

Caché 5.1 - práca s dopytmi

Náš model aplikácie je už dosť rozsiahly na to, aby sme nad jeho triedami mohli začať robiť dopytovanie. Tentoraz vám teda ukážeme, aké možnosti práce s dopytmi na dáta vracajúce súbory záznamov nám Caché ponúka.

Jedným z častých dopytov v účtovných aplikáciách je prehľad zostatkov na účtoch syntetickej evidencie alebo prehľad pohybov na účtoch - denník. Ukážeme si aj to, ako možno v Caché tvoriť uložené procedúry a ako s nimi z Javy pracovať (hoci je to z pohľadu objektového programovania, samozrejme, krok späť, pretože s Caché vie Java spolupracovať objektovo, ale niekedy sa relačný spôsob môže hodíť).

Dopytovať sa na dáta v Caché môžeme pomocou SQL i dynamicky, ale v takomto prípade musí server, pokiaľ nemá už vygenerovaný runtime kód nášho dopytu SQL nakešovaný, najskôr prísušný kód zostaviť, čo ho môže zdržovať. Preto je aspoň pri často používaných dopytoch vhodné nadefinovať ich ako súčasť definície triedy, takže sa stanú časťou aplikácie API.

Ukážeme si, ako zostaviť súbor záznamov so zostatkami na účtoch syntetickej evidencie.

V Caché Studiu otvoríme triedu **ucto.demo.Syntetika** na editovanie. Pomocou sprievodcu pridaním dopytu (menu Class -> Add -> New Query) nadefinujeme zoznam vstupných parametrov dopytu (v našom prípade žiadne nebudú) a potom vyberieme, či náš dopyt bude tvorený príkazom SQL alebo kódom v niektorom zo skriptovacích jazykov Caché; zvolíme SQL, zostavu nazveme **Osnova-Report**.

Definícia dopytu bude vyzerať takto:

```
Query OsnovaReport() As %SQLQuery(CONTAINID = 1)
{
  SELECT %ID,cisloUctu,nazov,typ,aktualnyZostatok as zostatok FROM ucto.demo.Syntetika
  ORDER BY cisloUctu
}
```

Vidíte, že je to klasický dopyt SQL. Jediné, čo vás môže zarazíť, je parameter CONTAINID=1. Ten hovorí, že dopyt vracia identifikátor objektu (ako %ID) a pre náš príklad nie je dôležitý. Tento parameter používajú iné technológie. ID sa však hodí na následné otváranie inštancií objektov.

Po skompilovaní triedy a vytvorení aktualizovanej verzie Java proxy triedy sa dopyt stane súčasťou Java projekcie. Dopyty Caché sú vystavené ako statické metódy vracajúce triedu **CacheQuery**. Nasledujúce riadky ukazujú jednoduchý kód volajúci definovaný dopyt:

```
String
url="jdbc:Cache://localhost:56773/" + namespace;
dbconnection = CacheDatabase.getConnection(url, username, password);
System.out.println("Spojeno.");

CacheQuery qry = Syntetika.query_OsnovaReport(dbconnection);
rs = qry.execute();
// hlavicka
String h = "";
for (int i = 1; i <= rs.getMetaData().getColumnCount(); i++) {
  if (i>1) h += " ";
  h += rs.getMetaData().getColumnName(i);
}
System.out.println(h);

while (rs.next()) {
  // vypiseme všetky stĺpce každého
  // najdeného riadku */
  String s = "";
  for (int i = 1; i <= rs.getMetaData().getColumnCount(); i++) {
    if (s.length() > 0) {
      s += " ";
    }
    s += rs.getString(i);
  }
  System.out.println(s);
}
/* Close the ResultSet object */
rs.close();
```

V predminulej časti nášho seriálu sme používali metódu **_open** na otváranie inštancií tried Caché. Táto metóda však predpokladá, že poznáme identifikátor inštancie. My sme pred chvíľou videli, ako si identifikátory inštancií objektov vrátiť: pomocou dopytu. No existuje i pohodnejší spôsob než samostatné volanie dopytu a následné použitie vráteného ID na použitie v metóde **_open**, a to metóda **OpenByQuery**, ktorá kombinuje oba kroky - nájdenie ID a otvorenie objektu.

Ukážka otvorenia inštancie syntetického účtu s číslom 221:

```
Iterator it = Syntetika.openByQuery(dbconnection,"cisloUctu = 2",
new String[] {"221"});
// vime (cisloUctu je unikatni), ze
// pokiaľ najdeme nejaké data, tak nanajvys
// jeden zaznam
if (it.hasNext()) {
```

```
Syntetika syn = (Syntetika)it.next();
System.out.println(syn.getCisloUctu() + " : " + syn.getNazev() + " :
zustatek: " + syn.getAktualniZustatek());
}
```

Možno si ešte vybavíte, že na počiatku tohto seriálu sme v jednej z častí venovaných technológií Jalapeno túto metódu tiež používali.

V tomto prípade sa síce dopyt SQL generuje dynamicky, ale keďže ide o jednoduchý select z jednej tabuľky, ržia na jeho zostavenie je minimálna.

Keď už sme pri dopytovaní, vzhľadom na to, že Caché plne podporuje SQL, možno použiť štandardné dopyty SQL prostredníctvom rozhrania JDBC. Na tom nie je nič obzvlášť, takže si to ukazovať nebudeme. Ale predvedieme, ako možno v Caché označiť metódu alebo dopyt ako uloženú procedúru SQL na volanie (nielen) z Javy nie prostredníctvom objektovej väzby, ale prostredníctvom JDBC.

Urobiť z vloženého dopytu alebo statickej metódy triedy Caché uloženú procedúru je veľmi jednoduché. Stačí pridať k deklarácii metódy či dopytu kľúčové slovo **SQLProc**, prípadne voliteľný názov uloženej procedúry. (Stále pracujeme s definíciou triedy Caché **ucto.demo.Syntetika**.)

```
Query Osnova() As %SQLQuery(CONTAINID = 1) [
SqlName = spOsnova, SqlProc ]
{
  SELECT %ID,cisloUctu,nazov,typ,IFN CONCAT(aktualnyZostatok,' KČ') as zostatok FROM
ucto.demo.Syntetika
ORDER BY cisloUctu
}
```

Uloženú procedúru potom spustíme jednoducho:

```
String url="jdbc:Cache://localhost:56773/" + namespace;
Class.forName ("com.intersys.jdbc.CacheDriver");
dbconnection = DriverManager.getConnection(url, username, password);
CallableStatement cs = dbconnection.prepareCall("{ call
ucto_demo.spOsnova() }");
ResultSet rs = cs.executeQuery();

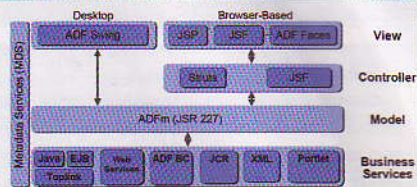
ResultSetMetaData rsmd = rs.getMetaData();
```

Nabudúce sa pustíme do projekcie tried Caché do Enterprise Java Beans.

■ DANIEL KUTÁČ,
Senior Sales Engineer spol. InterSystems B.V.
kutac@intersystems.cz

Konferencia: Oracle Develop 2007

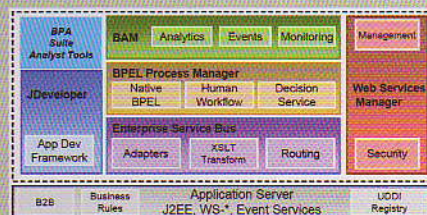
V júni sa v Prahe uskutočnila jedna z najdôležitejších konferencií určených pre vývojársku a databázovú komunitu v strednej Európe - Oracle Develop 2007. Nosnými motívmi konferencie boli nová databáza Oracle 11g, architektúra SOA zameraná na služby, Enterprise Java a vývoj pre .NET a PHP. Prednášky prebiehali v štyroch paralelných sekciami pre architektov SOA, vývojárov



Architektúra Oracle Application Development Framework

databázových aplikácií, vývojárov v Jave a vývojárov využívajúcich technologickú platformu .NET, takže každý si mohol vybrať sekciiu podľa svojho zamerania. Popritom v troch laboratóriách prebiehali pod vedením skúsených lektorov praktické cvičenia z oblasti prezentovaných technológií. Účastníkov zaujali nové črty procedurálneho databázového jazyka PL/SQL, architektúra Oracle ADF (Application Development Framework) na vývoj viacvrstvových aplikácií a aplikačných služieb a možnosti modelovania a vývoja v nových verziách vývojárskych nástrojov Oracle.

Aj vhodné spojenie technológií a filozofie správy dátových úložisk môže v mnohých prípadoch priniesť úsporu prevádzkových nákladov. Typickým príkladom je aplikovanie stratégie ILM (Information Lifecycle Management). Aktívne dáta sa po určitom čase automaticky presunú do úložiska údajov so strednodobou platnosťou a neskôr do archívu historických údajov. Tento postup zabezpečí, aby operačná databáza obsahovala len tie údaje, ktoré sú nevyhnutne potrebné.



Architektúra Oracle SOA Suite

V súvislosti s vývojom webových a intranetových aplikácií bola najviac pertraktovaná téma návrh používateľského rozhrania v zmysle filozofie RUI (Rich User Interface). V nadväznosti na túto filozofiu prebehlo niekoľko prednášok predstavujúcich možnosti technológií AJAX, PHP a Java v spolupráci s databázou a ostatnými middlewarovými produktmi Oracle.

■ LL