PEMANFAATAN JESSICA SEBAGAI *OBJECT RELATIONAL MAPPER* DALAM MEMBANGUN APLIKASI *JAVA* BERBASIS *DATABASE*

Daniel Adinugroho, Rosita Herawati
UNIKA Soegijapranata Semarang
adinugroho@unika.ac.id, rosita@unika.ac.id

Abstrak

Dengan jumlah data yang semakin hari semakin besar, diperlukan sebuah metode atau teknik penanganan data yang efisien dan memudahkan pembuat software. Banyak solusi yang telah ditawarkan. Salah satunya adalah penggunaan RDBMS. Dengan menggunakan RDBMS, pembuat software hanya perlu berkonsentrasi pada business logicnya saja tanpa perlu memikirkan bagaimana proses yang terjadi dalam penyimpanan data karena hal ini merupakan tanggung jawab software RDBMS. Namun, penggunaan RDBMS mengharuskan pembuat software untuk mempelajari bahasa baru yaitu Structured Query Language (SQL) yang merupakan bahasa yang digunakan untuk memanipulasi RDBMS.

Selain itu, konsep RDBMS terkadang tidak memenuhi konsep pemrograman yang saat ini menjadi standar dalam pembuatan software yaitu konsep pemrograman berbasis objek. Yang paling disoroti disini adalah data yang pada RDBMS bersifat publik yang artinya dapat diakses secara langsung dengan perintah-perintah SQL.

Untuk itulah, perlu adanya pengembangan sebuah platform khusus yang diberi nama Jessica, dimana database dapat diakses secara langsung dari platform Java sebagai contoh bahasa pemrograman yang paling sering diidentikan dengan konsep pemrograman berbasis objek, tanpa perlu menggunakan perintah-perintah SQL.

Pemanfaatan Jessica diharapkan dapat memudahkan pembuat software dalam mengembangkan sebuah software dengan bahasa pemrograman Java tanpa perlu menggunakan perintah-perintah SQL lagi, sehingga pembuat software tinggal berkonsentrasi terhadap business logic softwarenya itu sendiri.

***Kata Kunci**: Object Relational Mapper, Pemrograman berbasis Objek, SQL, Java.*

1. PENDAHULUAN

Kompleksitas problem yang dihadapi oleh seorang pembuat *software* semakin hari semakin tinggi. Kebutuhan akan pengumpulan informasi, penyimpanan data, pengolahan data, penyampaian data sudah menjadi satu bagian yang terintegrasi dan sulit untuk dipisahkan lagi. Belum lagi jumlah data yang harus dikelola, bukan lagi dalam hitungan ribuan melainkan sudah mencapai angka jutaan bahkan lebih.

Relational Database Management System (RDBMS) menawarkan konsep penyimpanan data yang dapat memudahkan seorang pembuat software dalam membangun aplikasi berbasis

database. Sebelum konsep ini dikembangkan, pembuat software sendiri yang bertanggung jawab untuk melakukan pengaturan penyimpanan data. Misalnya saja, dalam hal proses pencarian data (*data retrieval*), seorang prorammer harus memikirkan struktur data apa yang tepat sehingga proses pencarian menjadi cepat tetapi proses memasukan data (*insert*) atau penghapusan data (*delete*) tidaklah rumit.Contoh yang lain adalah bila software ini nanti akan digunakan oleh lebih dari satu pengguna. Pembuat software harus memikirkan mekanisme locking pada data sehingga intergritas data tetap terjamin. Dengan menggunakan RDBMS, pembuat software dibebaskan dari tugas-tugas di atas yang pada akhirnya akan mempermudah dan mempercepat proses pembuatan software.

Pada kenyataannya, banyak konsep RDBMS yang tidak sejalan dengan konsep pemrograman berbasis objek, antara lain konsep *Inheritance*, *Encapsulation*, *Polymorphism* dan *Abstraction*. Padahal seperti sudah diketahui, standar pembuatan software sekarang ini banyak yang menggunakan konsep berbasis objek. Kompleksitas dari permasalahan akan mudah diuraikan dengan mengikuti konsep-konsep pemrograman berbasis objek.

Tentu saja perbedaan konsep yang muncul akan menyulitkan pembuat software dalam membangun aplikasi berbasis database dengan mengunakan konsep programming berbasis objek secara utuh. Di sisi lain, penggunaan RDBMS juga mengharuskan pembuat software tersebut untuk menguasai perintah-perintah SQL yang tentu saja sangat berbeda dengan perintah-perintah yang digunakan oleh bahasa permrograman berbasis objek, seperti Java misalnya.

Java sebagai salah satu bahasa pemrograman berbasis objek yang populer dan banyak digunakan orang, mempunyai solusi untuk melakukan koneksi dengan berbagai macam RDBMS. Java menyediakan koneksi ke RDBMS manapun dengan mudah melalui konsep Java DataBase Connectivity (JDBC) [1].

2. OBJECT RELATIONAL MAPPER

Beberapa solusi telah ditawarkan. Salah satunya adalah konsep *Object Relational Mapper* (ORM) [1]. Konsep ini menekankan pemetaan sebuah objek dengan sebuah relasi atau tabel. Dengan konsep ini, pembuat software tidak lagi memerlukan perintah-perintah SQL untuk dihafalkan, karena ia dapat menggunakan perintah-perintah Java secara langsung untuk melakukan manipulasi database seperti pembuatan tabel, menampilkan isi tabel dan merubah atau menambahkan data pada tabel.

Memang sudah banyak ORM yang ditawarkan, diantaranya adalah hibernate. Namun dari banyak solusi yang ditawarkan, masih banyak yang tidak memenuhi konsep pemrograman berbasis objek secara utuh. Yang paling sering kita jumpai adalah pelanggaran (*violation*) terhadap konsep encapsulation dimana data yang seharusnya bersifat private, dapat diakses secara langsung tanpa menggunakan *mutator* dan *accessor*.

Dalam konsep *encapsulation*, yang ingin dicapai adalah integritas dan keamanan data itu sendiri. Dengan akses melalui *mutator*, data yang akan diset dapat terlebih dahulu diverifikasi nilainya sehingga tidak akan ada data yang memiliki nilai yang tidak kita inginkan. Di sisi lain, *accessor* mengijinkan pembuat software untuk melakukan penyajian data dalam berbagai format yang ada tergantung dari permintaan yang ada.

3. ODBMS

Alternatif lain dari penggunaan ORM adalah langsung menggunakan database yang dibangun dengan konsep permrograman berbasis objek atau

yang lebih dikenal dengan *Object Database Management System* (ODBMS) [3].

ODBMS memang sedari awal dibangun dengan konsep pemrograman berbasis objek. Diharapkan dengan konsep ini, akan memudahkan pembuat software karena database yang digunakan sudah menerapkan konsep-konsep pemrograman berbasis objek.

Namun, kelemahan yang mencolok yang menyebabkan ODBMS gagal untuk diterapkan secara masal adalah ODBMS lebih lambat dibanding RDBMS, untuk perintah-perintah umum yang belum dipersiapkan sebelumnya. Belum lagi kurangnya dukungan terhadap software *connectivity, backup* dan *reporting tools* yang sudah ada.

Perbedaan konsep pemrograman berbasis objek dan RDBMS memang terus menjadi perdebatan. RDBMS lebih melihat pada sisi *declarative* dan *attribute-driven viewpoint*, sedangkan pemrograman berbasis objek lebih meliha pada sisi *behavioral viewpoint* [4].

4. JESSICA

Dalam pengamatan penulis, ORM adalah solusi yang lebih tepat daripada penggunaaan ODBMS. Perbedaan sudut pandang pada pemrograman berbasis objek dengan RDBMS terlihat sulit untuk dicarikan titik temunya. Untuk itu solusi seperti ORM akan lebih efektif daripada memaksakan penggunaan ODBMS.

Kekurangan yang ada pada solusi-solusi ORM yang ada yang ditawarkan masih dapat diperbaiki atau dengan pembuatan solusi ORM baru.

Unuk itulah, penulis merasakan perlu adanya platform ORM baru yang benar-benar telah memenuhi konsep pemrograman berbasis objek sehingga dibuatlah Jessica.

Alasan lain adalah Jessica dapat digunakan sebagai model pembelajaran pemrograman database dengan konsep pemrograman berbasis objek. Sehingga pembuat program tidak lagi perlu menghafalkan perintah-perintah SQL bila perlu melakukan manipulasi pada database dan dapat lebih berkonsentrasi secara penuh pada *business logic* yang diperlukan pada softwarenya.

Jessica dapat dikatakan sebagai middle layer antara program dalam hal ini program yang dibuat dengan Java dan database. Database yang akan digunakan tidak boleh dibatasi pada hanya satu macam software database saja, tetapi database yang akan digunakan bisa database apa saja asalkan dia mempunyai dukungan terhadap SQL.

Meskipun nantinya pembuat program tidak perlu mengetahui perintah-perintah SQL, platform Jessica itu sendiri tidak dapat dilepaskan dari perlunya dukungan terhadap SQL.

Seperti pembahasan pada artikel [3] di bab sebelumnya, perbedaan sudut pandang konsep pemrograman berbasis objek dengan RDBMS, telah menyebabkan ODBMS tidak dapat berkembang. Solusi ORM yang memanfaatkan fleksibilitas dan kecepatan RDBMS menjadi lebih diminati untuk dikembangkan. Dengan desain objek yang baik dan tepat, ORM seperti Jessica akan memiliki kecepatan akses ke data yang mendekati akses langsung ke RDBMS.

Dalam tahap perngembangan ini, tentu saja Jessica belum mempunyai support yang sangat luas. Sementara ini Jessica baru diujikan pada database MySQL dan Firebird saja. Meskipun tidak menutup kemungkinan bila dapat dijalankan dalam software database yang lain mengingat Jessica juga menggunakan JDBC sebagai driver untuk koneksi ke database. Sejauh database yang digunakan mendukung SQL dan telah memiliki JDBC atau *Open Database Connectivity* (ODBC) driver (Java mendukung penggunakan ODBC driver melalui JDBC-ODBC Bridge [5]), kelihatannya tidak akan memiliki masalah untuk penggunaan Jessica.

5. STRUKTUR *PACKAGE* JESSICA

Package jessica mempunyai dua macam object yaitu database dan *table*. Object database harus dibuat terlebih dahulu dan nantinya objek *table* menjadi *member class* database. Dengan kata lain, objek database memiliki (*nas a relationship*) satu atau beberapa *tables*.

6. DDL PADA JESSICA

Data Definition Language (DDL) adalah perintah-perintah SQL untuk pembuatan database atau tabel dan menghapusnya. Jessica mempunyai dua macam objek untuk kepentingan yang berbeda. Objek database akan menampung tabel-tabel yang ada dalam database tersebut. Objek database menyimpan informasi berupa driver JDBC yang digunakan dan lokasi server RDBMS. Sedangkan, tabel berisi *field definition* dan data atau record dari sebuah tabel.

Pembuatan sebuah database sama dengan pembuatan objek di Java pada umumnya yaitu dengan membuat *instance* baru dari class database pada *package* jessica. Demikian juga dengan tabel.

Menghapus database atau tabel pun dilakukan hanya dengan memanggil method drop pada masing-masing objek yang telah dibuat. Pemanggilan method drop pada masing-masing objek akan memicu jessica untuk menjalankan perintah SQL drop database atau drop table.

7. DML PADA JESSICA

Data Manipulation Language (DML) adalah perintah-perintah SQL untuk menampilkan, menambahkan, merubah dan menghapus data. Jessica.

Jessica mendukung manipulasi tabel dengan method-method yang ada pada objek *table*. Meskipun untuk sementara *aggregation functions* an *subqueries* belum didukung sepenuhnya oleh Jessica.

Menampilkan isi tabel (select), menambahkan data baru (insert), merubah data (update), dan menghapus data (delete) dapat dilakukan dengan pemanggilan method-method select, insert, update dan delete yang berkaitan.

Polymorphism juga diterapkan dalam Jessica. Sebagai contoh, method select. Ada beberapa varian dari method select, diantaranya adalah tanpa parameter yang artinya menampilkan semua dield dari semua data yang ada, dengan satu parameter berupa array dari field-field yang akan ditampilkan. Kemudian ada sortedSelect untuk menampilkan secara berurutan dan selectWhere untuk menampilkan berdasarkan kriteria tertentu.

8. DCL PADA JESSICA

Data Control Language (DDL) adalah perintah-perintah SQL untuk mengkontrol akes terhadap database. Untuk sementara, DCL belum didukung oleh Jessica. Pada perencanaannya, DCL akan didukung dengan penambahan method-method yang diperlukan pada object database.

9. TIPE DATA DI JESSICA

Tipe data yang digunakan oleh Java dan yang digunakan oleh RDBMS memang ada perbedaan. Bahkan antara satu RDBMS dengan RDBMS yang lain pun memiliki perbedaan.

Untuk menyeragamkan dan menghindari kebinggungan penggunaan tipe data, Jessica menggunakan tipe data yang digunakan oleh Java.

Dalam tahap pengembangan awal ini, Jessica hanya mendukung tipe data **String**, **int**, dan **double**. Penyetaraannya dengan RDBMS yang digunakan misalnya, **String** dengan **VARCHAR** di MySQL.

10. PENGEMBANGAN JESSICA

Data Definition Language (DDL) adalah perintah-perintah SQL untuk pembuatan database atau tabel dan menghapusnya. Jessica mempunyai

dua macam objek untuk kepentingan yang berbeda. Objek database akan menampung tabel-tabel yang ada dalam database tersebut. Objek database menyimpan informasi berupa driver JDBC yang digunakan dan lokasi server RDBMS. Sedangkan, tabel berisi *field definition* dan data atau record dari sebuah tabel.

11. KESIMPULAN

Dalam tahap pengembangan awal ini memang harus diakui banyak sekali fitur-fitur dari RDBMS yang belum didukung oleh Jessica. Meskipun demikian, Jessica menjanjikan sebuah platform baru yang dapat digunakan untuk pembelajaran dan *production*, juga telah menunjukan bahwa perbedaan sudut pandang bisa diserasikan dengan model penyajian tertentu.

Pemanfaatan Jessica diharapkan dapat memudahkan pembuat software dalam mengembangkan sebuah software dengan bahasa pemrograman Java tanpa perlu menggunakan perintah-perintah SQL lagi, sehingga pembuat software tinggal berkonsentrasi terhadap *business logic* softwarenya itu sendiri.

12. DAFTAR PUSTAKA

[1] SUN Microsystems, "JDBC", http://java.sun.com/javase/technologies/database.jsp.

[2] Wikipedia, "Object Relational Mapping", http://en.wikipedia.org/wiki/Object-relational_mapping, 2006.

[3] Barry, Douglas and Duhl, Joshua, "Object Storage Fact Books: Object DBMSs and Object-Relational Mapping", Barry & Associates, Inc., 2001.

[4] Wikipedia, "Object-Relational Impedance http://java.sun.com/j2se/1.3/docs/guide/jdbc/bridge.html.