정 및 성능 조정에 대한 정보는 변경될 수 있으므로 프로그래밍 인터페이스로 사용해서는 안됩니다.

상표 및 서비스표

다음 용어는 미국 또는 기타 국가에서 사용되는 IBM Corporation의 상표 또는 등록상표입니다.

IBM
IBM 로고
AIX
CrossWorlds
DB2
DB2 Universal Database
Domino
Lotus
Lotus Notes
MQIntegrator
MQSeries
Tivoli
WebSphere

Microsoft, Windows, Windows NT 및 Windows 로고는 미국 또는 기타 국가에서 사용되는 Microsoft Corporation의 상표입니다.

MMX, Pentium 및 ProShare는 미국 또는 기타 국가에서 사용되는 Intel Corporation의 상표 또는 등록상표입니다.

Java 및 모든 Java 기반의 상표는 Sun Microsystems, Inc.의 상표 또는 등록상표입니다.

기타 회사, 제품 또는 서비스 이름은 해당 회사의 상표 또는 서비스표입니다.

Adapter for Manugistics Demand & Fulfillment Management에는 Eclipse Project(http://www.eclipse.org)가 개발한 소프트웨어가 포함됩니다.

IBM WebSphere InterChange Server V4.2.1, IBM WebSphere Business Integration Toolset V4.2.1, IBM WebSphere Business Integration Adapters V2.3.1, IBM WebSphere Business Integration Collaborations V4.2.

WebSphere Business Integration Adapter Framework V2.4.0.

IBM 한글 지원에 관한 설문

IBM WebSphere Business Integration Adapters
Adapter for Manugistics Demand & Fulfillment Management
사용자 안내서
버전 **1.0.x**

FAX : (02) 3787-0123

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

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.