"Tema: 4 (teknik dan energi baru dan terbarukan)"

DESAIN MODEL VIEW CONTROLLER DALAM IMPLEMENTASI REDIS SERVER UNTUK MEMPERCEPAT AKSES DATA RELASIONAL

Oleh

Mulki Indana Zulfa, Ari Fadli, Arief Wisnu Wardhana
Teknik Elektro, Universitas Jenderal Soedirman
mulki_indanazulfa@unsoed.ac.id

ABSTRAK

Database relasional masih menjadi database utama dalam pengembangan aplikasi berbasis web karena mampu menjamin konsistensi datanya. Database relasional membangun informasi dari data yang saling berelasi. Namun database relasional masih memiliki kelemahan dalam kecepatan akses data karena datanya masih disimpan di dalam harddisk. Oleh karena itu saat ini muncul teknologi baru pengelolaan data yang tidak lagi disimpan di dalam harddisk dan tidak juga harus dibentuk dari data yang saling berelasi. Teknologi ini disebut nosql database. Nosql database banyak digunakan untuk menyelesaikan masalah pengelolaan big data. Penelitian ini bertujuan mendesain implementasi nosql database yang menggunakan redis server dengan pendekatan Model View Controller (MVC) menggunakan codeigniter. Hasilnya yang dapat menunjukkan bahwa nosql database dapat diimplementasikan dalam framework tersebut dan dapat mempercepat akses data relasional untuk studi kasus menampilkan data Indeks Prestasi Kumulatif (IPK) mahasiswa.

Kata kunci: *database, data relasional, nosql database, MVC, codeigniter*.

ABSTRACT

Relational database is still the main database in developing web-based applications because it is able to guarantee the consistency of the data. Relational databases build information from interrelated data. But relational databases still have weaknesses in data access speed because the data is still stored on the hard disk. Therefore, currently a new technology of data management is emerging that is no longer stored on the hard disk and does not have to be formed from interconnected data. This technology is called nosql database. Nosql database is widely used to solve big data management problems. This study aims to design the implementation of Nosql database using redis server with the Model View Controller (MVC) approach using codeigniter. The results can show that Nosql database can be implemented in the framework and can accelerate relational data access for case studies displaying student cumulative achievement index (GPA) data.

Keywords: database, relational data, nosql database, MVC, codeigniter.

PENDAHULUAN

Setiap hari penyedia dan pengguna layanan berbasis web semakin banyak (J. Yan et al., 2014). Hal ini menjadi pekerjaan rumah tersendiri bagi para pengembang aplikasi web karena jumlah pengunjung yang masif dan simultan dapat menyebabkan *bottleneck* dan *Internet latency* di sisi server (W.Z.S. Bouchenak et al., 2006). Di sisi lain para pengembang aplikasi web masih mengandalkan *relational database* sebagai database utamanya karena mampu mengelola dengan baik data terstruktur dan mampu menjamin konsistensinya (M. Indrawan, 2012). Relational database juga mampu menjamin properti ACID yaitu Atomicity Consistency Isolation dan Durability (A. E. Lotfy et al., 2016). Keempat properti tersebut harus mampu dijaga oleh Relational Database Management System (RDBMS) selama melakukan transaksi data (Archit Nangalia, 2016). Informasi dalam database relasional dibangun dari data yang saling berelasi. Semakin banyak relasi yang dibentuk maka semakin lengkap informasi yang dapat dihasilkan. Namun semakin banyak relasi yang dibangun maka semakin lama waktu yang dibutuhkan untuk mengelolanya (J. Teknovasi et al., 2014), karena RDBMS masih menyimpan datanya di dalam *harddisk* (M. Luthfi et al., 2018).

Di sisi lain perkembangan teknologi database sedang berkembang pesat. Salah satu inovasi besar yang dihasilkan dari perkembangan teknologi ini adalah teknologi penyimpanan data yang tidak lagi berbasis *harddisk* melainkan berbasis *memory* (M. Luthfi et al., 2018). Teknologi database ini dikenal dengan nosql database (F. Firdausillah et al., 2012). Nosql database telah banyak digunakan oleh perusahaan besar di dunia seperti Amazon, Google, IBM, Microsoft dan banyak juga digunakan oleh penyedia jasa cloud lainnya (W. Vogels, 2017; I. Amazon Web Services, 2019; K. Kaur et al., 2013). Nosql database juga mempunyai konsep yang berbeda untuk mengakses data yang dikelolanya, tidak lagi menggunakan *query* melainkan menggunakan konsep *key-value store* seperti yang diimplementasikan oleh redis server (W. Puangsaijai, 2017; D. J. Carlson, 2013).

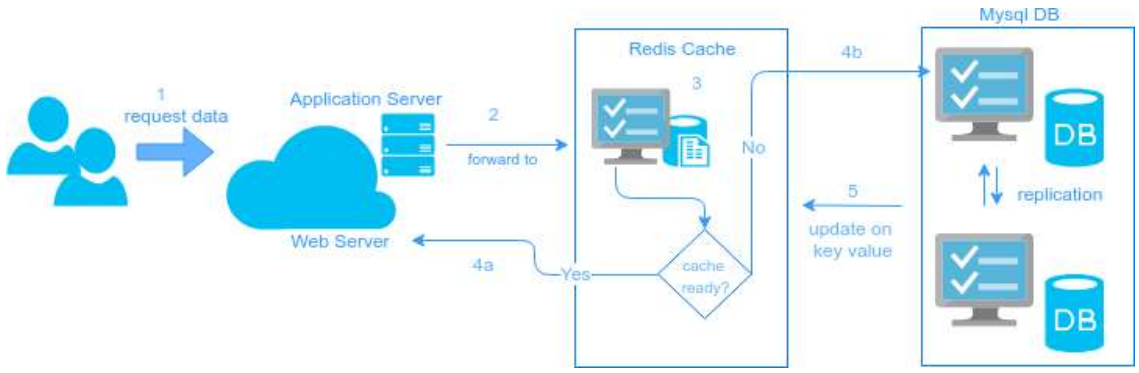
Penelitian ini bertujuan untuk mendesain penggunaan redis server dalam aplikasi berbasis web menggunakan konsep *Model View Controller* (MVC) berbasis *framework* codeigniter. Keamanan, kemudahan implementasi, *library* yang cukup ringan membuat *framework* ini cukup populer digunakan untuk membangun aplikasi berbasis website (D. Upton, 2007). Redis berjalan dengan dalam sistem operasi windows (D. Nielsen, 2018) maupun linux (E. Sverdlov, 2012). Dalam penelitian ini redis server dikembangkan dalam lingkungan sistem operasi linux.

METODE PENELITIAN

Penelitian ini dilakukan selama 6 bulan dimulai sejak April sampai dengan September. Penelitian dilakukan di kampus Fakultas Teknik Universitas Jenderal Soedirman dengan memanfaatkan koneksi jaringan Internet untuk mengakses *Virtual Private Server* (VPS) yang disewa oleh tim peneliti. Penelitian ini disimulasikan menggunakan aplikasi berbasis web menggunakan *framework codeigniter* untuk mengimplementasikan redis server sehingga dapat mempercepat akses

data relasional. Banyak developer web yang masih mengandalkan database relasional sebagai database utama dalam aplikasi yang dikembangkan. Hal ini disebabkan karena database relasional mampu mengelola data terstruktur dengan baik. Namun karena database relasional menyimpan datanya di dalam *harddisk* sehingga menyebabkan akses data relasional menjadi lambat. Oleh karena itu dibutuhkan tools tambahan berupa *in-memory* database menggunakan redis server untuk membantu kinerja database relasional. Data yang sering diakses akan disimpan di dalam *in-memory* database sehingga data yang sama tidak akan dimuat ulang dari database relasional. Hal ini akan meringankan beban kerja database relasional.

Untuk mengimplementasikan hal tersebut maka dirancanglah topologi aplikasi web seperti ditunjukkan pada gambar 1.



Gambar 1. Usulan topologi aplikasi web dengan memanfaatkan redis server.

Cara kerjanya adalah ketika ada permintaan data dari user, maka aplikasi server akan meneruskannya ke redis server. Jika nilai yang diminta sudah ada di redis server, maka hasilnya langsung ditampilkan ke user tetapi jika maka permintaan data dari user akan diteruskan ke database relasional, kemudian sistem langsung membuat *cache* atau *copy* datanya di dalam redis server tersebut.

Penelitian ini menggunakan metode *Rapid Application Development* (RAD) yang menitikberatkan pada eksperimental melalui topologi yang diusulkan yaitu membangun *caching* aplikasi menggunakan redis server yang kompatibel dengan database relasional. Metode ini cocok untuk digunakan dalam riset yang membutuhkan waktu kurang dari 6 bulan. Data penelitian menggunakan data akademik khususnya dalam studi kasus menampilkan data IPK mahasiswa karena data ini dibangun dengan membangun banyak relasi dengan data lain seperti data mahasiswa, dosen, dosen wali, program studi, matakuliah, dan nilai matakuliah. Penelitian ini menggunakan alat dan bahan seperti ditunjukkan pada tabel 1.

Tabel 1. Tools yang digunakan implementasi redis server.

No	Nama Tools	Versi
1	Apache Web Server	2.4
2	Redis server	2.8
3	Codeigniter	3
4	Database relasional: Maria DB	10.3
5	Sistem Operasi	Ubuntu

Proses penelitian diawali dengan melakukan pengecekan terhadap proses instalasi dan konfigurasi semua *tools* pada tabel 1. Kemudian database relasional akan diisi dengan *sample* data mahasiswa dari jurusan Teknik Elektro sejak angkatan 2007/2008-gasal hingga 2017/2018-genap. Data tersebut akan dimodelkan menggunakan konsep data relasional yang kemudian akan dimodelkan juga menggunakan skema *key value* sebagai inisial data pada *redis server*. Setelah data ini masuk ke dalam database resional dan redis, maka percobaan terhadap usulan strategi akan diuji coba. Hasil percobaan akan diukur berdasarkan variabel *response time*.

Pemodelan data akademik dalam penelitian ini dibangun menyerupai data akademik yang ditunjukkan oleh Sistem Informasi Akademik (SIA) Universitas Jenderal Soedirman. Sehingga hasil penelitian ini dapat dikontribusikan dalam pengembangan SIA agar memiliki akses data relasional yang lebih cepat.

HASIL DAN PEMBAHASAN

A. Pemodelan Data Akademik

Data akademik adalah data relasional yang dibangun dari beberapa tabel yang saling berhubungan. Studi kasus penelitian ini fokus dalam menampilkan data IPK mahasiswa. Data IPK ini dibangun dari banyak tabel yang saling berelasi sehingga jika dalam waktu yang bersamaan banyak *request* terhadap permintaan data ini maka *response time* yang diberikan oleh aplikasi server dapat semakin lama. Oleh karena itu data akademik yang dibentuk oleh database relasional akan ditranslasikan ke dalam struktur *in-memory* database untuk membantu aplikasi server agar lebih cepat dalam memberikan respon hasil kepada user.

Dalam penelitian ini, data akademik yang dibutuhkan untuk menampilkan IPK mahasiswanya diwakilkan ke dalam lima tabel yaitu tabel nilai, tabel matakuliah, tabel mahasiswa, tabel dosen, dan tabel program studi. Data dalam lima tabel tersebut dikelola menggunakan *join query* untuk menampilkan data IPK mahasiswa. *Join query* yang digunakan untuk menampilkan data IPK sebagai berikut: `select a.nim, b.nama_mhs, round(sum((c.sks_mk * a.bobot))/sum(c.sks_mk),2) as totalnilai from tb_nilai a join tb_mahasiswa b on a.nim = b.nim join tb_matakuliah c on c.kode_mk = a.kode_mk where a.nim = 'C1X015070' GROUP by a.nim`.

Berbeda dengan database relasional, *in-memory* database tidak mengenal relasi antardata sehingga data akademik tersebut dimodelkan dengan cara lain melalui skema *key-value*. Walaupun begitu data akademik yang meliputi nama mahasiswa dan IPK tetap dapat ditampilkan melalui aplikasi. Salah satu struktur data yang dapat digunakan dalam *in-memory* database, redis, adalah `Set` dan `Get`. `Set` membutuhkan sebuah *key* yang unik untuk dapat menyimpan sebuah data. Data ini dapat diubah dan dihapus berdasarkan *key* tersebut. Untuk menampilkan data yang sudah tersimpan, dibutuhkan operasi `Get`. Setiap key tersebut dapat diberikan waktu hidup menggunakan operasi `Setex` sehingga keberadaannya di dalam redis dapat diatur (M. Arslan, 2016). Sehingga operasi `Set` dan `Get` sering digunakan untuk operasi *Create Read Update Delete* (CRUD) dalam implementasi *in-memory* database untuk aplikasi berbasis web.

In-memory database dalam penelitian ini tidak menjadi database utama. Keberadaannya hanya sebagai database pendamping yang berfungsi sebagai *cache* sehingga jika ada *request* data yang sama tidak harus dimuat ulang dari database, cukup dimuat dari redis dengan waktu respon yang jauh lebih cepat.

B. Pemodelan *Model View Controller*

Data akademik adalah data relasional yang dibangun dari beberapa tabel yang saling berhubungan yang dikelola oleh RDBMS. Sedangkan redis adalah *in-memory* database yang dikelola oleh redis yang berfungsi sebagai *cache* sehingga dapat mempercepat respon terhadap *request* data yang sama. Untuk menjembatani kedua jenis data tersebut maka dibutuhkan desain *model* dan *controller* yang baik agar kedua jenis data yang dikelola oleh dua tools yang berbeda (RDBMS dan Redis) dapat saling melengkapi. *Framework* pengembangan aplikasi web modern menggunakan pendekatan *Model View Controller* (MVC) sebagai *rule* penulisan bahasa programnya. Penelitian ini menggunakan framework codeigniter karena mempunyai struktur *library* yang cukup sederhana dan mudah dipahami sehingga cocok untuk digunakan dalam penelitian dengan durasi waktu kurang dari 6 bulan (D. Upton, 2007).

Penelitian ini menggunakan *library* predis agar dapat redis dalam aplikasi web. *Library* predis memiliki kemudahan dalam instalasi dan integrasi dengan *framework* apapun sehingga dipilih untuk digunakan dalam penelitian ini (D. Alessandri, 2015; M. Arslan, 2016). Dalam *framework codeigniter*, *library* predis dimuat di *controller* tepatnya di dalam *method* construct sehingga teknik penulisan dan pemanggilan *library* ini cukup satu kali saja (lihat tabel 1).

Tabel 2. Cara menggunakan library predis dalam controller.

Lokasi	*controller*
Method	public function_contruct()
Code	`public function __construct()`

```
{
        parent::__construct();
        $this->load->model('Model_data');

        require "predis/autoload.php";
        Predis\Autoloader::register();

        try {
        $redis = new Predis\Client(
              array(
         "scheme" => "tcp",
         "host" => "127.0.0.1",
         "port" => 6379)
              );
        }
        catch (Exception $e) {
              echo "Couldn't connected to Redis";
              echo $e->getMessage();
        }
}
```

Setelah *library* predis dimuat dalam *controller* maka berikutnya adalah mendefinisikan *join query* dalam *Model*. Dalam penelitian ini studi kasus data akademik diformulasikan dalam proses perhitungan IPK mahasiswa. *Join query* yang digunakan dapat dilihat pada tabel 3.

Tabel 3. Join query untuk menghitung IPK mahasiswa.

Lokasi	*model*
Method	Bebas, sesuai dengan definisi user Contoh : `public function ipkbynim ($nim)`
Code	`select`

```
a.nim as nim, b.nama_mhs as nama, d.nama_prodi
as prodi, b.tahun_angkatan as angkatan,
round(sum((c.sks_mk *
a.bobot))/sum(c.sks_mk),4) as ipk

from tb_nilai a
join tb_mahasiswa b on a.nim = b.nim
join tb_matakuliah c on c.kode_mk = a.kode_mk
join tb_prodi d on a.id_prodi = d.id_prodi

where a.nim = '$nim'

GROUP by a.nim, b.nama_mhs
```

Berdasarkan usulan topologi pada gambar 1, maka perlu method tambahan (lihat tabel 4) di dalam controller untuk memberikan arah permintaan data yang pertama kali harus diminta kepada redis sebagai cache. Jika di dalam redis maka data yang dimaksud tidak ditemukan maka permintaan data tersebut diarahkan ke database utama, kemudian datanya di-*copy* ke redis agar pada permintaan berikutnya data tersebut dapat langsung diberikan oleh redis tanpa harus meminta ulang kepada database utama.

Tabel 4. Source code meminta data ke redis dan database utama.

Lokasi	*controller*
Method	Bebas, sesuai dengan definisi user Contoh : `public function ipkbynim ($nim)`
Code	`if(!empty($result))` `{` `$this->benchmark->mark('code_start');` `print_r($result);` `echo '  via REDIS';` `$this->benchmark->mark('code_end');` `echo $this->benchmark->elapsed_time('code_start', 'code_end');` `}` `else` `{` `$this->benchmark->mark('code_start');` `$data = $this->Model_data->tampilipk(" where a.nim = '".$nim."' ");` `print_r($data);` `echo '  via DB';` `$this->benchmark->mark('code_end');` `echo $this->benchmark->elapsed_time('code_start', 'code_end');` `}`

KESIMPULAN

Database relasional masih menjadi pilihan database utama para pengembang aplikasi berbasis web karena memiliki kemampuan yang baik dalam mengelola data yang terstruktur. Namun database ini memiliki kelemahan dalam waktu respon yang cukup lambat karena data yang disajikan masih harus dimuat dalam harddisk. Kelemahan ini dapat ditutup dengan menambahkan in-memory database yang berfungsi sebagai cache sehingga permintaan data yang sama dapat direspon dengan jauh lebih cepat. Untuk menggabungkan kinerja database relasional dan *in-memory* database dalam aplikasi web diperlukan desain *model* dan *controller* yang baik agar permintaan data dapat dilayani oleh keduanya dengan cepat.

UCAPAN TERIMA KASIH

Terimakasih kepada LPPM Universitas Jenderal Soedirman yang telah memberikan dana penelitian BLU melalui skim Penelitian Peningkatan Kompetensi di tahun 2019. Semoga hasil penelitian ini perlahan dapat diimplementasikan di sistem informasi yang ada di Universitas Jenderal Soedirman.

DAFTAR PUSTAKA

A. E. Lotfy, A. I. Saleh, H. A. El-Ghareeb, and H. A. Ali. 2016. A middle layer solution to support ACID properties for NoSQL databases. *J. King Saud Univ. - Comput. Inf. Sci.* 28(1): 133 – 145 pp.

Archit Nangalia. 2016. ACID Properties in DBMS. Geeks for Geeks. https://www.geeksforgeeks.org/acid-properties-in-dbms/.

D. Alessandri. 2015. Predis. https://github.com/nrk/predis. Diakses pada 23 November 2018.

D. J. Carlson. 2013. *Ebook Redis in Action*. Manning Publications.

D. Nielsen. 2018. Running Redis on Windows 10.https://redislabs.com/blog/redis-on-windows-10/. Diakses pada 23 November 2018.

D. Upton. 2007. *Code Igniter for Rapid PHP Application Development*. Packt Publishing. Birmingham.

E. Sverdlov. 2012. How To Install and Use Redis. https://www.digitalocean.com/community/tutorials/how-to-install-and-use-redis. Diakses pada 23 November 2018.

F. Firdausillah, E. Y. Hidayat, and I. N. Dewi. 2012. NoSQL: Latar Belakang, Konsep, dan Kritik. *Semantik* 2012: 432 – 438 pp.

I. Amazon Web Services. 2019. Use Cases and How ElastiCache Can Help. https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/elasticache-use-cases.html#elasticache-use-cases-data-store. Diakses pada 22 Maret 2019.

J. Teknovasi, J. Sinuraya, M. Zarlis, E. B. Nababan, and H. Join. 2014. PERBANDINGAN PENCARIAN DATA MENGGUNAKAN. 1: 71 – 93 pp.

J. Yan, J. Chen, and W. Jiang. 2014. Data caching techniques in web application. *Proceedings - 2nd International Conference on Enterprise Systems, ES 2014*. 289 – 293 pp.

K. Kaur, R. Rani, C. Sci, and E. Deptt. 2013. Modeling and Querying Data in NoSQL Databases. *IEEE International Conference on Big Data*

M. Arslan. 2016. Bermain PHP dan Redis Menggunakan PRedis. Codepolitan. https://www.codepolitan.com/bermain-php-dan-redis-menggunakan-predis.

M. Arslan. 2016. PDKT dengan Redis. Codepolitan. https://www.codepolitan.com/pdkt-dengan-redis.

M. Indrawan-santiago. 2012. Database Research: Are We At A Crossroad? Reflection on NoSQL. *15th International Conference on Network-Based Information Systems*

M. Luthfi, M. Data, and W. Yahya. 2018. Perbandingan Performa Reverse Proxy Caching Nginx dan Varnish Pada Web Server Apache. *J. Pengemb. Teknol. Inf. dan Ilmu Komput.* 2(4): 1457 – 1463 pp.

W. Puangsaijai and Sutheera Puntheeranurak, "A Comparative Study of Relational Database and Key-Value Database for Big Data Applications," in *International Electrical Engineering Congress*, 2017, no. March, pp. 8–10.

W. Vogels. 2017. Scaling Amazon ElastiCache for Redis with Online Cluster Resizing. https://www.allthingsdistributed.com/2017/11/scaling-amazon-elasticache.html. Diakses pada 23 November 2018.

W. Z. S. Bouchenak, A. Cox, S. Dropsho, S. Mittal. 2006. Middleware 2006. *Internationarel Conference on Middlewa* 4290: 1 – 21 pp.