OPP/APEXPosed

Oracle-event ideaal om te netwerken

De Oracle Development Tools User Group (ODTUG) organiseert elk jaar naast haar grote conferentie Kaleidoscope ook een aantal Seriously Practical Conferences. Een daarvan is OPP/APEXPosed. Patrick Barel is in november van het vorige jaar naar Atlanta afgereisd waar hij dit tweedaagse seminar bezocht en ook twee sessies verzorgde. Voor Optimize heeft hij zijn bevindingen uit de Verenigde Staten beschreven en blikt terug.

OPP/APEXPosed kent in 2009 zijn vierde editie. OPP staat voor Oracle PL/SQL Programming en het idee van dit seminar komt uit de koker van PL/SQL-goeroe Steven Feuerstein. Vier jaar eerder is bij hem het idee opgeborreld om het tienjarig jubileum te vieren van zijn bestseller 'PL/SQL Programming', waarvan inmiddels alweer de vijfde druk is verschenen. Het eerste jaar heeft de ODTUG nog geen rol gehad in de organisatie, maar sinds twee jaar combineert de organisatie dit seminar met een conferentie gericht op APEX.

Kleinschalig evenement

Het is een kleinschalig evenement dat vooral bedoeld is als lokale Oracle-conferentie met een heel specialistisch onderwerp. Desondanks komen er zo'n 250 bezoekers op OPP/APEXPosed af en voor een lokaalgerichte conferentie is het publiek erg divers. Het organiserend comité laat weten dat er bezoekers zijn uit 33 staten van de VS en naast Amerikanen zijn er mensen uit zeven landen aanwezig. Voor een lokaalgericht evenement mag je dit best opmerkelijk noemen. Het voordeel van een kleinschalig evenement is dat iedereen makkelijk aanspreekbaar is. Het is eenvoudig om contact te leggen met sprekers en productmanagers om eens van gedachten te wisselen. Een ideaal moment daarvoor is een speciaal georganiseerde receptie aan het einde van de eerste dag. Na een volle dag met sessies is het leuk om daarover na te praten.
OPP/APEXPosed is zoals gezegd een gecombineerd evenement. Bij het registeren moet je als bezoeker dan ook al aangeven voor welk deel van het programma je komt. De splitsing is waar te nemen aan de naambordjes die iedereen draagt. PL/SQL-liefhebbers hebben een oranje rand aan de bovenzijde van hun naamkaartje en groen is de kleur voor mensen die vooral voor APEX komen. Desondanks is het goed mogelijk om zelf je programma samen te stellen en van allebei de onderwerpen iets mee te pikken. Er zijn zes parallelsessies per tijdslot, dus keuze is er genoeg.

Aankondigingen

Grote aankondiging vanuit Oracle, zoals die op de conferenties ODTUG Kaleidoscope en Open World worden gedaan, zijn er tijdens dit evenement niet. Dat is eigenlijk ook wel te verwachten gezien het onderwerp en het programma. We hebben ons daarom ook niet ingesteld op grote onthullingen. Toch heeft Bryn Llewellyn in zijn keynote veel losgelaten over 'Edition Based Redefinition', de nieuwe feature van Oracle 11G release 2.

> ***Op grote aankondigingen vanuit Oracle hoef je op OPP/APEXPosed niet te rekenen. Daar is het de conferentie niet voor.***

Dit is een methode om een upgrade van objecten in de database te doen, zonder dat hiervoor downtime nodig is. De bestaande sessies draaien gewoon door terwijl de nieuwe versie van de objecten in een nieuwe edition worden aangemaakt. Zodra deze edition klaar is, dan kunnen de nieuwe sessies hier gebruik van maken. Als niemand meer gebruik maakt van de 'oude' editie, dan kan deze edition verwijderd worden. Zo kan bijvoorbeeld nieuwe functionaliteit getest worden in een live omgeving met de echte data. Maar het is ook mogelijk om bepaalde gebruikers met nieuwe functionaliteit te laten werken, terwijl de anderen nog met de bestaande functionaliteit wer-

ken. Ze kunnen echter wel gewoon elkaars data zien. Meer informatie over 'Edition-Based Redefinition' is te vinden in de documentatie op de site van Oracle.

Don Bales heeft in zijn sessie zijn pijlen gericht op het onderwerp 'Object-Orientated Development With PQPL/SQL'. De gemiddelde PL/SQL-ontwikkelaar is een gewone, rechttoe-rechtaan ontwikkelaar met een procedurele blik. Het is met Oracle echter ook mogelijk om OO te ontwikkelen. Volgens Bales kun je met een drietal objecten bijna alle tabellen aan. Door het slim toepassen van inheritance of overerving bereik je een grote mate van SPOD (Single Point Of Definition). Hier moet wel bij opgemerkt worden dat dit niet van de ene op de andere dag is ontstaan. Wat wel duidelijk is, is dat bij elk stukje nieuwe functionaliteit goed wordt nagedacht op welk niveau deze functionaliteit het best gebouwd kan worden. Dit vereist een hele andere manier van denken dan we gewend zijn, maar het heeft zeker zijn toepassingen.
In de afsluitende sessie van APEXPosed is wel het een en ander van APEX 4.0 getoond. Wanneer deze versie beschikbaar komt is echter niet duidelijk geworden. 'Binnen een paar maanden en dan hangt het een beetje af van jouw definitie van een paar' was het antwoord op deze vraag. Wat nu wel bekend is, is dat er met een APEX 4.0 gespeeld kan worden op http://tryapexnow.com/. Dit is een 'early adopter' release van APEX 4.0 waar we al een beetje kunnen proeven van de nieuwe features die we straks mogen verwachten.

Seriously, Practical?

Deze conferentie wordt getypeerd als 'Seriously Practical', maar dat geldt eigenlijk alleen voor de presentatoren. Het practical-deel dan. Zelf heb ik dit keer twee presentaties gehouden. De eerste met de titel 'Pipelined Table Functions' en voor mijn tweede sessie heb ik de titel 'Optimizing SQL With Collections' gekozen. Beide presentaties zijn ondergebracht in het track 'Performance'.
Door gebruik te maken van Pipelined Table Functions kun je de uitkomst van een PL/SQL functie in SQL benaderen alsof het een 'gewone' tabel is en de functie opnemen in de FROM clause van je select statement. Je kunt natuurlijk je functie ook in de SELECT clause opnemen, maar dan wordt de functie voor elke rij uit het resultaat uitgevoerd. En elke keer dat de functie uitgevoerd moet worden, krijg je een context-switch om je oren. Er moet dan gewisseld worden tussen de SQL en de PL/SQL engine. Dit schakelen kost veel machineinstructies en is dus duur. Toegegeven, het gaat om milliseconden, maar als je 50.000 keer 10 milliseconden kwijt raakt, dan mis je al 500 seconden, ofwel ruim 8 minuten. Als je de functie in een predicaat gebruikt (ofwel een where clause), dan wordt de functie voor elke rij die bekeken wordt uitgevoerd en wordt er vervolgens gekeken of de rij voor mag komen in het resultaat. Door in je functie handig gebruik te maken van bulk operaties kan je ervoor zorgen dat er slechts een beperkt aantal context switches uitgevoerd wordt waarmee je de performance van je query aardig kunt op krikken. Ook heb je (natuurlijk) de beschikking over alle mogelijkheden die PL/SQL je biedt, zoals Package Variabelen, Exception Handling enzovoort...

Mijn tweede presentatie heb ik gehouden over het gebruik van Collections en dan met name het optimaliseren van SQL met collections. Door gebruik te maken van Bulk Operations (Bulk Collect en Forall) kun je het aantal context switches tussen de SQL en de PL/SQL engine minimaliseren waardoor je code dramatisch veel sneller wordt (kan worden). Het gebruik van bijvoorbeeld Bulk Collect is relatief eenvoudig, het enige waar je rekening mee moet houden is dat je een hele stapel resultaten terugkrijgt in plaats van 1 enkele. Dus in plaats van het resultaat op te vangen in een (of meer) simpele scalar variabele, moet je nu het resultaat opvangen in een collectie.

```
declare


  cursor c_employees
  is
  select employee_id
       , first_name
       , last_name
       , email
       , phone_number
       , hire_date
       , job_id
       , salary
       , commission_pct
       , manager_id
       , department_id
    from employees;
  r_employees employees%rowtype;
begin
  open c_employees;
  fetch c_employees into r_employees;

  while c_employees%found loop
    dbms_output.put_line(r_employees.last_name);
    fetch c_employees into r_employees;
  end loop;
end;
/
```

```
         declare
  type employees_tt is table of employees%rowtype index by binary_inte-
ger;

  cursor c_employees
  is
  select employee_id
       , first_name
       , last_name
       , email
       , phone_number
       , hire_date
       , job_id
       , salary
       , commission_pct
       , manager_id
```

```
       , department_id
    from employees;
  r_employees employees_tt;
begin
  open c_employees;
  fetch c_employees bulk collect into r_employees;
  if r_employees.count > 0 then
    for indx in r_employees.first .. r_employees.last loop
      dbms_output.put_line(r_employees(indx).last_name);
    end loop;
  end if;

end;
/
```

Ook het terugsturen van de data naar de database kan nu in één enkele keer. In plaats van voor elk record een nieuwe insert/update of delete uit te voeren, kun je deze statements nu 'bundelen' en als een enkele actie aanbieden aan de SQL engine. In plaats van een Numeric For Loop om de records stuk voor stuk te inserten, kun je via een Forall statement hetzelfde bereiken. Het enige dat je in principe moet doen is een for loop vervangen door een forall statement.

```
...
  if r_employees.count > 0 then
    for indx in r_employees.first .. r_employees.last loop
      insert into employees_bu
        ( employee_id, first_name
        , last_name, email
        , phone_number, hire_date
        , job_id, salary
        , commission_pct, manager_id
        , department_id)
      values
        ( r_employees(indx).employee_id, r_employees(indx).first_name
        , r_employees(indx).last_name, r_employees(indx).email
        , r_employees(indx).phone_number, r_employees(indx).hire_date
        , r_employees(indx).job_id, r_employees(indx).salary
        , r_employees(indx).commission_pct, r_employees(indx).manager_
id
        , r_employees(indx).department_id);
    end loop;
  end if;
...       ...
  if r_employees.count > 0 then
    forall indx in r_employees.first .. r_employees.last
      insert into employees_bu
        ( employee_id, first_name
        , last_name, email
        , phone_number, hire_date
        , job_id, salary
        , commission_pct, manager_id
        , department_id)
      values
        ( r_employees(indx).employee_id, r_employees(indx).first_name
        , r_employees(indx).last_name, r_employees(indx).email
        , r_employees(indx).phone_number, r_employees(indx).hire_date
        , r_employees(indx).job_id, r_employees(indx).salary
        , r_employees(indx).commission_pct, r_employees(indx).manager_
id
        , r_employees(indx).department_id);
  end if;
...
```

Tools Track

Naast de sessies over verschillende onderwerpen met betrekking tot PL/SQL en APEX is er ook een zogenaamde 'Tools Track'. Daarin staan de verschillende tools die een PL/SQL-ontwikkelaar tegenwoordig tot zijn beschikking heeft centraal. Deze track heeft demonstraties van SQL Developer, Toad, CodeTester en PL/SQL Developer laten zien. Deze laatste presentatie heb ik zelf verzorgd.
De opkomst bij zo'n sessie is kleiner dan bij een van de andere presentaties, maar dat maakt het mogelijk om de sessie meer interactief te maken. Nog steeds niet echt 'Practical' aangezien ik nog steeds degene ben die achter de knoppen zit, maar wel om praktische vragen te behandelen. Misschien ook iets minder 'Serious', alhoewel ik wel serieus op de vragen ben in gegaan, maar er was ruimte voor meer humor. Wat je merkt is dat mensen al gauw de vergelijking maken met hun bestaande IDE, in de meeste gevallen TOAD. Het leuke is dat je (bijna) alles wat je in TOAD kan ook in PL/SQL Developer kunt doen. Sommige dingen zijn echter wel beschikbaar in TOAD en niet in PL/SQL Developer. Maar daar biedt PL/SQL Developer een uitgebreid plug-in framework voor en er zijn door ontwikkelaars meer dan 42 plug-ins gemaakt om tekortkomingen in de IDE (of speciale wensen) te ondervangen.

Deze conferentie is vooral geschikt om te netwerken met sprekers en product-managers.

Geslaagde conferentie

De OPP/ApexPosed conferentie is een leuke, kleine conferentie. Vooral voor het netwerken met de bekende sprekers en de product managers van Oracle is dit een goede gelegenheid. Maar als 'presenter' ben je ook vaak bezig met het beantwoorden van vragen van verschillende deelnemers. Soms ook over onderwerpen waar je geen presentatie over doet. Naast professionele vragen kwamen er natuurlijk ook gewone vragen aan de orde. Vooral het feit dat ik uit Nederland ben afgereisd naar Atlanta was een geliefd onderwerp van gesprek. Al met al een geslaagde conferentie.

Links

- *http://download.oracle.com/docs/cd/E11882_01/appdev.112/e10471/adfns_editions.htm*
- *White Paper van Bryn Llewellyn, http://www.oracle.com/technology/deploy/availability/pdf/edition_based_redefinition.pdf*

Patrick Barel is consultant bij AMIS Services. Hij schrijft op het blog van AMIS (http://technology.amis.nl/blog) en op zijn eigen blog (http://blog.bar-solutions.com).