# 데이타베이스 웹 서비스

*Oracle 백서*
*2002년 11월*

ORACLE®

# 데이타베이스 웹 서비스

# 데이타베이스 웹 서비스

## 서론

웹 서비스는 플랫폼, 언어 또는 데이타 형식에 상관없이 웹에서 애플리케이션 간의 상호 작용을 가능하게 합니다. 소프트웨어 업계에서는 XML, SOAP, WSDL 및 UDDI 등의 이러한 핵심 구성 요소들을 채택하기도 했습니다. 웹 서비스는 일반적으로 중간 계층 애플리케이션 서버에서 구현 및 배포된 서비스를 말합니다. 그러나 이기종 및 연결 해제된 환경에서 웹 서비스 인터페이스를 통해 데이타, 메타 데이타 및 내장 프로시저에 액세스하려는 요구가 점점 증가하고 있습니다. 데이타베이스 웹 서비스(Database Web services)는 웹 서비스에 대한 데이타베이스 중심의 비전을 제시해 주며, 이는 두 가지 방향 즉, 데이타베이스 리소스를 웹 서비스로서 액세스하는 것과 데이타베이스 자체에서 외부 웹 서비스를 사용하는 것으로 작용하고 있습니다. Oracle 데이타베이스를 웹 서비스 제공자로 전환하는 것은 PL/SQL 패키지에 대한 투자를 활용하는 것입니다. 반면 데이타베이스 자체에서 외부 웹 서비스를 사용하는 것과 SQL 엔진과의 통합은 기업 정보 통합을 가능하도록 만들어 줍니다.

이 백서에서는 Oracle9*i*AS(Oracle9*i* Application Server) 및 Oracle Database를 사용하여 Oracle9*i* Database와 특히 PL/SQL 패키지를 웹 서비스 분야에 개방함으로써 얻게 되는 이점에 대해 집중적으로 설명하도록 하겠습니다.

여러분은 PL/SQL 패키지를 J2EE 호환 웹 서비스로서 Oracle9*i* Application Server에 쉽게 게시하는 방법에 대해 배우게 됩니다. 또한 웹 서비스에 보안을 제공하고 테스트하는 방법과 웹 서비스를 호출하는 정적 또는 동적 Java 클라이언트를 작성하는 방법에 대해서도 배우게 됩니다. 이 외에도 SQL 유형의 매핑을 조정하는 일반적인 방법 뿐만 아니라, 기존의 PL/SQL 관련 유형 및 REF CURSOR를 사용하기 위한 배경 지식에 대해서도 설명합니다.

데이타베이스 세션에서 웹 서비스로 콜아웃(call-out)해야 할 경우, PL/SQL 뿐만 아니라 Java로도 이를 쉽게 수행할 있도록 Oracle9*i* Database에서 SOAP 클라이언트 라이브러리를 사용하는 방법을 알려 드립니다. 또한 테이블 함수(Table Function)와 이러한 콜아웃(call-out)의 통합으로 데이타베이스 테이블 또는 뷰인 것처럼 웹 서비스 호출의 결과에 액세스할 수 있습니다. 그리고 예제 웹 서비스 콜아웃(call-out) 애플리케이션에 대해서도 설명합니다.

이 백서에서는 새로운 데이타베이스 웹 서비스 영역에 대한 일반적인 로드맵을 제공하여, SQL 질의, DML 문, XML 연산, Advanced Queues 연산, Streams 연산, 비동기 및 동기 호출을 통한 Java 내장 프로시저와 같은 추가 데이타베이스 기능에 웹 서비스 지원을 매핑합니다.

## 웹 서비스에 대한 간략한 배경 지식

W3C(World Wide Web Consortium)는 웹 서비스를 다음과 같은 속성을 지닌 소프트웨어 애플리케이션 또는 구성 요소로 정의하고 있습니다.

- URI로 식별됩니다.
- 인터페이스와 바인딩을 XML로 설명할 수 있습니다.
- 인터넷 기반 프로토콜을 경유하여 XML을 통해 웹 서비스와 직접 상호 작용할 수 있습니다.

이는 표준 메커니즘과 프로토콜을 기반으로 하기 때문에 웹 서비스는 구현 언어(Java, Managed C++, JScript, Perl, VB.NET, C#, J#), 객체 모델(EJB, COM 등), 플랫폼(J2EE, .NET 등)에 독립적입니다. 애플리케이션 아키텍처를 위해 웹 서비스를 활용하려면, 기존 애플리케이션을 웹 서비스로서 캡슐화, 설명 및 게시할 수 있어야 하고, 반대로 애플리케이션에서 외부 웹 서비스를 사용할 수 있어야 합니다. 이 백서에서는 PL/SQL 또는 Java로 작성된 데이타베이스 중심의 애플리케이션에 대한 두 가지 방향 즉, 이러한 애플리케이션을 웹 서비스로서 외부에서 사용할 수 있는 방법과 이 애플리케이션이 외부 웹 서비스를 활용하는 방법에 대해 차례로 설명하겠습니다.

문서 호출(doc-invocation)에 대한 SOAP, WSDL, UDDI, RPC라는 용어는 웹 서비스 아키텍처를 이해하는 데 있어 핵심적인 요소입니다. 데이타베이스 중심의 사용 사례를 검토하기 전에 이들 용어에 대해 간단히 소개하겠습니다.

- SOAP은 웹 서비스가 사용하는 XML 기반 메시지 프로토콜입니다. 전송 메커니즘(HTTP, FTP, SMTP, JMS 등)은 정해져 있지 않습니다. 그러나 대부분의 웹 서비스는 방화벽 친화적인 HTTP 또는 HTTPS 포맷을 받아 들입니다.
- WSDL(Web services Description Language)은 웹 서비스가 제공하는 매개변수(매개변수 유형 포함)와 연산을 지정하는 XML 문서 포맷입니다. 또한 서비스의 위치, 전송 프로토콜, 호출 스타일을 설명합니다.
- UDDI(Universal Description, Discovery, and Integration)는 웹 서비스를 나열한다는 점을 제외하고는 전화 디렉토리와 비슷한 기능을 합니다. UDDI는 고유 식별자(화이트 페이지), 비즈니스 범주(엘로우 페이지), 서비스 프로토콜을 바인딩하는 방법(그린 페이지)과 같은 웹 서비스에 대한 정보 등록을 허용하는 표준 프로토콜입니다.
- 웹 서비스는 다양한 방법으로 이루어집니다. 디스패치는 동기(일반적으로) 또는 비동기 방식으로 이루어질 수 있고, 호출은 RPC 스타일(인수가 있는 하나의 연산이

전송되고 응답 하나가 반환됨) 또는 메시지 스타일(단방향 SOAP 문서 교환)로 수행될 수 있으며, 다양한 암호화 규칙(해석 가능한 또는 암호화된)이 사용될 수 있습니다. 웹 서비스를 호출할 때, 호출 전에 웹 서비스에 대한 모든 것을 알 수 있거나(정적 호출), 또는 호출하자마자 웹 서비스의 연산 및 전송 엔드포인트에 대해 알 수 있습니다(동적 호출).

다음 그림은 여러 가지 웹 서비스 구성 요소와 이들 구성 요소가 상호 작용하는 방법을 보여 줍니다.

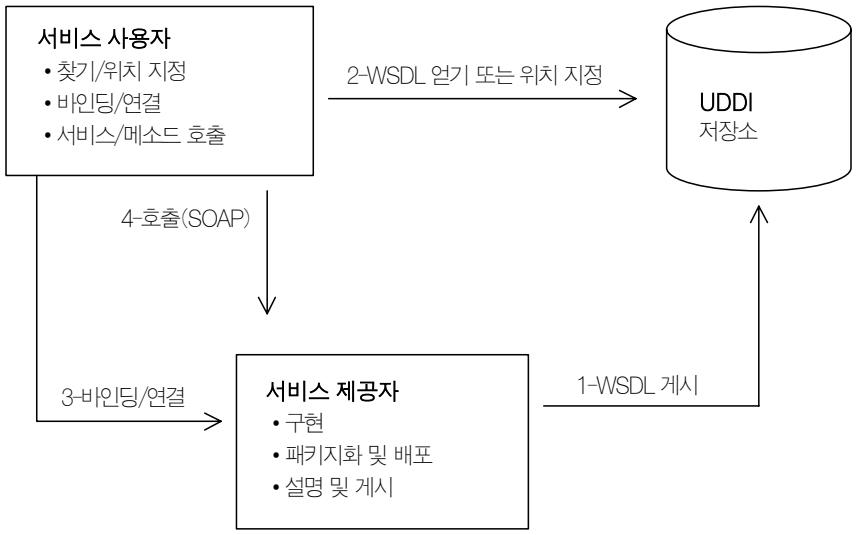



**그림 1 웹 서비스 제공 및 사용**

웹 서비스 호출은 XML 메시지의 교환으로 볼 수 있기 때문에 전송 단계에서 웹 서비스를 액세스 및 제공할 수 있습니다. 그러나 이렇게 하는 것이 권장되는 방법은 아닙니다. SOAP 엔벌로프(envelop), SOAP 헤더 및 구성 요소, 본문에서 사용되는 암호화 포맷 등을 비롯한 SOAP 포맷을 완벽하게 이해해야 하기 때문입니다. 또한 UDDI 서버와 상호 작용하는 방법을 이해해야 할 뿐만 아니라 모든 종류의 WSDL 설명을 구문 분석(파싱) 및 해석해야 합니다.

자체 SOAP 라이브러리 및 도구를 작성 및 유지 관리하는 것보다 기존 및 표준 SOAP

라이브러리 및 도구를 사용하는 것이 좋습니다. 데이타베이스에 대한 웹 서비스 콜인(call-in)을 위해서는 Oracle9*i*AS가 제공하는 J2EE 웹 서비스 환경을 사용할 수 있습니다. 또한 데이타베이스에서 외부 웹 서비스를 호출할 경우에도 기존 SOAP 클라이언트 라이브러리를 재사용할 수 있습니다. 이 백서에서는 이 두 가지 방향 즉, 데이타베이스에 대한 웹 서비스 콜인(call-in)과 데이타베이스에서 외부 웹 서비스를 호출하는 것에 대해 차례로 설명합니다.

## 데이타베이스를 웹 서비스 제공자로 활용

### 데이타베이스 웹 서비스 콜인(call-in)의 이점

웹 서비스 요청에 의한 데이타베이스 연산 트리거링은 이기종 및 연결 해제된 기업 인트라넷에서 데이타베이스를 개방하고 데이타, 데이타 로직, 메타 데이타, 비즈니스 엔티티를 공유하는 표준의 안전하고 제어된 방법을 제공합니다.

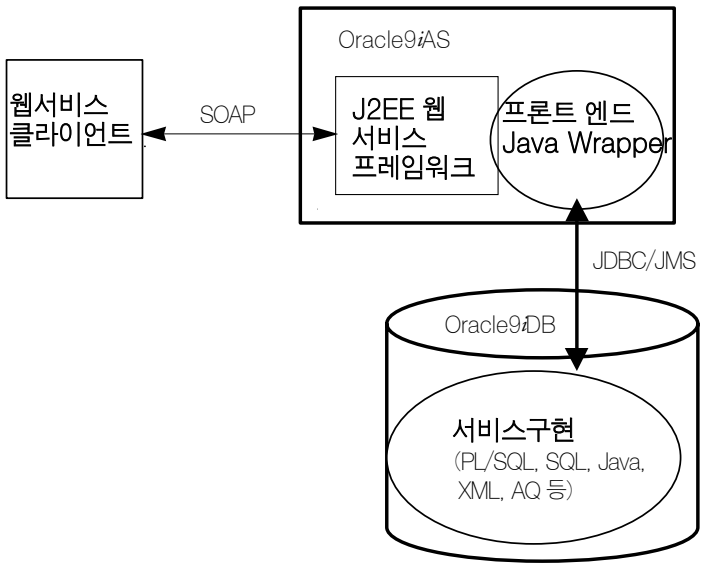
데이타베이스 웹 서비스 콜인(call-in)은 다음 사항을 허용합니다.

- 데이타베이스에서 실행되는 기존 또는 향후 PL/SQL 내장 프로시저 및 Java 클래스를 활용.
- SQL 및 XQuery를 통한 웹 서비스로서의 XML 문서 저장 및 검색. 예를 들어, 데스크톱 애플리케이션은 XML 문서를 다운로드, 편집 및 업로드할 수 있습니다. 또한 애플리케이션은 감사 및 로깅 정보를 표시하는 XML 문서의 저장을 위해 데이터베이스로 전송하고 검색할 수 있습니다.
- 사전 정의된 SQL 질의, DML 및 XQuery 연산 등을 통해 웹 서비스로서의 관계형, 객체, 텍스트, 공간, 멀티미디어 데이타 저장 및 검색 제공. 이러한 웹 서비스의 예로는 비즈니스 인텔리전스 서비스, 카탈로그 검색 서비스, Map/GIS 서비스가 있습니다.
- 데이타베이스 대기 및 메시징 연산을 웹 서비스 엔드포인트로서 제공.

### 아키텍처

Oracle9*i* Database는 SQL, PL/SQL 및 Java 내장 프로시저, XDB(XML Database)의 XML 기능, AQ(Advanced Queuing) 및 Streams와 같은 다양한 리소스 및 기능을 제공합니다. Java 클라이언트는 일반적으로 Oracle JDBC 드라이버와 JMS(Java Messaging Service)를 통해 이러한 기능에 액세스합니다. 오라클은 한 걸음 더 나아가 웹 서비스를

통해 이러한 기능을 제공하는 것을 목표로 하고 있습니다. 확장성 있는 표준 J2EE 컨테이너와 함께 완벽한 J2EE 웹 서비스 프레임워크를 제공하는 Oracle9*i* Application Server를 기본으로 사용합니다.




데이타베이스 웹 서비스의 개발, 배포 및 관리를 위해 오라클은 Oracle9*i* JDeveloper, Enterprise Manager 및 Web Services Assembler 등 J2EE 애플리케이션에 사용되는 것과 동일한 완벽한 통합 툴 세트를 제공합니다.

### 데이타베이스 웹 서비스의 게시

Oracle9*i*AS를 통해 Java 클래스, PL/SQL 내장 프로시저 및 J2EE 구성 요소를 기반으로 한 웹 서비스를 생성할 수 있습니다. 이 단원은 PL/SQL 패키지를 게시하는 방법에 초점을 두고 있으며, 오라클의 툴이 투명하게 처리할 수 있는 상세한 내용을 개발자에게 제공합니다. 여러분이 SCOTT 스키마에서 사용할 수 있는 MYAPP 패키지를 가지고 있다고 가정하고, 이 논의에서는 웹 서비스를 OC4J(Oracle9*i*AS Containers for J2EE)에 배포된 J2EE 호환 EAR 파일로서 생성하기 위해 Oracle9*i*AS 9.0.3 이후 버전에서 사용할 수 있는 웹 서비스 어셈블러(Web services Assembler) 도구를 사용합니다. ant 기반 도구를 활용하는 이전 릴리스에도 이와 유사한 기능이 있습니다.

### 웹 서비스 어셈블러의 구성 파일

웹 서비스 어셈블러(Web Service Assembler)에 대한 입력은 구성 파일 config.xml을 통해 제공됩니다. MYAPP을 기반으로 한 웹 서비스의 구성 파일은 다음과 같습니다.

```
<web-service>

<display-name>Web Service using MYAPP</display-name>

<description>Exposing the PL/SQL package MYAPP

as a Web Service under the endpoint: /oow/MyApp

</description>

<destination-path>./myapp.ear</destination-path>

<temporary-directory>./tmp</temporary-directory>

<context>/oow</context>

<!-- A stateless service based on a PL/SQL package. -->

<stateless-stored-procedure-java-service>

<!-- The URL under the context /oow for the service -->

<uri>/MyApp</uri>

<!-- Info needed at publishing time -->

<jar-generation>

<!-- Connection information -->

<schema>scott/tiger</schema>

<db-url>jdbc:oracle:thin:@localhost:1521:orcl</db-url>

<!-- PL/SQL package information. By default, this is

also used for the Java class name -->

<db-pkg-name>MyApp</db-pkg-name>

<!-- Java package information -->
```

```
<prefix>mypack</prefix>

</jar-generation>

<!-- Info needed at runtime - the JNDI DB connection -->

<database-JNDI-name>jdbc/OracleDS</database-JNDI-name>

</stateless-stored-procedure-java-service>

</web-service>
```

〈schema〉 및 〈db-url〉의 연결 정보는 코드 생성 시에만 사용된다는 점을 기억하시기 바랍니다. Oracle9*i*AS에서 배포된 웹 서비스를 호출하는 경우, 〈database-JNDI-name〉에 지정되어 있는 JNDI 데이타 소스 jdbc/OracleDS는 데이타베이스로의 연결에 사용됩니다. 따라서 이 데이타 소스가 정의되어 있고 MYAPP 패키지에 액세스할 수 있다는 점을 확인하시기 바랍니다.

**웹 서비스의 배포 및 테스트**

이 구성 파일을 사용하여 웹 서비스 어셈블러 도구(Web services Assembler)를 실행하고, 배포 가능한 EAR 파일 myapp.ear를 생성할 수 있습니다.

```
java -jar WebServicesAssembler.jar -config config.xml
```

생성된 myapp.ear 파일을 Enterprise Manager나 DCM 제어 도구를 사용하여 OC4J에 배포하였으면(또는 admin.jar 도구를 사용하여 독립형 OC4J 설치에 배포) 여러분은 이제 서비스를 테스트할 준비가 된 것입니다. Oracle9*i*AS 설치에 맞게 호스트 및 포트 설정을 사용하여 다음 주소에 브라우저를 연결합니다.

```
http://localhost:8888/oow/MyApp
```

여러분은 웹 서비스 연산 목록을 보게 되고, 브라우저를 통해 연산을 각각 테스트할 수 있습니다. 또한 엔드포인트 /oow/MyApp?WSDL에서 서비스의 WSDL 사양에 액세스할 수 있고, /oow/MyApp?proxy_jar에서 클라이언트 애플리케이션에 통합될 수 있는 Java 클라이언트 클래스를 얻어 웹 서비스를 호출할 수 있습니다.

더욱 자세한 배경 지식에 대해서는 *Oracle9iAS Web Services* 개발자 가이드 제5장 "내장 프로시저 웹 서비스의 개발 및 배포"를 참조하십시오.

### JDeveloper를 사용하여 PL/SQL 웹 서비스를 배포하는 방법

Oracle9*i* JDeveloper 9.0.3 및 그 이후 버전에서는 웹 서비스를 게시하는 데 필요한 모든 단계를 자동화하고 있습니다. 여러분은 Connections에서 필요한 데이타베이스 스키마로 이동하여 Packages 노드를 확장하기만 하면 됩니다. PL/SQL 패키지를 마우스 오른쪽 단추로 누른 다음 Publish as Web Service를 선택하여 PL/SQL 웹 서비스 마법사를 시작합니다. 마법사는 웹 서비스 게시 단계를안내합니다. 또한 http://otn.oracle.com/tech/webservices/htdocs/series/plsql/content.html에서도 JDeveloper를 사용하여 PL/SQL 웹 서비스를 게시하는 방법에 관한 자습서를 볼 수 있습니다.

### Oracle Enterprise Manager를 사용한 웹 서비스 보안 및 UDDI 게시

Oracle Enterprise Manager를 사용하면 OC4J에서 실행되는 모든 기타 J2EE 구성 요소와 동일한 방법으로 웹 서비스에 보안을 제공할 수 있습니다. PL/SQL 웹 서비스에 표준 기반 암호화, 인증 및 권한 부여를 사용할 수 있고, 싱글 사인 온을 사용하여 보안의 모든 측면을 중앙 집중적으로 관리할 수 있습니다.

또한 Oracle Enterprise Manager를 통해 웹 서비스를 Oracle UDDI Registry 인스턴스에 게시할 수 있습니다. 애플리케이션 배포 마법사(Deploy Application Wizard)를 사용하여 웹 서비스 EAR 파일을 OC4J에 배포하는 경우, 서비스(서블릿)를 선택하여 UDDI에 게시하고, 필요한 분류 정보를 제공하며, UDDI Registry Web Services Details 창을 통해 자세한 상세 내역을 추가할 수 있습니다.

## 유형 지원 및 매핑 융통성

웹 서비스 어셈블러(Web Services Assembler) 도구 또는 JDeveloper를 통해 생성된 PL/SQL 기반 데이타베이스 웹 서비스가 여러분의 요구를 충족시킨다면 그걸로 충분합니다. 그러나 LOB 유형, XMLTYPE, REF CURSOR, OUT 및 IN OUT 인수에 대한 지원이 부족한 Oracle9*i*AS Release 9.0.3을 사용하는 경우에는 1가지 이상의 제한 사항에 직면할 수 있습니다. 향후 릴리스에서 이러한 모든 제한 사항이 처리되기는 하겠지만, 다음 단원에서는 PL/SQL 메소드 및 SQL 유형을 웹 서비스로서 제공하는 방법으로 완벽한 융통성을 얻을 수 있는 대안 접근 방법에 대해 설명하겠습니다. 그렇게 하려면 PL/SQL 패키지를 기반으로 한 웹 서비스를 생성하는 경우 어떤 일이 발생하는 지에 대해 살펴보아야 합니다.

### Oracle JPublisher: PL/SQL을 Java에 매핑

Oracle JPublisher는 데이타베이스로부터 PL/SQL 패키지 사양을 얻어 이 패키지의 메소드를 호출하기 위한 Java 클래스(실제로는 SQLJ 소드 코드)를 생성합니다. JPublisher를 실행하려면 명령행에 jpub을 입력합니다. 예에서는 이 클래스를 직접 얻기 위해 다음 명령을 실행 했습니다.

```
jpub - sql=MYAPP:MyApp -package=mypack -user=scott/tiger \
     -url=jdbc:oracle:thin:@localhost:1521:orcl
sqlj -d $HOME/classes MyApp.sqlj
```

-sql 및 -package 옵션은 MYAPP 패키지가 Java 클래스 mypack.MyApp로서 게시되어야 한다는 것을 JPublisher에 알려 줍니다. 여러분은 구성 파일의 〈schema〉 및 〈db-url〉 태그를 기억하고 계십니까? 여기서 여러분은 이들 태그가 JPublisher 옵션 - user 및 url에서 사용되는 방법을 알 수 있습니다. sqlj 명령행은 생성된 SQLJ 파일을 번역하고 컴파일된 CLASS 파일을 $HOME/classes 아래에 둡니다. 다음 단원에서는 JPublisher 코드 생성의 첫 번째 단계를 제어하는 방법에 대해 설명합니다.

### REF CURSOR 인수를 지원하기 위해 생성된 코드를 하위 클래스화하려면

생성된 코드의 동작을 수정하려면 JPublisher가 사용자가 제공한 하위 클래스를 활용하는 코드를 생성하도록 해야 합니다. 예에서는 JPublisher의 sql 옵션을 다음과 같이 변경합니다.

```
jpub -sql=MYAPP:MyAppBase:MyApp ...
```

MyAppBase의 기존 메소드를 무효화하는 여러분 고유의 코드를 둘 수 있는 경우, JPublisher는 코드를 MyAppBase.sqlj에 두고 MyApp.sqlj의 첫 번째 버전을 생성합니다. 이는 여러분이 REF CURSOR를 반환하는 PL/SQL 메소드를 가지고 있는 경우에 유용합니다. JPublisher는 자동으로 반환 유형을 java.sql.ResultSet에 매핑하지만 이 ResultSet 유형은 웹 서비스 매개변수로서 게시될 수 없습니다. 이제 여러분은 MyAppBase 클래스에 다음 메소드를 가지게 됩니다.

```
public ResultSet getRefCursor(String arg1, Integer arg2)
```

MyApp에 다음과 같이 웹 서비스가 지원하는 포맷으로 결과 집합을 반환할 수 있는 새로운 메소드를 둡니다.

```
public String[] readRefCursorArray(String arg1, Integer arg2)

{ java.sql.ResultSet rs = getRefCursor(arg1,arg2);

... create a String[] from rs and return it ... }
```

이제 여러분은 getRefCursor 메소드가 웹 서비스 연산에서 삭제되고 readRefCursorArray 메소드가 대신 포함되어 있다는 것을 확인해야 합니다. 이를 수행하려면 여러분이 게시하고 싶은 메소드만을 정확하게 포함하고 있는 인터페이스, 가령 MyAppInterf를 정의합니다. JPublisher를 통해 다음 -sql 옵션 설정을 사용함으로써 이 작업을 간단하게 할 수 있다는 점을 기억하시기 바랍니다.

```
jpub -sql =MYAPP:MyAppBase:MyApp#MyAppInterf ...
```

이 옵션은 JPublisher가 MyAppInterf의 첫 번째 버전을 추가로 생성하도록 요청합니다. 메소드 선언을 적절하게 추가 또는 삭제합니다. 애플리케이션을 위한 코드를 $HOME/classes에 번역한 후 모든 클래스 파일을 하나의 Java 아카이브 파일(예: myapp_pregenerated.jar)로 아카이브합니다. 이제 다시 웹 서비스 어셈블러(Web Services Assembler)를 사용하여 배포할 수 있는 웹 서비스 EAR 파일을 생성할 차례입니다.

**웹 서비스 어셈블러를 통한 인터페이스 및 미리 생성된 Java 아카이브 파일 사용 방법**

이미 Java 아카이브 파일을 생성했기 때문에 이번에는 웹 서비스 어셈블러(Web Services Assembler)가 Java 아카이브 파일을 생성하도록 요청할 필요가 없습니다. 따라서 config.xml 파일에서 〈jar-generation〉 부분을 제거합니다. 대신 이 부분에 다음의 세 가지 새로운 태그를 정의해야 합니다.

- `<class-name>mypack.MyApp</class-name>` 웹 서비스를 구현하는 클래스의 이름
- `<interface-name>mypack.MyAppInterf</interface-name>` 제공될 메소드를 지정하는 인터페이스의 이름
- `<java-resource>myapp_pregenerated.jar</java-resource>` 방금 아카이브한 Java 아카이브 파일

다음으로 이전의 웹 서비스를 배포 및 테스트할 때와 동일한 단계를 수행합니다.

메소드마다 이 접근 방법을 사용하여 지원되지 않는 인수를 웹 서비스가 이해하는 인수로 매핑하거나, 또는 반환 및 OUT(또는 IN OUT) 인수를 둘 다 메소드가 반환하는 결합된 JavaBean 구조로 매핑합니다. 그러나 데이타베이스의 PL/SQL 함수를 통해 수행될 수 있는 PL/SQL(또는 SQL) 유형에서 또 다른 SQL 유형으로의 매핑으로 여러분의 요구사항을 표현할 수 있는 경우에는, 인수 변환에 JPublisher의 유형 매핑 기능을 활용할 수 있습니다.

### SQL 변환 함수를 사용하여 JPublisher로 코드 사용자 정의

예를 들어 PL/SQL BOOLEAN 인수를 사용해 봅시다. JDBC는 이 유형을 지원하지 않습니다. 이러한 인수가 있는 메소드를 어떻게 웹 서비스로 제공할 수 있을까요? 여러분은 BOOLEAN을 INTEGER와 같이 웹 서비스(Java 포맷으로) 뿐만 아니라 JDBC가 이해하는 또 다른 SQL 유형으로 매핑해야 합니다. 다음과 같이 PL/SQL로 된 두 가지 변환 함수를 제공해야 합니다. 하나는 BOOLEAN을 INTEGER로 매핑하고, 하나는 INTEGER를 BOOLEAN으로 매핑합니다.

```
FUNCTION INT2BOOL(i INTEGER) RETURN BOOLEAN;
FUNCTION BOOL2INT(b BOOLEAN) RETURN INTEGER;
```

이제 생성된 코드에서 SQL BOOLEAN 유형이 Java Boolean으로 매핑되었다고 JPublisher에 알려야 합니다. 그 대신 실제로 데이타베이스에서 SQL INTEGER 유형을 주고 받을 수 있습니다

(이는 Java Boolean이 SQL 수치를 받을 수 있기 때문에 JDBC에 대해 작동합니다). 또한 서버에서 INT2BOOL 및 BOOL2INT 함수를 사용하여 이 두 가지 SQL 유형 간의 변환을 수행합니다. 이를 다음 명령 행 설정을 사용하여 지정합니다.

```
jpub -addtypemap=BOOLEAN:boolean:INTEGER:INT2BOOL:BOOL2INT ...
```

이 특정 변환은 실제로는 이미 JPublisher에 프로그래밍되어 있습니다(이 변환 함수를 찾을 수 없는 경우 데이타베이스에 SQL 파일 〔Oracle Home〕/sqlj/lib/sqljutl.sql이 설치되어 있는지 확인합니다). 그러나 이 설명에서는 이러한 유형 변환을 통해 얻게 되는 개념을 전달할 것입니다. 또한 변환할 수 있는 유형이 내장 프로시저 또는 함수의 OUT 또는 IN OUT 인수로서 발생하는 경우, 여러분이 생성된 코드를 사용할 수 있게 되기 전에 데이타베이스에 로드해야 하는 SQL 파일에 JPublisher가 추가 PL/SQL 래퍼 코드를 생성한다는 점을 기억하시기 바랍니다.

두 가지 접근 방법 모두 데이타베이스 웹 서비스 콜인(call-in)을 위한 적절한 기능을 사용할 수 있도록 도와 줍니다. 여러분은 미리 생성된(사용자가 작성했음을 의미) Java 아카이브 파일의 기능을 통해 데이타베이스 웹 서비스가 런타임에서 필요로 하는 것이 데이타베이스 연결 뿐인 동안에는 어떤 데이타베이스 웹 서비스라도 이해할 수 있음을 이미 알아챘을 것입니다. 여러분이 해야 할 일은 Connection 객체와 인스턴스화될 수 있는 Java 클래스를 작성하는 것 뿐입니다.

이제 데이타베이스에 콜인(call-in)을 만드는 웹 서비스의 생성 방법을 알고 있으므로, 내장 프로시저의 일부로서 데이타베이스 내부에서 실제로 실행되는 코드를 살펴보고 웹 서비스로 콜아웃(call-out)해 봅시다. 이것은 어떻게 수행될까요?

## 통합 정보 데이타베이스에서 외부 웹 서비스 사용

앞 단원에서는 웹 서비스를 통해 사용할 수 있고 액세스할 수 있는 데이타베이스 리소스를 생성하는 방법에 대해 설명했습니다. 이 단원에서는 외부 웹 서비스를 SQL 데이타 소스로 전환하기 위한 지시 사항과 이러한 전환의 이점에 대해 설명합니다.

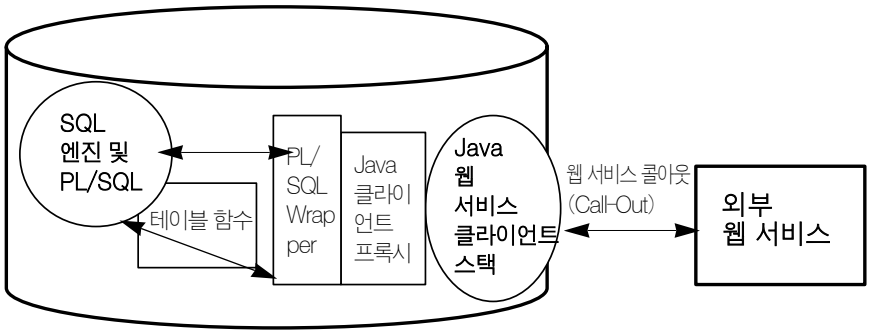
### 데이타베이스 웹 서비스 콜아웃(call-out)의 이점

빠르고 확장성 있는 단일 관계형, XML, 텍스트, 멀티미디어 데이타 저장소의 이점 외에도, 웹 서비스를 비롯한 여러 소스의 데이타를 결합하고 하나의 통합 데이타베이스에 있는 것처럼 결과 데이타를 질의하려는 요구가 점점 증가하고 있습니다. SQL의 파워와 이기종 데이타에 대한 통합 뷰를 결합한 여러 사용 사례가 있습니다. 예를 들어 주가의 추적 및 마이닝, 과학 데이타(예: 게놈 데이타)의 갱신 및 마이닝, IRS 세액표에 대한 질의 및 갱신, 날씨 정보의 추적 및 마이닝(예제 애플리케이션 데모 참조)은 새로운 기업 정보 통합을 위한 빌딩 블록입니다.

또한 데이타는 외부 웹 서비스 호출을 발생시킬 수 있습니다. 여러분은 데이타베이스 테이블 내에서 추적된 재고 상태, 주문 상태 등의 비즈니스 정보를 상태 기반 계산에 활용할 수 있습니다. 몇 가지 예로는 자동화된 주문 추적 및 처리, 배송 추적, 웹 쇼핑 처리가 있습니다 (샘플 애플리케이션 데모 참조).

### 아키텍처

외부 웹 서비스를 사용하려면 이러한 서비스로 콜아웃(call-out)할 수 있어야 합니다. 이 단원은 웹 서비스 데이타와 SQL 엔진의 통합 및 데이타베이스 기능 확장에 대한 개요입니다.




앞에서 언급했듯이 가장 기본 단계에서 여러분은 HTTP 연결을 통해 직접 웹 서비스를 호출할 수 있습니다. PL/SQL에서 HTTP 콜아웃(call-out)을 사용하는 이 기술에 대한 예는 http://otn.oracle.com/tech/webservices/htdocs/samples/dbwebservice/DBWebServic

es_PLSQL.html에서 볼 수 있습니다. 이 접근 방법은 여러분이 미리 웹 서비스와 이의 포맷을  알고 있는 경우, 스스로 서비스의 WSDL 사양을 읽고 해석하려는 경우에는 적합합니다. 보다 일반적인 해결책은 WSDL을 이해하는 기존 SOAP 스택을 사용하는 것입니다. 이를 통해 여러분은 웹 서비스와 동적으로 상호 작용하고 미리 생성된 클라이언트-프록시 코드를 활용할 수 있습니다. 클라이언트-프록시는 웹 서비스 호출을 단순화합니다. 즉, 프록시 인스턴스를 생성하면 프록시 인스턴스에서 필요한 웹 서비스 연산을 호출할 수 있습니다.

### Oracle Database에 SOAP 클라이언트 설치

이제 해야 할 일은 SOAP를 위한 Java 클라이언트를 얻어 이를 OracleJVM에 설치하여 데이타베이스 Java 코드에서 사용할 수 있도록 하는 것입니다. 다음 지시 사항은 J2SE 1.3.x 호환 OracleJVM이 포함된 Oracle9i Database Release 2에 대해 성공적으로 테스트되었으며, J2SE 1.2.x 호환 OracleJVM이 포함된 이 데이타베이스의 이전 릴리스와도 작동합니다.

현재 Oracle SOAP 클래스(OC4J 다운로드에서 사용할 수 있음)를 사용하려면 다음 명령을 실행합니다.

```
loadjava -thin -user sys/<sys-passwd>@<host>:<port>:<SID> \
-resolve -synonym -verbose -grant public \
[OC4J_HOME]/soap/lib/soap.jar [OC4J_HOME]/lib/dms.jar \
[OC4J_HOME]/jlib/javax-ssl-1_1.jar \
[ORACLE_HOME]/lib/servlet.jar \
[OC4J_HOME]/jdk/jre/lib/ext/mail.jar
```

이 클래스들을 SYS 스키마에 로드하고 있는 것입니다. synonym 및 -grant 공용 옵션을 사용하면 다른 스키마에서도 이들을  실행할 뿐만 아니라 사용할  수도 있습니다. 또한 콜아웃(call-out)을 생성할 모든 사용자에게 네트워킹 소켓을 사용할 수 있는 사용 권한을 부여해야 합니다. 호스트와 포트에 '*'을 지정하면  연결에 대한 제한이 없는 액세스를 제공하게 됩니다.

```
execute dbms_java.grant_permission
('<db-user>','SYS:java.net.SocketPermission',
'<host>:<port>','connect,resolve');
```

Oracle SOAP 대신에 JAX-RPC 클라이언트를 사용하고 싶을 수 있습니다. 이러한 경우에는 다음과 같이 JAX-RPC 참조 구현의 Java 아카이브 파일을 로드합니다.
activation, commons-logging, dom, dom4j, jaxp-api, jaxrpc-api, jaxrpc-ri, mail, saaj-api, saaj-ri, and sax.

Oracle 데이타베이스의 통합된 Java VM의 이점은 SOAP 클라이언트와 같은 Java 기반

라이브러리를 로드함으로써 간단히 데이타베이스 기능을 확장할 수 있다는 것입니다. 또한 웹 서비스 인프라의 빠른 속도에 뒤떨어지지 않기 위해 SOAP 스택은 새로운 버전을 로드함으로써 간단히 업데이트될 수 있습니다.

### Java로 생성된 정적 및 동적 웹 서비스 클라이언트

서버측 Java 코드에서 웹 서비스를 호출하려는 경우 여러분은 이미 이를 수행했습니다. 웹 서비스 클라이언트의 동적 호출 인터페이스에 대해 프로그래밍하거나, 또는 정적 클라이언트-프록시 Java 아카이브 파일을 얻고, 로드하며, 사용합니다. 두 가지 경우 모두에서 동일한 SOAP 라이브러리를 사용하여 독립형 Java 클라이언트를 위해 작성하는 것과 동일한 Java 코드를 사용(및 배포)할 수 있습니다. JDeveloper를 사용하여 Oracle SOAP을 위한 클라이언트 프록시를 생성하고, JAX-RPC를 위해 wscompile 도구를 사용할 수 있다는 점을 기억하시기 바랍니다. 이제 우리는 Java 클라이언트를 갖게 되었습니다. PL/SQL의 경우는 어떨까요?

#### PL/SQL로 생성된 웹 서비스 클라이언트

정적 Java 메소드를 PL/SQL 내장 프로시저 및 함수로서 게시할 수 있다는 것을 알고 있습니다. 그러나 웹 서비스 연산은 클라이언트-프록시 클래스의 인스턴스 메소드로 제공됩니다.

이는 웹 서비스 연산을 Java 클래스의 정적 메소드로서 제공할 수 있도록 몇 가지 간단한 Java 연결 코드를 제공해야 한다는 것을 의미합니다.

그 다음은 이 연산을 PL/SQL 호출 사양과 함께 게시할 수 있습니다. 궁극적인 결과는 SQL 최상위 레벨에서 뿐만 아니라 PL/SQL 코드에서도 다른 표준 PL/SQL 프로시저처럼 웹 서비스를 호출할 수 있다는 것입니다. 비록 지금은 여러분이 이러한 수동 변환을 수행해야 하지만 이 작업은 틀림없이 곧 자동화될 것입니다.

### 테이블 함수를 사용하여 웹 서비스와 SQL 엔진을 통합하는 방법

데이타 중심적인 관점으로 봤을 때 우리는 개별적인 웹 서비스 콜아웃(call-out)에만 관심이 있는 것이 아니라, 특정 범위의 입력값에서 웹 서비스 함수의 그래프를 SQL 테이블 또는 뷰 구조로서 구체화하는 데 더 큰 관심이 있습니다. 이를 통해 우리는 이기종 소스로부터 데이타베이스의 중심이라고 할 수 있는 SQL 엔진까지의 웹 서비스 정보를 완벽하게 통합할 수 있습니다.

이를 수행하려면 특정 범위의 입력값에서 테이블 행을 구체화할 수 있는 테이블 함수를 편입시켜야 합니다. 테이블 함수 결과는 일반적으로 파이프 라인으로 나타납니다. 즉, 반환된 행은 중간 스테이지 처리 없이 사용 프로세스에 직접 스트림됩니다. 기본 웹 서비스

함수가 최종적이라면 이는 테이블 함수에서도 사용될 수 있습니다.

다음 예는 다음 함수처럼 PL/SQL을 통해 제공된 웹 서비스에 기초를 둔 테이블 함수의 생성 방법을 보여 줍니다.

FUNCTION CALL_WS(s VARCHAR2) RETURN VARCHAR2

웹 서비스의 그래프와 그 자체가 이러한 테이블 행을 파이프 라인 방식으로 반환하는 테이블 함수를 유지하려면 SQL 유형을 정의해야 합니다.

```
CREATE TYPE WS_ROW_T -- hold graph of Web Service function
    AS OBJECT (res VARCHAR2(255), s VARCHAR2(255));
CREATE FUNCTION WS_TABFUN(cur SYS_REFCURSOR) -- table function
    RETURN WS_ROW PIPELINED AS
s VARCHAR2(255);
BEGIN LOOP FETCH cur INTO s;
           EXIT WHEN cur%NOTFOUND;
           PIPE ROW(WS_ROW_T(s,CALL_WS(s)));
      END LOOP;
      CLOSE cur; RETURN;
END WS_TABFUN;
```

이제 테이블 함수를 가상 테이블로 사용할 수 있습니다. 아래의 코드에서 내부의 SELECT는 CALL_WS 웹 서비스 콜아웃(call-out)을 호출하기 위한 인수로서 사용되는 열을 가진 행을 생성합니다. 테이블 표현식은 뷰 생성 등을 위한 다른 SQL 질의에서 사용될 수 있습니다.

```
SELECT <some-columns> FROM
   TABLE(WS_TABFUN(CURSOR(SELECT s FROM <some_table>))),
   ... WHERE ...
```

### 웹 서비스 콜아웃(call-out) 데모 예제 애플리케이션

다음 데모는 테이블 함수 뿐만 아니라 직접 호출을 사용하여 Oracle9*i* Database 내에서 외부 웹 서비스를 사용하는 방법에 대해 설명합니다.

#### 데이타베이스에서 Java로 외부 웹 서비스 호출 자동화된 주문 처리

이 예제는 구매 주문 처리 시나리오를 구현합니다.

(i) 사용자는 온라인으로 제품 카탈로그를 살펴보고 구매합니다.

(ii) 사용자가 제공하는 로그인 정보를 기초로 하여, Credit Agency 웹 서비스(고객과 고객의 신용 승인 정보가 들어 있는 저장소를 유지 관리함)는 관심 있는 제품을 구입하려는 사용자를 인증 및 승인합니다. 주문 상태는 'PENDING'으로 저장되고 처리되며 이후에 배송됩니다.

(iii) 데이타베이스 일괄 처리 작업은 매 6분마다 실행되도록 계획되고(데모 목적을 위해), 고객 정보, 승인 ID(이전에 획득), 정확한 배송량(처음 주문량과는 다를 수 있음)으로 Credit Agency 웹 서비스를 콜아웃(call-out)함으로써 "PENDING" 주문을 처리합니다.

Credit Agency 웹 서비스에 대한 지불 요청 응답이 okay이면 주문 상태가 "IN_SHIPPING"에서 "SHIPPED"로 변경됩니다. 이는 이러한 특정 주문 라이프 사이클의 끝을 표시하는 것입니다. 사용자는 주문 상태를 언제라도 볼 수 있습니다.

전체 데모는 otn.oracle.com/sample_code/tech/java/jsp/samples/wsclient/Readme.html에서 볼 수 있습니다.

외부 웹 서비스를 SQL 데이타 소스로 변환 날씨 기온 추적 및 마이닝이 예제는 일부 미국 도시의 기온 보고를 마이닝하려는 최종 사용자가 사용할 수 있는 날씨 보고 시스템을 구현합니다.

(i) 이 애플리케이션은 http://services.xmethods.com/ve2/ViewListing.po;jsessionid=hnXezdkbZATXgSu-vQtcxPuM(QhxieSRM)?serviceid=8의 공공 웹 서비스를 통해 이용할 수 있는 미국 주요 도시의 기온을 추적합니다.

(ii) 특정 범위의 입력값(우편 번호)을 인수로 가지는 테이블 함수는 외부 웹 서비스 호출의 결과 데이타를 가상 관계형 테이블로서 제공하는 데 사용됩니다.

(iii) 현재 기온 마이닝: 최종 사용자는 선택한 도시의 현재 기온을 요청(이는 선택한 도시의 우편 번호로 Temperature Web service를 즉시 호출함)하고 결과 데이타에 avg, min, max와 같은 집계 함수를 비롯한 SQL의 모든 파워를 적용할 수 있습니다.

(iv) 최근 수집된 기온 마이닝: 데이타베이스 테이블은 정해진 시간까지 수집된 기온을 저장하는 데 사용됩니다. 데이타베이스 일괄 처리 작업은 매 6분마다 실행되도록 계획되고(데모 목적을 위해), 모든 도시의 우편 번호로 Temperature Web service를 콜아웃(call-out)하며, 실제 데이타베이스 테이블에 데이타를 저장합니다. 최종 사용자는 avg, min, max와 같은 집계 함수를 비롯한 SQL의 모든 파워를 사용하여 이 테이블에서 선택한 도시의 최근 수집된 기온을 "마이닝"할 수 있습니다.

이 추적 및 마이닝 방식은 주가, 과학 데이타, IRS 세액표 등과 같은 외부 웹 서비스로서

사용될 수 있는 모든 "비즈니스 데이타"에 적용될 수 있습니다.

전체 데모는 otn.oracle.com/sample_code/tech/java/jsp/samples/tablefunction/Readme.html 에서 볼 수 있습니다.

## 데이타베이스 웹 서비스 로드맵

이제 한 걸음 뒤로 물러서서 여기서 제시한 예와 설명에 비추어 일반화해 봅시다. Oracle Database 웹 서비스는 다음의 두 가지 기능을 발전시킬 것입니다.

- 데이타베이스 웹 서비스 콜인(call-in) : 표준 웹 서비스 메커니즘을 통해 SQL, XML, PL/SQL, Java, Advanced Queuing 및 Streams를 비롯한 모든 데이타베이스 API를 사용하여 구현된 서비스를 호출할 수 있음.
- 데이타베이스 웹 서비스 콜아웃(call-out) : 데이타베이스 자체에서 J2EE 및 .NET을 기반으로 한 웹 서비스와 같은 외부 웹 서비스를 호출하고 SQL의 파워와 결합할 수 있음.

### Oracle Database 웹 서비스 빌딩 블록

여기에서는 장차 활용하게 될 기술을 가능하게 해주는 기타 목록을 제공합니다.

#### Java/J2EE 웹 서비스 프레임워크

서비스 제공자로서의 데이타베이스를 위해(웹 서비스는 데이타베이스를 호출함), 우리는 Oracle9*i*AS J2EE에 구축된 Oracle9*i*AS의 기존 및 신규 웹 서비스 기능을 활용할 것입니다.

따라서 보안, 확장성, 신뢰성, 가용성 및 성능과 같은 내장된 서비스 품질 뿐만 아니라 상호 운용성, J2EE 배포물이 모두 무료로 제공됩니다. 또한 서비스 사용자로서의 데이타베이스를 위해(웹 서비스 콜아웃(call-out)), 우리는 정적 및 동적 서비스 클라이언트를 데이타베이스에서 직접 수용할 수 있도록 Oracle9*i*AS 웹 서비스 프레임워크의 클라이언트 하위 집합을 활용할 것입니다.

#### *첨부가 있는 SOAP*

데이타베이스 웹 서비스는 BLOB 및 CLOB과 같은 이진 데이타와 대규모 XML 문서를 반환할 경우(별도의 mime 부분에서), SOAP 엔벌로프 내의 데이타를 인라이닝(in-lining)하기 위한 보다 나은 대안으로서 첨부가 있는 SOAP을 활용할 것입니다.

### PL/SQL

PL/SQL은 SQL에 대한 Oracle의 절차적인 확장으로서, SQL 및 데이타베이스 유형과의 완벽하고 긴밀한 통합을 제공합니다. PL/SQL은 데이타 중심 로직 및 메타 데이타를 구현하는 Oracle 개발자들의 대규모 커뮤니티에서 널리 사용되고 있습니다. PL/SQL 내장 프로시저는 비유적으로 말해서 데이타베이스 웹 서비스의 1등 시민(first-class citizen)이라고 할 수 있습니다. 앞에서 강조한 것처럼 원시 PL/SQL은 SOAP 요청 메시지를 구성 및 실행하는 데 사용될 수 있지만, Java를 사용한 방법이 훨씬 더 간단할 뿐만 아니라 래퍼를 통해 SQL 및 PL/SQL에 제공할 수도 있습니다.

### JDBC - Web RowSet(JSR 114)

JDBC RowSet는 일반적으로 데이타베이스인 데이타 소스로부터 RowSet(표 형식 데이타)를 생성하는 기능을 지정합니다. CachedRowSet는 rowset에 대한 수정 사항을 데이타베이스에 다시 전달 및 동기화하는 기능을 통해 연결되지 않은 상태에서도 지속될 수 있는 행 데이타 및 메타 데이타와 같은 연결 해제된 RowSet를 가능하게 만들어 줍니다. JSR 114 사양은 WebRowSet를 XML로 직렬화 가능한 CachedRowSet으로 정의합니다. WebRowSet는 웹 서비스 뿐만 아니라 J2EE 구성 요소 간에도 쉽게 공유될 수 있습니다. 데이타베이스 웹 서비스는 질의 결과를 반환하기 위한 한 가지 옵션으로서 이 표준 기능을 활용할 것입니다.

### 데이타베이스에서의 XML 지원 - XDB

XML 문서는 웹 서비스의 중심에 있습니다. Oracle9*i* Release 2는 SQL 연산과 새로운 XML 표준을 사용하여 SQL 데이타와 XML의 완벽한 조작 뿐만 아니라, 확장성 있는 XML 문서의 저장 및 검색을 가능하게 하는 고유의 XDB(XML 지원)를 제공합니다. 데이타베이스 웹 서비스는 XMLType, XML Schema, XPath, SQLX, XML Developer's Kit(XSU로 알려진 XML SQL 유틸리티 포함) 및 XQuery와 같은 데이타베이스의 기존 및 신규 XML 기능을 모두 활용할 것입니다.

### 데이타베이스에서의 Java - OracleJVM

Oracle8*i* Release 1(Oracle 8.1.5) 이래로 Oracle은 Oracle의 데이타베이스 세션 아키텍처를 지원하는 긴밀하게 통합된 JVM을 제공해 왔습니다. 어떤 데이타베이스 세션이라도 첫 번째 Java 코드 호출 동안 가상의 전용 JVM을 사용할 수 있습니다. 따라서 이후의 사용자들은 이미 Java가 지원되는 세션을 통해 이점을 얻을 수 있습니다. 실제로 모든 세션은 동일한 세션 분리 및 데이타 통합 능력을 Java 세션에 제공하기 위해 동일한 JVM 코드와 statics를 공유합니다. 이 세션 기반 아키텍처는 작은 메모리 푸트프린트를 제공하고, Oracle 데이타베이스와 동일한 선형의 SMP 확장성을 OracleJVM에 제공합니다. 데이타

무결성을 위한 별도의 처리는 필요하지 않습니다. OracleJVM은 표준 Java 라이브러리를 로드 및 변환하고 이를 모든 데이타베이스 세션에서 사용할 수 있도록 함으로써 데이타베이스 기능의 확장을 가능하게 만들어 줍니다(서비스 사용자로서의 데이타베이스를 위한 JAX-RPC 클라이언트를 사용함). 웹 서비스 클라이언트-프록시를 나타내는 Java 클래스는 데이타베이스의 JVM 내에 배포되고, 외부 웹 서비스를 사용하는 데이타베이스 연산을 위해 서비스 메소드에 대한 호출을 수행합니다.

서비스 제공자로서의 데이타베이스를 위해 OracleJVM은 JDBC 및 PL/SQL 래퍼를 통해서나 또는 RMI와 같은 새로운 데이타베이스의 Java 고유의 인터페이스를 통해 직접 호출될 수 있도록 Java 클래스가 데이타베이스에 배포되도록 합니다.

### AQ

AQ(Advanced Queuing)는 Oracle 데이타베이스의 통합 메시지 대기열 시스템으로서, 신뢰성, 무결성, 고가용성, 보안 및 확장성과 같은 데이타베이스의 모든 고유한 이점을 메시징에 제공합니다. 데이타베이스 웹 서비스를 위해 우리는 대기열 관리 작업 뿐만 아니라 메시지를 대기열에 넣고 빼는 것과 같은 데이타베이스 대기열 작업을 제공할 것입니다. 또한 데이타베이스 작업(SQL 문, PL/SQL 패키지, Java 클래스)의 지연 비동기 호출을 허용하기 위해 AQ와 DBMS_JOB과 같은 실행 에이전트를 활용할 것입니다.

### 가상 테이블 테이블 함수

테이블 함수1는 SQL 질의의 FROM 절에서 이 가상 테이블을 사용함으로써 물리 데이타베이스 테이블처럼 질의될 수 있는 행 모음을 생산하는 PL/SQL, Java 또는 C로 작성된 함수입니다.

테이블 함수는 행 집합 또는 커서를 입력 매개변수로 사용할 수 있습니다. 데이타베이스 웹 서비스는 SQL 연산을 (예를 들어, 외부 웹 서비스 호출로 인해 발생하는 데이타에) 적용함으로써 이러한 기능과 향후의 향상된 기능도 활용합니다.

Oracle9*i* Database는 스트리밍, 파이프 라이닝 및 병렬 실행을 통해 테이블 함수 질의 성능과 메모리 사용을 향상시켜 줍니다.

- 병렬 실행: 테이블 함수의 다중 스레드, 동시 실행을 가능하게 함.
- 스트리밍: 프로세스들 간의 중간 스테이지 처리를 제거함.
- 파이프 라이닝: 전체 모음이 테이블 또는 메모리에서 스테이지 처리될 때까지 기다린 다음 전체 모음을 반환하는 대신, 행이 생산됨에 따라 테이블 함수가 반환하는 모음의 결과 행을 제공함.

### 질의 결과 포맷

첫 번째 단계에서 데이타베이스 웹 서비스는 다음과 같은 방식으로 질의 결과를 구체화 합니다.

- JavaBean이 결과 집합의 한 행을 표시하는 경우 JavaBean의 배열로서 구체화
- Oracle의 XML SQL Utility의 포맷과 설명을 사용하여 XML 문서로서 구체화
- WebRowset 포맷 및 설명을 사용하여 XML 문서로서 구체화

특정 XML 스키마는 현재 업계 또는 컨소시엄에서 정의하고 있습니다.

[1]http://otn.oracle.com/docs/products/oracle9i/doc_library/release2/appdev.920/a96595.pdf

우리는 계속해서 이러한 스키마에 대한 지원을 조사할 계획입니다.

### 추가 코드 예제

추가코드 예제, 데모 및 *"길잡이"*는 Oracle Technology Network의 데이타베이스 웹 서비스 코드 예제 페이지 http://otn.oracle.com/sample_code/tech/java/jsp/dbwebservices.html를 확인하시기 바랍니다.

## 결론

앞에서 Oracle9*i* Application Server와 Oracle9*i* Database를 사용하여 데이타베이스 중심의 웹 서비스 기능을 활용하는 방법에 대해 간단히 설명했습니다. 이 제품들은 데이타베이스가 글로벌 또는 기업 수준의 웹 서비스 아키텍처에서 사용자 및 제공자로 작용할 수 있도록 합니다. 데이타베이스 웹 서비스는 SQL, PL/SQL 및 Java에 구축된 기존의 서버 측 인프라를 활용합니다. 또한 그 이상으로 데이타베이스 웹 서비스는 SQL 및 XML 표준을 통한 고유한 XML 문서 저장 및 처리, 메시징 및 지연 실행을 위한 Advanced Queuing 기능과 같은 Oracle 데이타베이스의 추가 기능을 사용할 것입니다.

데이타베이스 웹 서비스 접근 방법은 궁극적으로 데이타베이스 스키마, 테이블, 대기열 및 그 연산을 가상화 및 글로벌화 할 것입니다.

Oracle은 여러분이 데이타베이스 웹 서비스를 통해 Oracle Database 및 Application Server의 결합된 파워의 총체적인 잠재 능력을 인식하도록 돕기 위해 노력하고 있습니다. 몇 개월 후에 제공될 이러한 흥미로운 새 기술에 대한 업데이트는 Oracle Technology Network의 http://otn.oracle.com/tech/webservices/database.html에서 확인하시기 바랍니다.

ORACLE®


## 한국오라클(주)
서울특별시 강남구 삼성동 144-17
삼화빌딩
대표전화 : 2194-8000
FAX : 2194-8001

## 한국오라클교육센타
서울특별시 영등포구 여의도동 23-10
SK증권빌딩 11층(사무실)
19 · 20층(강의실)
대표전화 : 3779-4000
FAX : 3779-4100 1

## 대전사무소
대전광역시 서구 둔산동 929번지
대전둔산사학연금회관 18층
대표전화 : (042)483-4131 2
FAX : (042)483-4133

## 대구사무소
대구광역시 동구 신천동 111번지
영남타워빌딩 9층
대표전화 : (053)741-4513 4
FAX : (053)741-4515

## 부산사무소
부산광역시 동구 초량동 1211 7
정암빌딩 8층
대표전화 : (051)465-9996
FAX : (051)465-9958

## 울산사무소
울산광역시 남구 달동 1319-15번지
정우빌딩 3층
대표전화 : (052)267-4262
FAX : (052)267-4267

## 광주사무소
광주광역시 서구 양동 60-37
금호생명빌딩 8층
대표전화 : (062)350-0131
FAX : (062)350-0130


고객에게 완전하고 효과적인
정보관리 솔루션을 제공하기 위하여
오라클사는 전 세계 145개국에서
제품, 기술지원, 교육 및
컨설팅 서비스를
제공하고 있습니다.

http://www.oracle.com
http://www.oracle.com/kr

제품구입문의


수신자부담 전화번호 : 00368-440-0051 수신자부담 팩스번호 : 00368-440-0062 E-Mail문의 : oracleisd_kr@oracle.com