BAB 6

KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan dari penelitian yang telah dilakukan penulis. Saran penulis berupa saran dalam pengembangan penelitian lanjutan terhadap VoltDB selanjutnya.

6.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan terhadap VoltDB, berikut kesimpulan yang dapat diambil:

1. Perbedaan mendasar antara DBMS konvensional dengan DBMS berbasis memori terletak dalam segi arsitektur beserta cara kerjanya. DBMS konvensional memanfaatkan *disk* sebagai media penyimpanan utama datanya. Ketika suatu data atau *record* (dalam konteks DBMS) diperlukan untuk diproses maka *record* tersebut akan diambil dari *disk* dengan cara membaca piringan *disk* yang berputar. Setelah *record* yang diinginkan berhasil didapatkan maka data tersebut perlu dipindahkan ke bagian memori utama untuk pemrosesan utamanya. Pemindahan *record* dari *disk* ke memori utama biasa disebut *buffering* atau *caching*. *Buffer* dikelola oleh komponen *buffer manager*. *Buffer manager* berfungsi untuk memindahkan *record*/data/*page* dari *disk* ke bagian memori utama yang disebut *buffer pool*. Letak *page* tersebut kemudian diinformasikan oleh *buffer manager* kepada *file manager* sehingga apabila *record* tersebut diperlukan lagi, tidak perlu dilakukan *buffering* atau *caching* kembali - langsung diarahkan ke *buffer pool*. Pada DBMS berbasis memori seluruh datanya sudah disimpan di memori. Hal ini berbeda dengan DBMS konvensional yang memanfaatkan *disk* sebagai media penyimpanan utamanya. Pada memori tidak terdapat *rotational latency* sebagaimana yang ada pada *disk*. Hal tersebut memungkinkan memori untuk dapat mengakses data secara langsung di lokasi manapun dalam waktu yang cepat. DBMS berbasis memori tidak memanfaatkan *buffer manager* dan *buffer pool* untuk pengambilan datanya karena seluruh data sudah berada pada memori.

2. VoltDB merupakan perangkat lunak DBMS berbasis memori yang mengimplementasikan desain *database* partisi dan replikasi. Dengan desain *database* demikian, VoltDB akan cocok dengan perangkat lunak yang memiliki kebutuhan *throughput* yang besar. Untuk pembangunan koneksi dan interaksi dengan *database* VoltDB dapat dilakukan melalui *command-line interface* (CLI). Seluruh transaksi yang terjadi dalam VoltDB berbentuk *stored procedure*. *Stored procedure* tersebut dapat dibuat dengan menggunakan bahasa pemrograman *Java* atau dapat ditulis langsung pada CLI. Transaksi dalam VoltDB dibagi menjadi dua yaitu, transaksi partisi tunggal dan transaksi multi-partisi.

3. Berdasarkan eksperimen uji performa waktu pemrosesan kueri yang dilakukan, VoltDB unggul $\pm$48 kali lebih cepat ketika tabel direplikasi dan $\pm$96 kali lebih cepat ketika tabel dipartisi dari MySQL pada operasi *write* pada *database* (INSERT dan DELETE). Pada operasi *read-only* (SELECT), VoltDB masih tetap unggul meski perbedaan waktu eksekusi kuerinya tidak terlampau jauh. VoltDB unggul $\pm$13 kali lebih cepat (*asynchronous stored procedure*) dan

±2 kali lebih cepat (*synchronous stored procedure*) dari MySQL. Dengan *execution plan* yang sama VoltDB unggul di seluruh poin pengujian terhadap MySQL. Keunggulan dalam hal kecepatan pemrosesan data pada VoltDB dapat dimanfaatkan dengan baik ketika transaksinya merupakan transaksi partisi tunggal. Transaksi partisi tunggal memungkinkan *stored procedure asynchronous* berjalan secara maksimal.

4. VoltDB menyediakan berbagai *interface* untuk berbagai bahasa pemrograman. *Interface* tersebut berguna agar aplikasi klien dapat berinteraksi dengan *database* dalam VoltDB. Salah satu *interface* yang disediakan VoltDB ialah untuk bahasa pemrograman *Java*. Pada skripsi ini telah dapat dibangun perangkat lunak untuk mendemokan fitur-fitur yang disediakan oleh VoltDB pada bab 5 dalam bahasa pemrograman *Java*.

5. VoltDB kurang cocok untuk mengolah data besar karena kebutuhan *space* memori yang besar apabila suatu tabel direplikasi. Hal tersebut perlu memakan biaya yang jauh lebih besar sehingga kurang cocok bagi bisnis yang memiliki *resource* berupa kapasitas RAM yang terbatas.

6.2 Saran Penelitian Lanjutan

Berdasarkan eksperimen dan kesimpulan diatas, berikut saran penelitian lanjutan yang dapat penulis berikan:

1. Dapat dilakukan penelitian perbandingan MySQL dan VoltDB dengan data yang berukuran lebih besar.

2. Agar dapat mengolah data dengan ukuran sangat besar, perlu dibangun suatu *cluster* yang setiap *node*nya memiliki kapabilitas RAM yang memadai.

3. Untuk mendapat hasil waktu eksekusi kueri yang cepat dapat menggunakan perangkat keras dengan spesifikasi yang lebih baik.

4. VoltDB dapat dibandingkan dengan perangkat lunak DBMS berbasis memori lainnya.

DAFTAR REFERENSI

[1] Gracia-Molina, H. dan Salem, K. (1992) Main memory database systems: An overview. *IEEE Transaction on Knowledge and Data Engineering*, **4**, 509–516.

[2] VoltDB (2019) Using voltdb. https://docs.voltdb.com/UsingVoltDB/. 3 Maret 2019.

[3] Ramakrishnan, R. dan Gehrke, J. (1999) *Database Management Systems*, 2nd edition. McGraw-Hill, USA.

[4] Hollander, S., A., Denna, E. L., dan Cherrington, J. O. (2000) *Accounting Information Technology and Business Solution*, 2nd edition. McGraw-Hill, USA.

[5] Kedar, S. (2009) *Database Management Systems*, 1st edition. Technical Publications Pune, India.

[6] Elmasri, R. dan Navathe, S. B. (2011) *Fundamentals of Database Systems*, 6th edition. Addison-Wesley, USA.

[7] Microsoft (2017) Working with the odbc. https://docs.microsoft.com/en-us/sql/odbc/reference/odbc-programmer-s-reference?view=sql-server-2017. 20 November 2018.

[8] Silberschatz, A., Korth, H. F., dan Sudarshan, S. (2010) *Database System Concepts*, 6th edition. McGraw-Hill, USA.

[9] Hellerstein, J. M., Stonebraker, M., dan Hamilton, J. (2007) Architecture of a database system. *Foundations and Trends in Databases*, **1**, 141–259.

[10] Shaffer, C. A. (2011) *Data Structures Algorithm Analysis inJava*, 3rd edition. Dover Publications, USA.

[11] Faerber, F., Kemper, A., Larson, P.-A., Levandovski, J., Neumann, T., dan Pavlo, A. (2016) *Main Memory Database Systems*, 3rd edition. Springer-Verlag, Berlin.

[12] Pavlo, A. (2014) On Scalable Transaction Execution in Partitioned Main Memory Database Management Systems. Disertasi. Brown University, Rhode Island.

[13] H-Store (2019) H-store architecture overview. http://hstore.cs.brown.edu/documentation/architecture-overview/. 5 Maret 2019.

[14] VoltDB (2013) Voltdb technical overview. http://www.odbms.org/wp-content/uploads/2013/11/VoltDBTechnicalOverview.pdf. 5 Maret 2019.

[15] Stonebraker, M. (1986) The case for shared nothing. *Database Engineering*, **9**, 4–9.

[16] Malviya, N., Weisberg, A., Madden, S., dan Stonebraker, M. (2014) Rethinking main memory oltp recovery. *ICDE Conference 2014*, **1**, 604–615.