Erweiterung der Oracle E-Business Suite mit Oracle Application Express: schnell, sicher und einfach!

Yves Chassein, Promatis Software GmbH

Mit Oracle Application Express (APEX) ist es möglich, ausschließlich im Web-Browser sehr rasch anspruchsvolle Datenbank-basierte Desktop- und Mobile-Anwendungen zu entwickeln und zu veröffentlichen. Durch seinen einfachen, deklarativen Ansatz eignet sich Oracle APEX ganz besonders für die schnelle, professionelle Entwicklung von Datenbank-zentrierten Web-Applikationen. Mit diesem Tool kann die Funktionalität der Oracle E-Business Suite einfach erweitert werden. Die ansprechende Bedienung sowie die kurze und steile Lernkurve machen Oracle APEX zu einem idealen Entwicklungswerkzeug für Datenbank-Administratoren, funktionale Anwender und Entwickler. Es gibt verschiedene Möglichkeiten, eine Integration zwischen Oracle E-Business Suite und APEX auf sicheren Wegen herzustellen.

Was ist APEX?

Oracle APEX ist eine Entwicklungsplattform, die SQL, PL/SQL, JavaScript und HTML verwendet, um Anwendungen auf einer Oracle-Datenbank zu erstellen. Die Entwicklung in APEX erfordert lediglich einen Browser. Als Low-Code-Plattform ermöglicht sie auch Anwendern mit wenig Erfahrung, einfache Webanwendungen schnell und ohne großen Aufwand zu programmieren. Natürlich sind auch Realisierungen von Webanwendungen mit hoher Komplexität wie beispielsweise einem Kundenportal möglich. APEX-Anwendungen sind leicht skalierbar. Da APEX im Gegensatz zu Oracle Application Framework (OAF) als Hauptprogrammiersprachen SQL und PL/SQL verwendet, ist es für Oracle-Forms-Entwickler leichter zu erlernen.

Die Hauptmerkmale von APEX:

- Erstellung von Berichten auf Datenbanktabellen, die der Anwender selbst ändern (interaktiver Bericht) und als CSV- oder XLS-Dateien exportieren kann *(siehe Abbildungen 1 und 2)*.
- Erstellen von Formularen zum Eingeben, Ändern oder Löschen von Daten in Datenbanktabellen *(siehe Abbildung 3)*.

Die Geschichte von APEX

Die erste Version von APEX wurde 2004 veröffentlicht, damals unter dem Namen HTML DB. Mit der Version 2.1 im Jahr 2006 wurde sie erstmals APEX genannt.

Ein Jahr später wurden mit der Version 3.1 interaktive Berichte eingeführt. Diese ermöglichten es, den Endanwendern spezielle Filter- und Aggregationsfunktionen für ihre Berichte zur Verfügung zu stellen, um die angezeigten Daten weiter zu analysieren und zu sortieren.

Im Jahr 2010 führte die Version 4.0 Dynamic Actions sowie Plug-ins ein und erweiterte damit die Entwicklungswerkzeuge enorm. Mit der Veröffentlichung von APEX 5.0 im Jahr 2015 wurde die gesamte IDE überarbeitet. Die Entwicklung erfolgte nun auf nur noch einer Seite, dem Page Designer – bis dato hatte jede Komponente ihre eigene Seite gehabt. Die Einführung von Universal Theme ermöglichte den Entwicklern zudem, Anwendungen einfach anzupassen und ansprechende Benutzeroberflächen zu erstellen.

Mit der Version 5.1 wurde das interaktive Grid eingeführt, das die Funktionalität von Formularen und interaktiven Berichten kombiniert. Im Mai 2018 erschien anstelle von APEX 5.1.4 die Version 18.1, die sich somit auf die neuen, globalen Versionsnummern bei Oracle bezieht.

Abbildung 4 veranschaulicht die Versionshistorie von Oracle APEX.

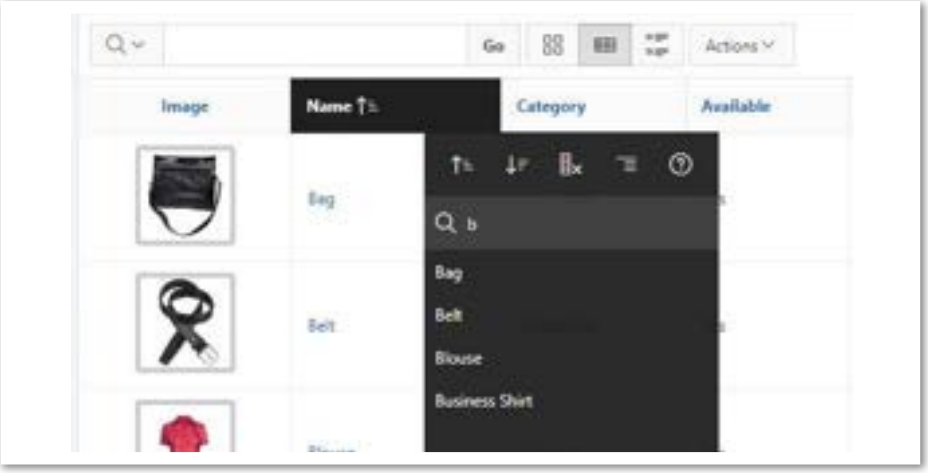

Abbildung 1: Sortierung und Suche in Interactive Reports (Quelle: Oracle)

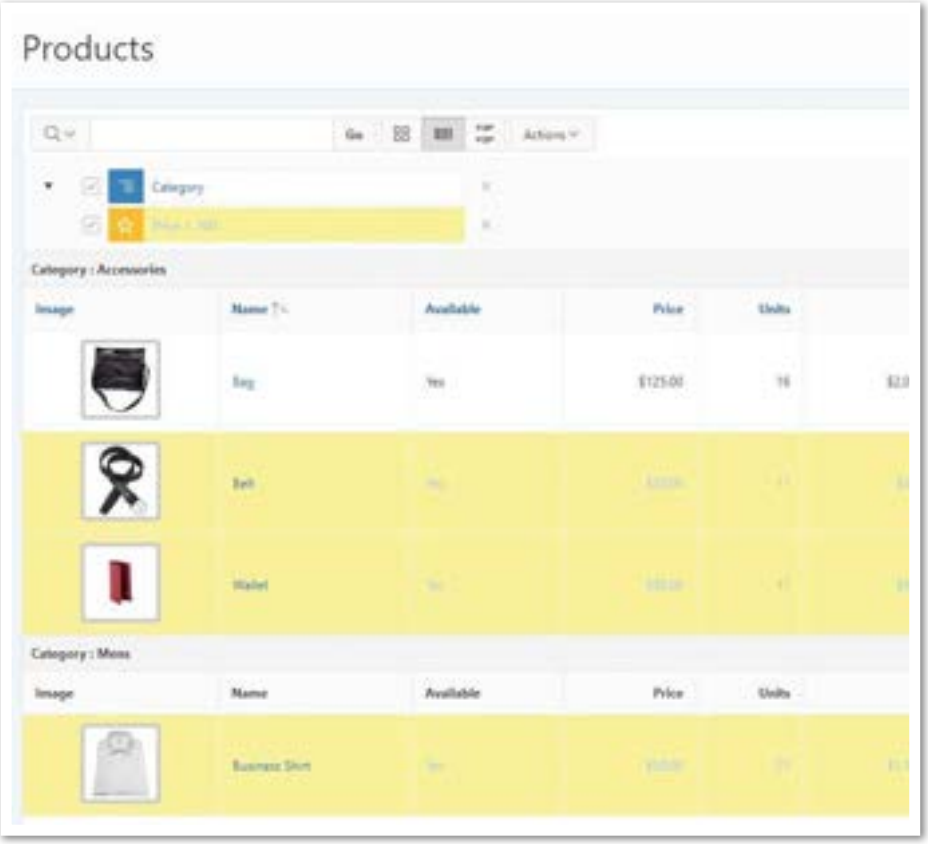

Abbildung 2: Hervorheben von Zeilen (Quelle: Oracle)

APEX-Architektur

APEX verwendet eine 3-Tier-Architektur. Über einen Webserver werden Anfragen vom Browser an die Datenbank gesendet. Die gesamte Geschäftslogik wird auf der Datenbank ausgeführt. So kann beispielsweise auf dem Webserver ein Oracle Rest Data Service (ORDS) verwendet werden, der die Anfrage entgegennimmt und an die Datenbank weiterleitet, wo sie anschließend verarbeitet wird *(siehe Abbildung 5)*. Nach Abschluss der Verarbeitung wird das Ergebnis über ORDS an den Browser zurückgegeben.

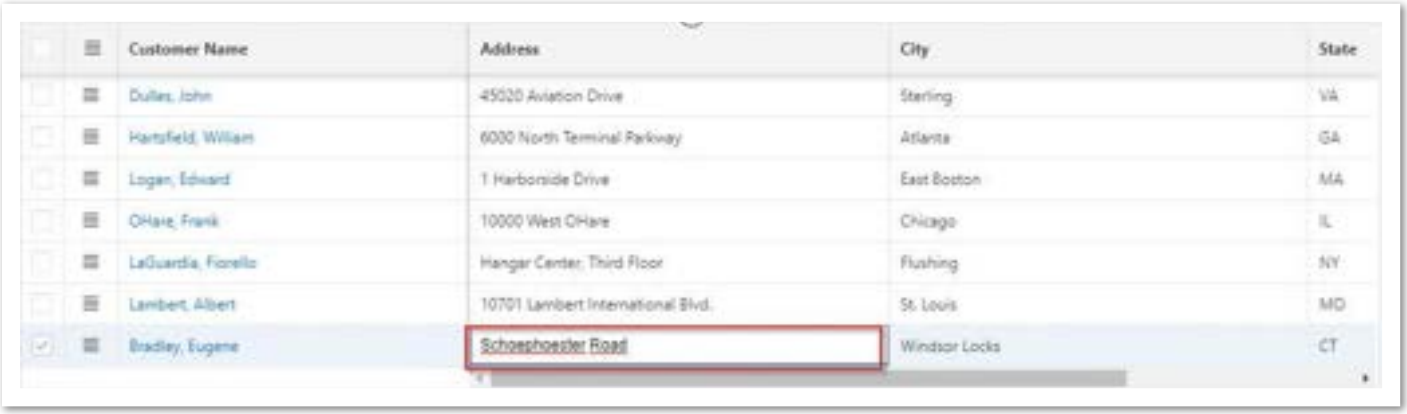

Abbildung 3: Interactive Grid zum Editieren von Daten (Quelle: Oracle)

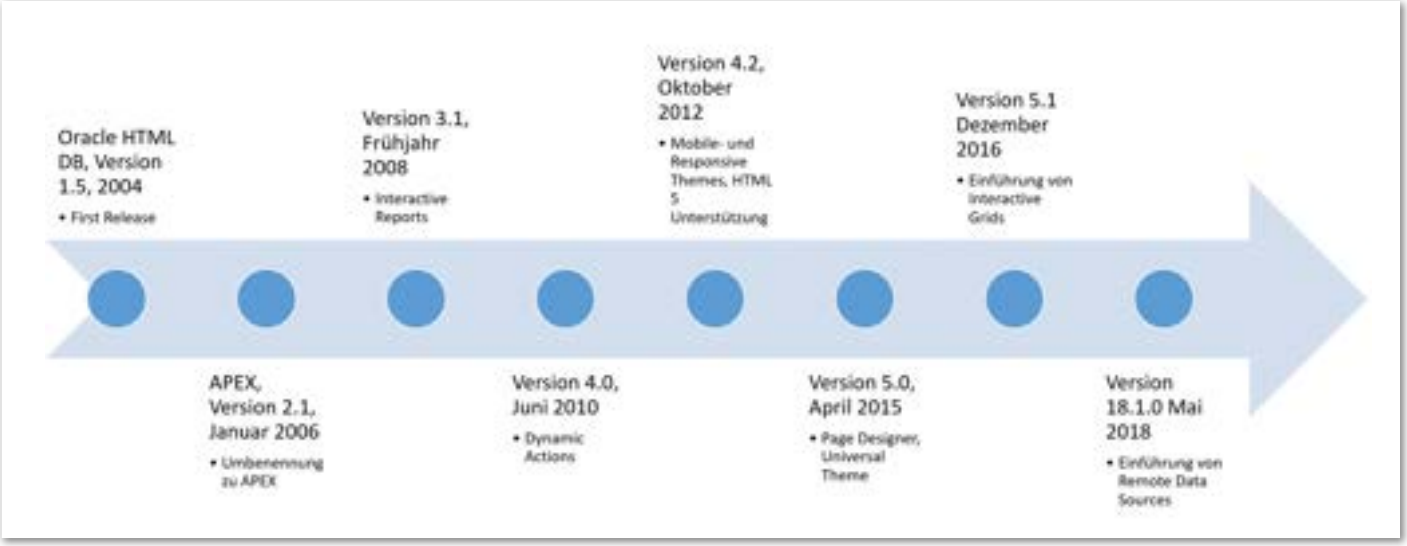

Abbildung 4: Zeitliche Entwicklung von Oracle Application Express

Anstelle des ORDS ist es auch möglich, den in die Datenbank integrierten Webserver zu verwenden, jedoch wird von Oracle empfohlen, dies nicht auf einer produktiven Instanz zu tun. Eine weitere, aber veraltete Möglichkeit ist die Verwendung des Oracle HTTP Servers zusammen mit mod_plsql. Die erprobte Vorgehensweise sieht den Einsatz von ORDS vor. Hierbei ist es möglich, ORDS im Stand-alone-Modus (nur für Testinstanzen) oder auf einem Anwendungsserver wie Tomcat oder WebLogic anzuwenden.

Im Falle einer Integration mit der Oracle E-Business Suite kann APEX in derselben Datenbank installiert werden – ebenso wie ein zusätzlich benötigter Webserver auf dem physischen Anwendungsserver der Oracle E-Business Suite (EBS). Des Weiteren besteht die Option, APEX auf einem separaten Datenbank- und Applikationsserver zu installieren. Für den Zugriff auf die EBS-Daten ist eine Datenbankverbindung zwischen den beiden Instanzen erforderlich. In vielen Fällen ist es sinnvoll, einen zusätzlichen Tomcat-Server auf dem physischen EBS-Anwendungsserver zu installieren, auf dem der ORDS bereitgestellt werden soll.

Integration zwischen Oracle E-Business Suite und Oracle APEX

Grundsätzlich verfügt APEX über mehrere Alternativen, eine Benutzeranmeldung zu implementieren. Standardmäßig erfolgt das Login mit den APEX- Anmeldeinformationen oder über den Datenbankbenutzer. Es ist möglich, einen dieser Login-Mechanismen zu verwenden, um den Benutzern den Zugriff auf die Anwendung zu erlauben. Aber im Falle der EBS würde dies nicht ausreichen: Hier müssten die gleichen Anmeldeinformationen verwendet werden, mit der sich der Benutzer in der EBS anmeldet. In APEX kann dies durch die Verwendung einer benutzerdefinierten Authentifizierungsfunktion erreicht werden. Für eine bessere Benutzerfreundlichkeit ist jedoch eine einmalige Anmeldung anstelle einer einzigen Authentifizierung sinnvoll. Zusätzlich zum Single Sign-on sollte die Integration diese Anforderungen berücksichtigen:

- Login/SSO
  - Von der EBS kommend sollte für die Anmeldung kein Passwort benötigt werden.
  - EBS-Session-ID, Verantwortlichkeits-, Benutzer- und Organisationsinformationen sollten sicher und nicht änderbar über die URL an APEX übergeben werden.
  - Der Login-Mechanismus sollte ohne Verwendung des Session-Cookies der EBS funktionieren – falls der Host des ORDS nicht mit dem Host der EBS identisch ist.
  - Die Login-URL sollte nur für eine begrenzte Zeit gültig sein (keine Weitergabe der URL zur Anmeldung anderer Benutzer).

- Session-Management
  - Jeder Aufruf einer APEX-Seite sollte die EBS-Sitzung aktivieren, wenn sie von dort aus gestartet wird.
  - Wenn die Sitzung abgelaufen ist, sollte der Benutzer auf das EBS-Anmeldeformular umgeleitet werden.
  - Der Logout-Button in APEX und in der EBS sollte die entsprechende Sitzung auch in der anderen Anwendung beenden.
  - Benutzer sollten sich bei APEX selbst anmelden können, ohne sich auch in der EBS anmelden zu müssen.

- Menü in APEX
  - Das Menü, das der Benutzer in APEX sehen kann, sollte von der Sicherheitsfunktion innerhalb der EBS gesteuert werden, jede Seite in APEX benötigt also eine Verbindung zu einer EBS-Funktion.

- Berechtigung
  - Jede Seite sollte mit einer Funktion in der EBS verbunden sein.
  - Beim Seitenaufruf prüft die Berechtigungsfunktion, ob der Benutzer Zugriff auf die entsprechende EBS-Funktion hat.

- Kontext-Initialisierung
  - Dies sollte bei Bedarf von jeder APEX-Seite durchgeführt werden.

Es gibt mehrere publizierte Ansätze, die einige – aber nicht alle – dieser Anforderungen abdecken. Eine unkomplizierte Vorgehensweise ist die Übergabe des EBS-Benutzernamens als Parameter an APEX. Die Authentifizierungsfunktion prüft nur, ob für den Anwender eine aktive Sitzung vorhanden ist. Ist dies der Fall, erhält der Anwender Zugriff auf die Applikation. Wenn keine aktive Sitzung gefunden werden kann, wird der Zugriff verweigert. Diese Lösung ist sehr unsi-

Abbildung 5: Architektur von Oracle ORDS und APEX (Quelle: Oracle)

cher, da auch ohne Zugriff auf die EBS Benutzernamen erraten und in die URL eingefügt werden können. Selbst wenn anstatt des Benutzernamens die Benutzer-ID in der URL übermittelt wird, kann diese durch Erraten oder Ausprobieren geändert werden.

Daher ist ein sicherer Weg ohne die Übergabe von Variablen im Klartext notwendig. Um diese Anforderung zu erfüllen, muss ein Token für die URL in der EBS erstellt werden, das dann an APEX gesendet und dort decodiert wird, um den Benutzer anzumelden. Das Token besteht aus dem Benutzernamen und den Zuständigkeits-, Sitzungs- und Organisationsinformationen. Zusammen mit diesen Attributen im Token wird auch die Session-ID der EBS in APEX verwendet. Dies ermöglicht, nach der Abmeldung in APEX die Sitzung auch in der EBS zu beenden. Eine EBS-Sitzung bleibt unberührt, wenn sich der Benutzer parallel dazu direkt in APEX anmeldet. Um zu verhindern, dass Sitzungen brachliegen, hat Promatis in APEX einen Timeout eingebaut, der dem von Oracle in der EBS implementierten Session-Timeout entspricht. Sobald die APEX-Sitzung abgelaufen ist, wird auch die EBS-Sitzung beendet. Die gleiche Funktionalität ist auch für die entgegengesetzte Richtung implementiert. So wird ebenfalls ein Timeout ausgelöst, wenn die Sitzung direkt in APEX gestartet ist.

Nach Öffnen der Anwendung ist eine Art Berechtigung erforderlich, da sonst alle Benutzer vollen Zugriff auf alle Seiten hätten, obwohl die entsprechende Zuständigkeit in der EBS nicht existiert. In APEX ist es möglich, Berechtigungsschemata in PL/SQL so zu erstellen, dass auf jeder Seite für jeden Benutzer nach einer EBS-Funktion gesucht werden kann. Mit dieser Funktionalität können Benutzerzugriffe für bestimmte Seiten in APEX definiert werden. Wie in den Anforderungen beschrieben, sollte jede APEX-Funktion mit einer EBS-Funktion verbunden sein. Diese Berechtigung kann direkt in der EBS gepflegt werden.

Für den Zugriff auf EBS, API und Tabellen wird der richtige Session-Kontext benötigt. In einer EBS-Sitzung geschieht dies automatisch. In APEX muss es auf jeder Seite manuell eingestellt werden, um eine korrekte Funktionalität zu gewährleisten. Dazu können die Informationen aus dem Token verwendet werden. Durch den Einsatz von Benutzer-, Zuständigkeits- und Organisations-ID ist es möglich, die Oracle EBS und den Organisationskontext zu initialisieren.

Mit dem Promatis-Ansatz werden alle Anforderungen erfüllt und die Oracle EBS kann sicher in APEX integriert werden.

Einsatzmöglichkeit

Mithilfe von APEX kann die Oracle EBS um Reports oder Formulare erweitert werden. Diese können beispielsweise zusätzliche Daten wie einen dynamischen Footer für BI Publisher Reports pflegen. Auch ein gesamtes Portal, mit dem der Fachbereich arbeitet, kann vor die Oracle EBS geschaltet werden.

Ebenso können über Webservices in APEX Informationen in die EBS geladen werden. In APEX kann man die Staging-Tabellen anzeigen und zum Update bereitstellen, um menschliche Fehler wie Tippfehler zu korrigieren oder Mappings einzurichten.

Die Vorteile im Überblick

Mit APEX auf Oracle EBS erhalten wir ein Framework, bei dem die Erweiterung der Funktionalität für Oracle-Forms-Entwickler leicht erlernbar ist.

Die APEX-Anwendung ist über einen Browser zugänglich, sodass für die Nutzung von APEX mit der Oracle EBS keine zusätzlichen Tools – insbesondere kein Java oder Java Web Start – erforderlich sind. Dieser Vorteil ist von großer Bedeutung, da alle bereitgestellten Browser Java von ihrer Kernfunktionalität ausschließen.

Da APEX eine Reihe von leistungsstarken Berichtsfunktionen wie JET-Diagramme, interaktive Berichte und Grids enthält, können wir die Daten leicht überprüfen und in verschiedenen Formaten wie PDF oder CSV ausdrucken. Die Berichte sind sogar anpassbar; mit zusätzlichen Tools wie dem BI Publisher stellt das Drucken von DOC- oder XLS-Dateien kein Problem dar.

Darüber hinaus sind alle Anwendungen aufgrund der reaktionsschnellen APEX-Oberfläche für mobile Geräte geeignet; dies erhöht die Verfügbarkeit und Zugänglichkeit. Obwohl es keinen offiziellen Oracle-Support gibt, werden während der Entwicklung auftretende Probleme dank der großen und hilfreichen Community in der Regel schnell gelöst.

Yves Chassein
yves.chassein@promatis.de

Yves Chassein ist Vice President und Mitglied des Management Boards der Promatis-Gruppe sowie Leiter des internen APEX-Entwicklungsteams. Er besitzt mehrjährige Erfahrung in den Bereichen Geschäftsprozesse, prozessorientierte Informationssysteme sowie Anwendungsentwicklung mit APEX, SQL, PL/SQL und Java. Chassein ist Spezialist in der Realisierung komplexer transaktionaler und analytischer Zusatzfunktionen für Oracle-Applikationen und führt die Planung und Umsetzung SOA-basierter Integrationskonzepte on premises und in der Cloud durch.