W

;/--n

**TUGAS AKHIR – KI1502**

**DESAIN DAN IMPLEMENTASI APLIKASI UNTUK MENJAWAB PERMASALAHAN *WHY-NOT ON REACHING K SUBSCRIBERS***

**I PUTU EKA WIRA MAHARDIKA**

**NRP 0511140000025**

**Dosen Pembimbing**

**Henning Titi Ciptaningtyas, S.Kom., M.Kom.**

**Bagus Jati Santoso, S.Kom., Ph.D.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

**

**TUGAS AKHIR – KI1502**

**DESAIN DAN IMPLEMENTASI APLIKASI UNTUK MENJAWAB PERMASALAHAN *WHY-NOT ON REACHING K SUBSCRIBERS***

**I PUTU EKA WIRA MAHARDIKA**

**NRP 0511140000025**

**Dosen Pembimbing**

**Henning Titi Ciptaningtyas, S.Kom., M.Kom.**

**Bagus Jati Santoso, S.Kom., Ph.D.**

**DEPARTEMEN INFORMATIKA**

**Fakultas Teknologi Informasi dan Komunikasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

*[Halaman ini sengaja dikosongkan]*

****

**UNDERGRADUATE THESES – KI1502**

**DESIGN AND IMPLEMENTATION APPLICATION TO ANSWER WHY-NOT ON REACHING K SUBSCRIBERS PROBLEM**

**I PUTU EKA WIRA MAHARDIKA**

**NRP 05111440000025**

**Supervisors**

**Henning Titi Ciptaningtyas, S.Kom., M.Kom.**

**Bagus Jati Santoso, S.Kom., Ph.D.**

**DEPARTMENT OF INFORMATICS**

**Faculty of Information Technology and Communication**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2018**

*[Halaman ini sengaja dikosongkan]*

# 

# LEMBAR PENGESAHAN

[Halaman ini sengaja dikosongkan]

**DESAIN DAN IMPLEMENTASI PENGOLAHAN TOP-K DOMINATING QUERY PADA DATA BERBASIS KELOMPOK**

**Nama Mahasiswa : I Putu Eka Wira Mahardika**

**NRP : 05111440000025**

**Jurusan : Teknik Informatika FTIf-ITS**

**Dosen Pembimbing 1 : Henning Titi Ciptaningtyas, S.Kom., M.Kom.**

**Dosen Pembimbing 2 : Bagus Jati Santoso, S.Kom., Ph.D.**

# 

# Abstrak

***Kata kunci:***

# KATA PENGANTAR

# **DAFTAR ISI**

# DAFTAR GAMBAR

# DAFTAR TABEL

# DAFTAR KODE SUMBER

*[Halaman ini sengaja dikosongkan]*

# 

BAB I   
PENDAHULUAN

Bab ini menjelaskan garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan, batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

Latar Belakang

Penelitian dan pengembangan performa, kualitas, dan penggunaan sistem basis data selama dekade terakhir ini telah mendapat perhatian lebih dalam beberapa tahun terakhir [1]. Berbagai macam algoritma kueri untuk membantu pengguna mencari data dengan nilai atribut terbaik seperti *Top-K Query*, *Skyline Query*, dan *Top-K Dominating Query* mulai dikembangkan. Dengan dikembangkannya berbagai macam algoritma kueri tersebut, maka mulai muncul berbagai permasalahan baru, salah satunya adalah dimana saat pengguna merasa hasil dari suatu kueri tidak sesuai dengan ekspektasi dan mulai bertanya kenapa data tertentu tidak muncul di hasil kueri. Permasalahan tersebut juga dikenal dengan “*Why-Not Question*”. Permasalahan tersebut sudah beberapa kali diangkat untuk menjadi topik jurnal, harapannya dengan dikembangkannya algoritma yang dapat menjawab permasalahan tersebut dengan memperbaiki kueri awal agar dapat mencapai ekspektasi namun tetap memperhitungkan penalti dari perubahannya.

Berangkat dari permasalahan “*Why-Not Question*”, terdapat masalah lain yang dapat diangkat untuk dicari jalan keluarnya. Diketahui bahwa pengguna memiliki sebuah produk dengan suatu nilai tertentu, dan terdapat sebuah dataset yang berisi data preferensi untuk masing-masing konsumen. Kemudian pengguna melakukan kueri untuk mengetahui berapa banyak data konsumen yang kriterianya cocok dengan produk tersebut. Permasalahan akan muncul saat jumlah konsumen hasil kueri tersebut lebih rendah dari ekspektasi pengguna. Permasalahan seperti ini selanjutnya akan disebut dengan istilah “*Why-Not on Reaching k Subscribers*”. Perbedaan mendasar dari permasalahan “*Why-Not Question*” dengan “*Why-Not on Reaching k Subscribers*” terletak pada masukan permasalahan. Pada “*Why-Not Question*” yang menjadi masukan adalah data mana yang hilang dari hasil kueri, sedangkan pada “*Why-Not on Reaching k Subscribers*” adalah jumlah data yang diharapkan dari hasil kueri,

Berdasarkan uraian di atas maka dapat dilihat bahwa dibutuhkan sebuah algoritma untuk menjawab permasalahan “*Why-Not on Reaching k Subscribers*”. Algoritma yang akan dirancang harapannya dapat memperbaiki nilai pada setiap atribut dari produk agar hasil kueri dapat mencapai jumlah data yang diharapkan pengguna. Dalam memperbaiki data produk, nilai penalti juga akan dipertimbangkan sebagai biaya dari suatu perubahan. Dengan begitu diharapkan juga hasil perbaikan nilai atribut juga memiliki nilai penalti serendah mungkin.

Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana mengindeks seluruh data konsumen untuk mendukung solusi atas permasalahan *Why-Not on Reaching k Subscribers*?
2. Bagaimana algoritma yang tepat untuk memperbaiki nilai atribut produk agar mencapai ekspektasi pengguna aplikasi terhadap jumlah konsumen?
3. Bagaimana pembuatan mekanisme penalti yang tepat untuk mengukur kualitas atas solusi perbaikan yang diberikan?

Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Algoritma ini hanya memproses nilai atribut yang bertipe numerik.
2. Implementasi algoritma ini akan menggunakan bahasa pemrograman Phyton.
3. Dataset yang digunakan adalah data *real-life* dan sintetis.
4. Semua nilai atribut pada data akan dinormalisasi terlebih dahulu untuk mempermudah penghitungan nilai penalti.

Tujuan

Tugas akhir ini mempunyai beberapa tujuan, yaitu sebagai berikut:

1. Menemukan formulasi yang tepat dalam mengindeks seluruh data konsumen untuk mendukung solusi atas permasalahan *Why-Not on Reaching k Subscribers*
2. Merancang algoritma yang tepat untuk memperbaiki nilai atribut produk agar mencapai ekspektasi pengguna aplikasi terhadap jumlah konsumen.
3. Membuat mekanisme penalti yang tepat untuk mengukur kualitas atas solusi perbaikan yang diberikan.

Manfaat

Manfaat dari pembuatan tugas akhir ini adalah:

1. Memberikan solusi bagi produsen untuk memperbaiki produknya secara efektif.
2. Memberikan kontribusi dalam pengembangan dan penelitian di bidang data engineering.

Metodologi Pembuatan Tugas Akhir

Tahapan-tahapan yang dilakukan dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. Penyusunan proposal tugas akhir.

Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.

1. Studi literatur

Pada studi literatur ini, akan dipelajari beberapa sejumlah referensi yang akan diperlukan untuk merancang algoritma yaitu mengenai *refined query*.

1. Analisis dan desain perangkat lunak

Pada tahap ini permasalahan ke dimodelkan dalam beberapa bentuk notasi matematika untuk mempermudah proses perancangan desain dari algoritma dan struktur data.

1. Implementasi perangkat lunak

Implementasi algoritma ini akan dibuat dalam bentuk program konsol yang dibangun dengan menggunakan bahasa pemrograman Python. IDE yang akan digunakan adalah PyCharm.

1. Pengujian dan evaluasi

Pengujian dalam algoritma ini akan dilakukan dalam beberapa cara, antara lain:

1. Pengujian Akurasi

Pengujian ini akan berfokus pada ketepatan solusi yang diberikan dalam menjawab permasalahan Why-Not on Reaching k Subscribers.

1. Pengujian Waktu Eksekusi

Pengujian ini akan berfokus pada seberapa lama waktu yang dibutuhkan untuk mengeksekusi algoritma dalam menjawab permasalahan Why-Not on Reaching k Subscribers.

1. Pengujian Penggunaan Memori

Pengujian ini akan berfokus pada pengukuran besarnya memori yang digunakan saat aplikasi dijalankan.

1. Penyusunan buku tugas akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
2. Latar Belakang
3. Rumusan Masalah
4. Batasan Masalah
5. Tujuan
6. Manfaat
7. Metodologi Pembuatan Tugas Akhir
8. Sistematika Penulisan Laporan Tugas Akhir
9. Tinjauan Pustaka
10. Analisis dan Perancangan Sistem
11. Pengujian dan Evaluasi
12. Kesimpulan dan Saran
13. Daftar Pustaka

Sistematika Penulisan Laporan Tugas Akhir

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini.

**Bab I Pendahuluan**

Bab yang berisi mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu metodologi yang digunakan dan sistematika penulisan laporan akhir juga merupakan bagian dari bab ini.

**Bab II Tinjauan Pustaka**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

**Bab III Analisis dan Perancangan Sistem**

Bab ini berisi tentang analisis permasalahan, deskripsi umum sistem, spesifikasi kebutuhan perangkat lunak, lingkungan perancangan, perancangan arsitektur sitem, diagram kelas, dan struktur data.

**Bab IV Implementasi**

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode sumber yang digunakan untuk proses implementasi.

**Bab V Pengujian dan Evaluasi**

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

**Bab VI Kesimpulan dan Saran**

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.



*[Halaman ini sengaja dikosongkan]*

BAB II  
TINJAUAN PUSTAKA

Bab ini menjelaskan teori-teori yang berkaitan dengan Desain Dan Implementasi Aplikasi Untuk Menjawab Permasalahan *Why-Not On Reaching K Subscribers* yang diajukan untuk tugas akhir ini. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap perangkat lunak yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

1. Query Refinement

*Query Refinement* merupakan suatu pendekaatan untuk menghasilkan kueri baru agar dapat menjawab permasalahan pada hasil kueri seperti *Why* dan *Why-Not*, kueri baru tersebut selanjutnya akan disebut *refined query*. Hasil riset terakhir pada [1] tentang pendekatan *query-refinement* menjelaskan 2 kriteria untuk refined query yang baik. Pertama kueri tersebut bersifat *similar* — memiliki sedikit perubahan jika dibandingkan dengan kueri aslinya. Kriteria kedua adalah *refined query* yang baik bersifat *precise* — memiliki sedikit data tambahan pada hasil kuerinya.

Karena jumlah kemungkinan *refined query* terhadap sebuah kueri sangat banyak, jadi diperlukan suatu metrik (alat ukur) untuk membandingkan kualitas dari sebuah *refined query*. Metrik tersebut akan bermanfaat agar algoritma yang memproduksi *refined query* hanya mengembalikan sebuah kueri terbaik sebagai solusinya. Sesuai dengan 2 kriteria dari *refined query* yang baik, maka kriteria tersebut dapat dijadikan sebagai metrik atas kualitas *refined query*.

Yang pertama adalah metrik ketidakmiripan. *Refined query* yang baik seharusnya semirip mungkin dari kueri aslinya. Diketahui sebuah kueri asli *Q* dan refined query *Q’*, kita bandingkan tingkat kemiripan antara *Q* dan *Q’* dengan mengukur jarak/selisih dari perubahan yang paling minimal. Dengan demikian kueri tersebut dikatakan lebih mirip (atau lebih tidak mirip) terhadap satu sama lain jika jarak perbahannya kecil (begitu pula sebaliknya). Karena hasil dari *Q* dan *Q’* bersifat *union-compatible* (seluruh atribut dalam klausa select pada *Q* dan *Q’* adalah sama), kita hanya mempertimbangkan klausa from dan where sebagai operator perubahan untuk mengubah *Q* menjadi *Q’*. Empat kunci dalam operasi perubahan kueri adalah: (*O1*) ubah beberapa nilai konstan di predikat *select* pada klausa *where*, (*O2*) tambahkan sebuah predikat select pada klausa *where*, (*O3*) tambahkan/hilangkan predikat join pada klausa *where*, dan (*O4*) tambahkan/hilangkan *relation* pada klausa *from*. Perlu diketahui bahwa tidak ada perubahan operator secara eksplisit saat menghilangkan sebuah predikat *select* seperti yang dimodelkan pada *O1*. Proses menghilangkan predikat *select* secara efisiensi setara dengan mengubah jarak dari nilai *select* untuk menutupi domain dari seluruh atribut. Selanjutnya, saat *O4* digunakan untuk menghilangkan relasi *Ri* pada klausa *from*, seluruh predikat *select* dan *join* yang berasosiasi dengan *Ri* juga dihilangkan sebagai bagian dari operasi perubahan.

Mari nyatakan *wi* sebagai biaya untuk operasi perubahan Oi, . Dengan begitu dapat diasumsikan bahwa . Mari nyatakan *ni* sebagai total dari operasi *Oi* yang digunakan untuk merubah *Q* menjadi *Q’*, . Jarak perubahan untuk transformasi ini diketahui sebagai . Kita mengacu pada jarak perubahan minimum untuk mengubah *Q* menjadi *Q’* sebagai ukuran untuk perbedaan antara *Q* dan *Q’*, yang dapat dihitung secara efisien untuk *Q* dan *Q’*.

Yang kedua adalah metrik ketidaktepatan. *Refined query* yang baik seharusnya hanya menghasilkan data yang dihasilkan pada kueri aslinya dan dengan tambahan data yang memang diinginkan (dalam hal ini data yang hilang pada permasalahan *Why-Not*). Data-data yang tidak diinginkan untuk tampil di dalam hasil kueri seharusnya dapat diminimalisir.

*Skyline refined query* adalah *refined query* yang memiliki nilai terbaik (paling rendah) pada metrik ketidakmiripan maupun pada metrik ketidaktepatan. Dari seluruh hasil *refined query* yang mungkin akan menjadi solusi, maka *Skyline refined query* adalah solusi terbaik [1].

1. Kueri Berbasis Preferensi

Preferensi telah menjadi peran utama dalam berbagai disiplin ilmu, termasuk pada ilmu computer seperti kecerdasan buatan, interaksi manusia dan komputer, dan basis data. Pada basis data, representasi dari preferensi memiliki dua kategori utama yaitu preferensi kualitatif dan preferensi kuantitatif. Preferensi kualitatif dispesifikasikan menggunakan predikat biner yang dapat membandingkan antara suatu data dengan data lainnya secara relatif. Sedangkan preferensi kuantitatif diekspresikan dengan menetapkan skor pada suatu data atau pada elemen kueri yang mendeskripsikan sebuah kumpulan data, yang dimana skor tersebut mengekspresikan derajat ketertarikan (*degree of interest*). Pada preferensi kuantitatif, suatu data *a* dikatakan lebih disukai (*prefered*) apabila skor untuk data *a* lebih tinggi dari skor untuk data *b* [2].

1. Dominasi Data

Dominasi data adalah suatu kondisi dimana sebuah objek memiliki nilai atribut lebih baik secara keseluruhan objek. Sebagai contoh, objek *a* dikatakan mendominasi objek *b* jika seluruh nilai atribut pada *a* tidak ada yang lebih buruk dari *b* dan setidaknya satu atribut pada *a* bernilai lebih baik daripada *b* [3].

1. Graf

Dalam ilmu matematika dan ilmu komputer, terutama pada teori graf, graf adalah sebuah struktur dari sekumpulan objek yang memiliki relasi dengan objek lainnya. Objek tersebut dalam abstraksi matematika disebut dengan *vertex* (disebut juga simpul, *node*, atau titik) dan setiap pasangan relasi antar *vertex* dinamakan *edge* (disebut juga garis atau busur). Biasanya, graf digambarkan dalam bentuk diagram sebagai serangkaian titik sebagai *vertex*, dan dihubungkan dengan garis sebagai *edge* [4].

1. Python

Python adalah bahasa pemrograman umum tingkat tinggi yang dapat diterapkan ke berbagai kelas masalah. Bahasa ini dilengkapi dengan pustaka standar besar yang mencakup area seperti pemrosesan string (*regular expression*, Unicode, menghitung perbedaan antar file), protokol Internet (HTTP, FTP, SMTP, XML-RPC, POP, IMAP, pemrograman CGI), rekayasa perangkat lunak (pengujian unit, pencatatan, pembuatan profil, penguraian kode Python), dan antarmuka sistem operasi (panggilan sistem, sistem file, soket TCP / IP) [5].

1. Apache Spark

Apache Spark adalah sistem komputasi terdistribusi yang cepat dan dapat digunakan untuk tujuan umum. Apache Spark menyediakan berbagai *high-level* API pada Java, Scala, Python, dan R. Apache Spark juga merupakan *engine* yang telah teroptimalisasi dan mendukung untuk eksekusi graf. Apache Spark juga mendukung fitur dengan level yang lebih tinggi seperti Spark SQL untuk SQL dan pemrosesan data terstruktur, MLlib untuk *machine learning*, GraphX untuk pemrosesan graf, dan Spark Streaming [6].

1. PHP

PHP (akronim untuk PHP: Hypertext Preprocessor) adalah bahasa scripting open source yang digunakan secara luas dan untuk tujuan umum yang sangat cocok untuk pengembangan web dan dapat dimasukkan ke dalam HTML.

Apa yang membedakan PHP dari sesuatu seperti *client-side* JavaScript adalah bahwa kode dijalankan di server, menghasilkan HTML yang kemudian dikirim ke *client*. *Client* akan menerima hasil menjalankan skrip tersebut, tetapi tidak akan tahu apa *source code-*nya. Kita bahkan dapat mengkonfigurasi server web untuk memproses semua file HTML dengan PHP, dan kemudian benar-benar tidak ada cara bagi pengguna untuk mengetahui apa yang kita miliki [7].

1. Codeigniter

CodeIgniter adalah *toolkit* untuk orang yang membangun aplikasi web menggunakan PHP. Tujuannya adalah untuk memungkinkan kita mengembangkan proyek lebih cepat daripada yang kita bisa jika menulis kode dari awal, dengan menyediakan kumpulan pustaka yang kaya untuk tugas-tugas yang sering dibutuhkan, serta antarmuka yang sederhana dan struktur logis untuk mengakses pustaka ini. CodeIgniter memungkinkan kita secara kreatif fokus pada proyek Anda dengan meminimalkan jumlah kode yang diperlukan untuk tugas yang diberikan.

CodeIgniter menggunakan pendekatan Model-View-Controller, yang memungkinkan pemisahan antara logika dan presentasi. Ini sangat bagus untuk proyek di mana desainer bekerja dengan file template, karena kode yang berisi file ini akan diminimalkan. Codeigniter mendeskripsikan MVC secara lebih rinci pada halamannya sendiri [8].

BAB III  
PERANCANGAN PERANGKAT LUNAK

Pada bab ini akan dijelaskan mengenai perancangan sistem perangkat lunak yang akan dibuat. Perancangan akan dibagi menjadi tiga proses utama, yaitu:

1. *Data preprocessing* untuk mengindeks data berdasarkan dominasinya terhadap data lainnya menggunakan struktur data graf.
2. Penerapan algoritma untuk menjawab permasalahan *Why-Not on Reaching k Subscribers*.
3. Penerapan visualisasi menggunakan website dan pustaka xxx untuk menampilkan struktur data graf yang dibentuk.
4. Desain Metode Secara Umum
5. Dataset