

BAB 2

TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Penelitian terdahulu berjudul "Penerapan Metode Optimasi *Exponential Smoothing* Untuk Peramalan Debit". *Exponential smoothing* merupakan teknik yang mudah untuk diterapkan dan sangat efektif sebagai peramalan. Metode yang digunakan dalam peramalan data debit yaitu dengan cara mengoptimasi nilai *error* atau kesalahan, dengan meminimumkan nilai *error* atau kesalahan maka akan didapatkan hasil ramalan yang maksimum, sehingga hasil ramalan akan mendekati serial data hasil pengamatan dilapangan. Optimasi dilakukan dengan cara menggunakan lingo 11 dan hasil yang didapat cukup baik [1].

Penelitian terdahulu berjudul "Forecasting Volume Produksi Tanaman Pangan, Tanaman Perkebunan Rakyat Kab. Magelang Dengan Metode *Exponential Smoothing* Berbantu Minitab". Metode peramalan *exponential smoothing* merupakan model ramalan data berkala (*time series*) yang digunakan untuk peramalan. Permasalahan pada penelitian ini adalah bagaimana menggunakan metode *exponential smoothing* untuk meramalkan volume produksi tanaman pangan dan produksi perkebunan rakyat Kabupaten Magelang dengan Minitab yang berupa ramalan volume produksi tanaman pangan dan tanaman perkebunan rakyat Kabupaten Magelang menggunakan metode *exponential smoothing*. Tujuan penelitian ini adalah mengetahui penggunaan metode *exponential smoothing* untuk peramalan volume produksi tanaman pangan dan tanaman perkebunan Rakyat Kabupaten Magelang menggunakan Minitab. Manfaat penelitian ini bagi BPS Kabupaten Magelang yaitu dalam meramal atau memprediksi volume produksi dapat menggunakan metode *exponential smoothing*. Metode pengumpulan data menggunakan teknik pengumpulan data sekunder. Dengan metode literatur yaitu informasi diperoleh dari buku, referensi dan karya ilmiah.

Metode dokumentasi dalam pengambilan data volume produksi tanaman pangan dan perkebunan rakyat Kabupaten Magelang tahun 1996-2010. Dengan metode *double exponential smoothing* pada volume produksi tanaman pangan dan tanaman perkebunan rakyat didapatkan nilai MAPE dengan $\alpha = 0,1$ lebih kecil bila dibandingkan dengan metode *single exponential smoothing* dengan nilai ramalan masing-masing yaitu 4.083.112 ton untuk volume produksi tanaman pangan dan 27.851,7 ton untuk volume produksi tanaman perkebunan rakyat. Nilai ramalan volume produksi tanaman pangan dan volume perkebunan rakyat Kabupaten Magelang pada tahun 2011 yaitu masing-masing 4.083.112 ton dan 27.851,7 ton. Saran dari penelitian ini adalah untuk memprediksi beberapa besar pelanggan di tahun mendatang akan lebih baik jika tidak menggunakan perhitungan manual tetapi dengan menggunakan program komputer Minitab atau program yang lain sehingga akurasinya lebih tepat [2].

Tabel 2.1 Penelitian Terkait

No	Nama Peneliti dan Tahun	Masalah	Metode	Hasil
1.	Budi Santosa, Suharyanto, Djoko Legono, 2008	Penerapan Metode Optimasi <i>Exponential Smoothing</i> Untuk Peramalan Debit	<i>exponential smoothing</i>	Hasil analisis dengan faktor $\alpha = 0,1, 0,5, 0,95$ didapatkan hasil optimasi mempunyai kesalahan (<i>error</i>) yang paling kecil
2.	Nur Sidik, 2010	Peramalan Tanaman Perkebunan Rakyat Kab. Magelang Dengan Metode <i>Exponential</i>	<i>exponential smoothing</i>	Ramalan tanaman pangan dan tanaman perkebunan rakyat tahun 2011 masing-masing adalah 4083112 ton dan 27851,7 ton

No	Nama Peneliti dan Tahun	Masalah	Metode	Hasil
		<i>Smoothing</i> Berbantu Minitab		

Dari dua penelitian terkait yang telah dijabarkan pada tabel 2.1 dapat disimpulkan bahwa proses peramalan debit dan tanaman perkebunan rakyat dapat menggunakan metode *exponential smoothing* dan didapatkan hasil optimasi mempunyai kesalahan (*error*) yang paling kecil. Dalam hal ini, metode *exponential smoothing* juga tentunya dapat digunakan untuk membantu penulis dalam melakukan peramalan pendapatan Pajak Kendaraan Bermotor (PKB) pada tahun 2016.

2.2 Landasan Teori

2.2.1 Data Mining

Data Mining adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning* untuk mengekstrasi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai database besar [3]. Istilah data mining memiliki hakikat sebagai disiplin ilmu yang tujuan utamanya adalah untuk menemukan, menggali, atau menambang pengetahuan dari data atau informasi yang dimiliki. Data mining sering juga disebut sebagai *Knowledge Discovery in Database* (KDD). KDD adalah kegiatan yang meliputi pengumpulan, pemakaian data, historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar.

1. Metode Pelatihan

Secara garis besar metode pelatihan yang digunakan dalam teknik-teknik data mining dibedakan ke dalam dua pendekatan, yaitu:

a. *Unsupervised Learning*

Metode ini diterapkan tanpa adanya latihan (*training*) dan tanpa ada guru (*teacher*). Guru di sini adalah label dari data.

b. *Supervised Learning*

Metode belajar dengan adanya latihan dan pelatih. Dalam pendekatan ini, untuk menemukan fungsi keputusan, fungsi pemisah atau fungsi regresi, digunakan beberapa contoh data yang mempunyai output atau label selama proses *training*.

2. Pengelompokan *Data Mining*

Ada beberapa teknik yang dimiliki *data mining* berdasarkan tugas yang biasa dilakukan, yaitu:

a. Deskripsi

Para peneliti biasanya mencoba menemukan cara untuk mendeskripsikan pola dan tren yang tersembunyi dalam data.

b. Estimasi

Estimasi mirip dengan klasifikasi, kecuali variable tujuan yang lebih ke arah numeric dari pada kategori.

c. Prediksi

Prediksi memiliki kemiripan dengan estimasi dan klasifikasi. Hanya saja, prediksi hasilnya menunjukkan sesuatu yang belum terjadi (mungkin terjadi di masa depan).

d. Klasifikasi

Dalam klasifikasi variabel, tujuan bersifat kategorik. Misalnya, Pengklasifikasian pendapatan dalam tiga kelas, yaitu pendapatan tinggi, pendapatan sedang, dan pendapatan rendah.

e. *Clustering*

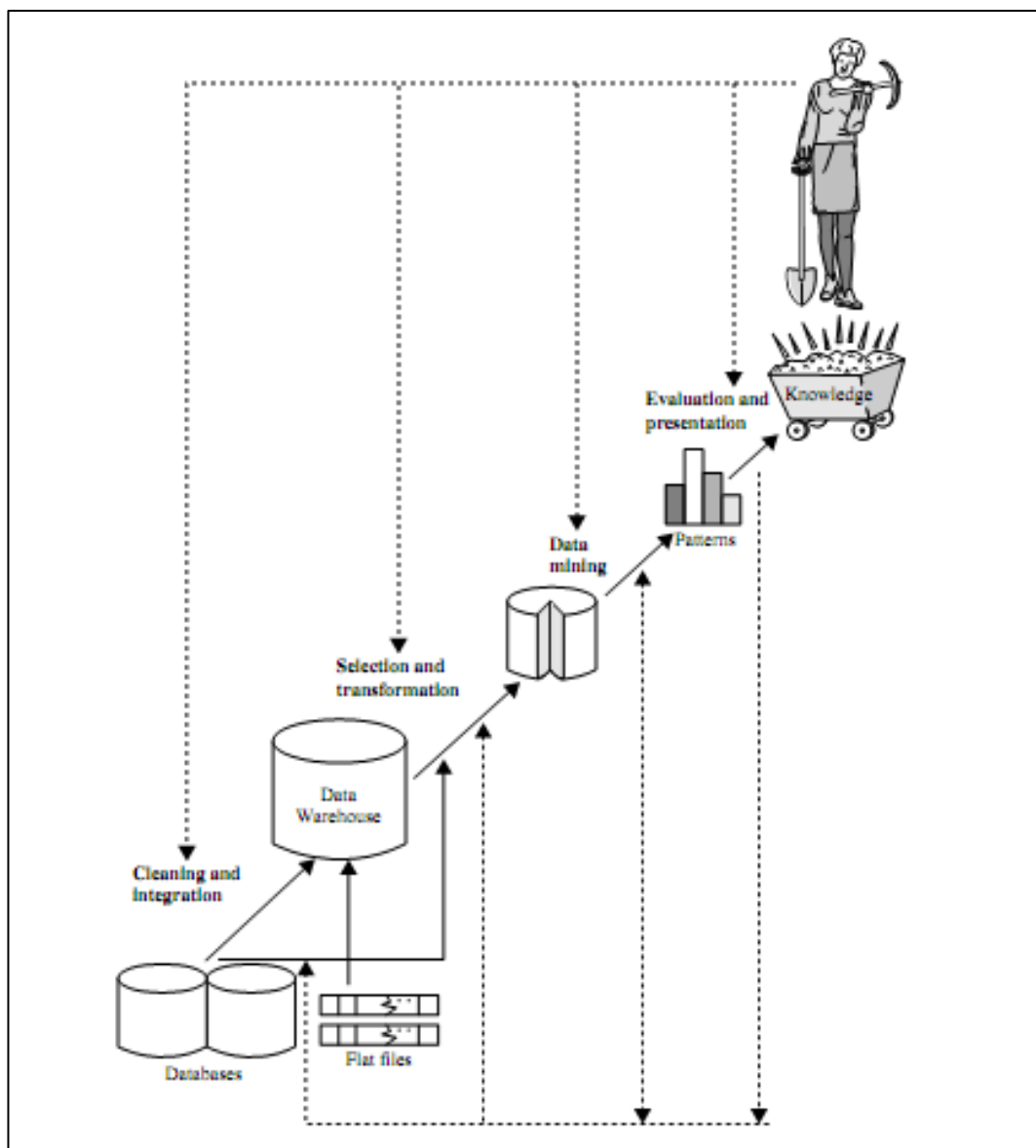
Clustering lebih ke arah pengelompokan *record*, pengamatan, atau kasus dalam kelas yang memiliki kemiripan.

f. Asosiasi

Mengidentifikasi hubungan antara berbagai peristiwa yang terjadi pada satu waktu.

3. Tahap-tahap Data Mining

Sebagai suatu rangkaian proses, data *mining* dapat dibagi menjadi beberapa tahap proses yang diilustrasikan pada Gambar 2.1. Tahap-tahap tersebut bersifat interaktif, pemakai terlibat langsung atau dengan perantaraan *knowledge base*.



Gambar 2.1 Tahap-Tahap Data Mining [4]

Tahap-tahap data *mining* adalah sebagai berikut:

a. Pembersihan data (*data cleaning*)

Pembersihan data merupakan proses menghasilkan *noise* dan data yang tidak konsisten atau data tidak relevan.

b. Integrasi data (*data integration*)

Integrasi data merupakan penggabungan data dari berbagai *database* ke dalam suatu *database* baru.

c. Seleksi data (*data selection*)

Data yang ada pada *database* sering kali tidak semuanya dipakai, oleh karena itu hanya data yang sesuai untuk dianalisis yang akan diambil dari *database*.

d. Transformasi data (*data transformasi*)

Data diubah atau digabung ke dalam format yang sesuai untuk diproses dalam *data mining*.

e. Proses *mining*

Merupakan suatu proses utama saat metode diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data.

f. Evaluasi pola (*pattern evaluation*)

Untuk mengidentifikasi pola-pola menarik ke dalam *knowledge based* yang ditemukan.

g. Presentasi pengetahuan (*knowledge presentation*)

Merupakan visualisasi dan penyajian pengetahuan mengenai metode yang digunakan untuk memperoleh pengetahuan yang diperoleh pengguna.

2.2.2 Peramalan/Prediksi

Forecasting atau peramalan atau prediksi adalah suatu usaha untuk meramalkan keadaan di masa mendatang melalui pengujian keadaan di masa lalu. Dalam kehidupan sosial segala sesuatu itu tidak pasti, sukar untuk diperkirakan secara tepat. *Forecast* yang dibuat selalu diupayakan agar dapat meminimumkan pengaruh ketidakpastian ini terhadap perusahaan. Dengan kata lain *forecasting* bertujuan mendapatkan *forecast* yang bisa meminimumkan kesalahan meramal (*forecast error*) yang biasanya diukur dengan *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE) dan sebagainya [5].

2.2.2.1 Jenis Peralaman

Jenis-jenis peralaman yaitu [5]:

1. Berdasarkan Jangka Waktu

Peramalan berdasarkan waktunya dibedakan menjadi dua, yaitu peramalan jangka panjang yang biasanya dilakukan oleh para pemimpin puncak suatu perusahaan dan bersifat umum, sedangkan peramalan jangka pendek yaitu yang biasanya dilakukan oleh pemimpin pada tingkat menengah maupun bawah dan lebih bersifat operasional.

2. Berdasarkan Metode

Berdasarkan metode dibedakan menjadi dua yaitu:

- a. Metode Kualitatif

Peramalan dengan metode ini adalah peramalan yang lebih didasarkan atas intuisi dan penilaian orang yang melakukan peramalan dari pada pemanipulasian data historis yang tersedia.

- b. Metode Kuantitatif

Peramalan dengan metode kuantitatif adalah peramalan yang didasarkan atas pemanipulasian data yang tersedia secara memadai serta tanpa intuisi maupun penilaian subyektif dari orang yang melakukan peramalan.

2.2.2.2 Proses Peramalan

Proses peramalan biasanya terdiri dari langkah-langkah sebagai berikut [5]:

1. Penentuan Tujuan

Analisis membicarakan dengan para pembuat keputusan dalam perusahaan untuk mengetahui apa kebutuhan-kebutuhan mereka dan menentukan:

- a. Variabel-variabel apa yang akan diestimasi.
- b. Siapa yang akan menggunakan hasil peramalan.
- c. Untuk tujuan-tujuan apa hasil peramalan digunakan.
- d. Estimasi jangka panjang atau jangka pendek yang diinginkan.
- e. Derajat ketepatan estimasi yang diinginkan.
- f. Kapan estimasi dibutuhkan.
- g. Bagian-bagian peramalan yang diinginkan, seperti peramalan untuk kelompok pembeli, kelompok produk atau daerah geografis.

2. Pengembangan Model

Setelah tujuan ditetapkan, langkah berikutnya adalah mengembangkan model, yang merupakan penyajian secara lebih sederhana sistem yang dipelajari. Dalam peramalan, model adalah suatu kerangka analitik yang apabila dimasukan data masukan menghasilkan estimasi penjualan di waktu mendatang (atau variabel apa saja yang akan diramal). Analisis hendaknya memilih suatu model yang menggambarkan secara realistis perilaku variabel-variabel yang dipertimbangkan.

3. Pengujian Model

Sebelum diterapkan, model biasanya diuji untuk menentukan tingkat akurasi, validitas, dan reliabilitas yang diharapkan. Ini sering mencakup penerapannya pada data historis, dan penyiapan estimasi untuk tahun-tahun sekarang dengan data nyata yang tersedia. Nilai suatu model ditentukan oleh derajat ketepatan hasil peramalan data aktual.

4. Penerapan Model

Setelah pengujian, analisis merupakan model dalam tahap ini, data historis dimasukan dalam model untuk menghasilkan suatu ramalan.

5. Revisi dan Evaluasi

Ramalan-ramalan yang telah dibuat harus senantiasa diperbaiki dan ditinjau kembali. Perbaikan mungkin perlu dilakukan karena adanya perubahan-perubahan dalam perusahaan atau lingkungannya, seperti tingkat harga produk perusahaan, karakteristik-karakteristik produk, pengeluaran-pengeluaran pengiklanan, tingkat pengeluaran pemerintah, kebijakan moneter dan kemajuan teknologi. Evaluasi, di pihak lain, merupakan perbandingan ramalan-ramalan dengan hasil nyata untuk menilai ketepatan penggunaan suatu metodologi atau teknik peramalan. Langkah ini diperlukan untuk menjaga kualitas estimasi-estimasi di waktu yang akan datang.

2.2.3 Metode *Single Exponential Smoothing*

Metode *exponential smoothing* merupakan pengembangan dari metode *moving average*. Dalam metode ini peramalan dilakukan dengan mengulang perhitungan secara terus-menerus dengan menggunakan data terbaru, setiap data terbaru diberi bobot yang lebih besar. Tujuan dari metode ini adalah menentukan nilai α yang meminimumkan MSE pada kelompok penguji [6].

Pada metode ini bobot yang diberikan pada data yang ada sebesar α untuk data yang terbaru, $\alpha(1 - \alpha)$ untuk data yang lama, $\alpha(1 - \alpha)^2$ untuk data yang lebih lama dan seterusnya. Besarnya α adalah antara 0 dan 1, semakin mendekati 1 berarti data terbesar lebih diperhatikan. Secara sistematis besarnya forecast adalah

$$St+1 = \alpha Xt + 1 - \alpha St \quad (2.1)$$

Dengan

$St+1$ = nilai peramalan ke t+1

Xt = data aktual ke t

α = parameter dengan nilai antara 0 sampai 1

St = nilai peramalan ke t

2.2.3.1 Menghitung kesalahan Ramalan

Kesalahan *error* dapat dihitung dengan menggunakan *mean absolute error* dan *mean square error*. *Mean absolute error* adalah rata-rata nilai *absolute* dari kesalahan meramal (tidak dihiraukan tanda positif atau negatifnya) .

$$MAE = \frac{\sum |X_t - F|}{n} \quad (2.2)$$

Mean square error adalah kuadrat rata-rata kesalahan peramalan

$$MSE = \frac{\sum |X_t - F|^2}{n} \quad (2.3)$$

Dengan

X_t : data sebenarnya terjadi

F_t : data ramalan dihitung dari model yang digunakan pada waktu atau tahun t

n : banyak data hasil ramalan

Prinsip dalam menghitung kesalahan peramalan (*forecast error*) yaitu model yang baik adalah model yang mempunyai kesalahan *error* paling kecil dari terhadap data pengamatan yang sebenarnya di lapangan.

2.2.4 Pajak

Pajak adalah iuran rakyat kepada kas Negara berdasarkan undang-undang yang dapat dipaksakan dengan tidak mendapat jasa timbal yang langsung dapat ditunjukkan dan digunakan untuk membayar pengeluaran umum. Wajib Pajak adalah orang pribadi atau Badan, meliputi pembayar pajak, pemotong pajak, dan pemungut pajak yang mempunyai hak dan kewajiban perpajakan sesuai dengan ketentuan Peraturan Perundang-undangan perpajakan. Orang Pribadi merupakan

subjek pajak yang bertempat tinggal atau berada di Indonesia ataupun di luar Indonesia [7]. Pajak merupakan sumber penerimaan Negara yang mempunyai 2 fungsi yaitu:

1. Fungsi anggaran (*budgetair*) sebagai sumber dana bagi pemerintah, untuk membiayai pengeluaran-pengeluarannya.
2. Fungsi mengatur (*regulerend*) sebagai alat pengatur atau melaksanakan pemerintah dalam bidang sosial ekonomi.

Sistem pemungutan pajak dapat dibagi menjadi 3 sistem yaitu:

1. *Official Assessment System*

Adalah suatu sistem pemungutan yang memberi wewenang kepada pemerintah (fiskus) untuk menentukan besarnya pajak yang terutang oleh Wajib Pajak.

2. *Self Assessment System*

Adalah suatu sistem pemungutan yang memberi wewenang sepenuhnya kepada Wajib Pajak untuk menghitung, memperhitungkan, membayar, dan melaporkan sendiri besarnya pajak yang terutang.

3. *With Holding System*

Adalah suatu sistem pemungutan yang memberi wewenang kepada pihak ketiga (bukan fiskus dan bukan Wajib Pajak yang bersangkutan) untuk menentukan besarnya pajak yang terutang oleh Wajib Pajak.

2.2.5 Pajak Kendaraan Bermotor (PKB)

Pajak Kendaraan Bermotor adalah pajak atas kepemilikan dan/atau penguasaan kendaraan bermotor. Sesuai Undang-undang Republik Indonesia Nomor 28 Tahun 2009 tentang Pajak Daerah dan Retribusi Daerah, Pajak Kendaraan Bermotor (PKB) merupakan jenis pajak yang kewenangannya ada pada Provinsi [8].

OBJEK PKB

Menurut Peraturan Daerah Provinsi Jawa Tengah Nomor 2 Tahun 2011 tentang Pajak Daerah Provinsi Jawa Tengah, objek Pajak Kendaraan Bermotor adalah kepemilikan dan/atau penguasaan kendaraan bermotor yang terdaftar di Provinsi Jawa Tengah.

Pengertian Kendaraan Bermotor adalah kendaraan bermotor beroda beserta gandengannya, yang dioperasikan di semua jenis jalan darat dan kendaraan bermotor yang dioperasikan di air dengan ukuran isi kotor GT 5 (lima Gross Tonnage) sampai dengan dengan GT 7 (tujuh Gross Tonnage).

SUBJEK PKB

Subjek Pajak Kendaraan Bermotor adalah orang pribadi, Badan atau Instansi Pemerintah (Pemerintah, TNI/POLRI, Pemerintah Provinsi, dan Pemerintah Kabupaten/Kota), yang memiliki dan/atau menguasai kendaraan bermotor.

WAJIB PKB

1. Wajib PKB adalah orang pribadi, Badan atau Instansi Pemerintah (Pemerintah, TNI/POLRI, Pemerintah Provinsi, dan Pemerintah Kabupaten/Kota) yang memiliki kendaraan bermotor. Pengertian "memiliki" adalah sebagaimana diatur dalam pasal 570 Kitab Undang-Undang Hukum Perdata (KUH Perdata), sedangkan Pengertian "menguasai" adalah sebagaimana diatur dalam pasal 1977 KUH Perdata.
2. Yang bertanggung jawab terhadap pembayaran PKB adalah :
 - a. untuk orang pribadi adalah orang yang bersangkutan, kuasanya dan/ atau ahli warisnya;
 - b. untuk badan/organisasi adalah pengurus atau kuasanya;
 - c. untuk Instansi Pemerintah adalah pejabat pengguna anggaran atau kuasa pengguna anggaran.
3. Wajib Pajak baik orang pribadi, Badan atau Instansi Pemerintah (Pemerintah, TNI/POLRI, Pemerintah Daerah, dan Pemerintah Kabupaten/Kota) yang menerima penyerahan kendaraan bermotor, jumlah pajaknya sebagian atau

seluruhnya belum dilunasi oleh pemilik lama, maka pihak yang menerima penyerahan tersebut bertanggung jawab terhadap pelunasan pajaknya (tanggung renteng).

Dasar pengenaan PKB dihitug sebagai perkalian 2 (dua) unsur pokok, yaitu:

1. Nilai Jual Kendaraan Bermotor (NJKB); dan
2. Bobot yang mencerminkan secara relatif tingkat kerusakan jalan dan/atau pencemaran lingkungan akibat penggunaan kendaraan bermotor.

Tarif PKB ditetapkan sebesar:

1. 1,5 % (satu koma lima persen) untuk kepemilikan pertama kendaraan bermotor pribadi dan badan;
2. 1,0 % (satu koma nol persen) untuk kendaraan bermotor angkutan umum;
3. 0,5 % (nol koma lima persen) untuk kendaraan bermotor ambulans, pemadam kebakaran, sosial keagamaan, lembaga sosial dan keagamaan, Instansi Pemerintah (Pemerintah, TNI/POLRI, Pemerintah Provinsi dan Pemerintah Kabupaten/Kota);
4. 0,2 % (nol koma dua persen) untuk kendaraan bermotor alat-alat berat dan alat-alat besar.

Tarif PKB Progresif ditetapkan sebesar:

1. 2 % (dua persen) untuk kepemilikan kedua;
2. 2,5 % (dua koma lima persen) untuk kepemilikan ketiga;
3. 3 % (tiga persen) untuk kepemilikan keempat;
4. 3,5 % (tiga koma lima persen) untuk kepemilikan kelima dan seterusnya.

Cara untuk menghitung besarnya PKB terutang dihitug berdasarkan perkalian antara tarif PKB dengan dasar pengenaan PKB:

1. $1,5\% \times (\text{NJKB} \times \text{Bobot})$ untuk kepemilikan pertama kendaraan bermotor orang pribadi dan badan.
2. $1\% \times (\text{NJKB} \times \text{Bobot})$ untuk kendaraan bermotor angkutan umum.
3. $0,5\% \times (\text{NJKB} \times \text{Bobot})$ untuk kendaraan bermotor ambulans, pemadam kebakaran, sosial keagamaan, lembaga sosial dan keagamaan, Instansi Pemerintah (Pemerintah, TNI/POLRI, Pemerintah Provinsi dan Pemerintah Kabupaten/Kota).
4. $0,2\% \times (\text{NJKB} \times \text{Bobot})$ untuk kendaraan bermotor alat-alat berat dan alat-alat besar.
5. $2\% \times (\text{NJKB} \times \text{Bobot})$ untuk kepemilikan kendaraan bermotor kedua.
6. $2,5\% \times (\text{NJKB} \times \text{Bobot})$ untuk kepemilikan kendaraan bermotor ketiga.
7. $3\% \times (\text{NJKB} \times \text{Bobot})$ untuk kepemilikan kendaraan bermotor keempat.
8. $3,5\% \times (\text{NJKB} \times \text{Bobot})$ untuk kepemilikan kendaraan bermotor kelima dan seterusnya.

Nilai jual kendaraan bermotor adalah Nilai Jual Kendaraan Bermotor yang berlaku yang ditetapkan dengan Peraturan Gubernur Jawa Tengah yang berpedoman pada Nilai Jual Kendaraan bermotor yang ditetapkan Peraturan Menteri Dalam Negeri. Sedangkan dasar pengenaan PKB adalah perkalian antara Tarif PKB dengan NJKB dan dikalikan dengan Bobot.

Berdasarkan jenisnya kendaraan bermotor dibedakan dari:

1. Mobil penumpang terdiri dari sedan, jeep, minibus, bemo dan sejenisnya.
2. Mobil bus terdiri dari bus, microbus dan sejenisnya.
3. Mobil barang/beban terdiri dari pick-up, pick up box delivery van, blind van, double cabin, light truck, dump truck, truck, head tractor, mixer, prime mover, trailer dan sejenisnya.

4. Kendaraan alat-alat berat dan alat-alat besar terdiri dari tracktor/forklift, wheel tracktor, bulldozer, excavator, grader, wheel loader, shovel, skider, crane, roller, compactor, vibrator dan sejenisnya.
5. Sepeda motor terdiri dari sepeda motor roda 2 (dua), sepeda motor roda 3 (tiga) dan scooter.

2.2.6 UML

Unified Modelling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. UML (*Unified Modelling Language*) adalah sekumpulan pemodelan konvensi yang digunakan untuk menentukan atau menggambarkan sebuah sistem perangkat lunak dalam kaitannya dengan objek. UML dapat juga diartikan sebuah bahasa grafik standar yang digunakan untuk memodelkan perangkat lunak berbasis objek.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*). Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia.

Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan.



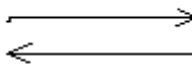
2.2.6.1 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* adalah sebuah diagram yang menjelaskan apa yang akan dilakukan oleh sistem yang akan dibangun dan siapa yang berinteraksi dengan sistem. [9]

Use case diagram dapat sangat membantu bila sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend* *use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Komponen atau simbol yang digunakan dalam *use case diagram* meliputi:

Tabel 2.2 Simbol Use Case [9]

No.	Komponen	Arti	Keterangan
1.		Actor	Manusia, user, pengguna sistem yang berhubungan secara langsung dengan sistem
2.		Proses	Perilaku yang ditunjukkan atau dilakukan oleh actor.
3.		Relasi	Penghubung antara actor dengan proses atau proses dengan proses.

Dalam relasi ini terdapat dua komponen yaitu :

1. *Include*

Suatu bagian dari elemen (yang ada di garis tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah).

2. *Extend*

Menunjukkan suatu bagian dari elemen di garis tanpa panah bisa disisipkan ke dalam elemen yang ada di garis dengan panah.

2.2.6.2 Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. [9]

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.




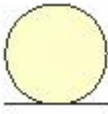
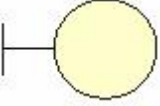

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

2.2.6.3 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. [9]

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*.

Tabel 2.3 Simbol *Sequence Diagram* [9]

NO	GAMBAR	NAMA	KETERANGAN
1		<i>LifeLine</i>	Menggambarkan objek <i>entity</i> , antarmuka yang saling berinteraksi.
2		<i>Message</i>	Menggambarkan pengiriman pesan
3		<i>Actor</i>	Menggambarkan orang yang sedang berinteraksi dengan sistem
4		<i>Entity Class</i>	Menggambarkan hubungan kegiatan yang akan dilakukan
5		<i>Boundary Class</i>	Menggambarkan sebuah penggambaran dari form
6		<i>Control Class</i>	Menggambarkan penghubung antara <i>boundary</i> dengan tabel

2.2.6.4 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. Pada *class diagram* terdapat beberapa komponen yang digunakan [9]:

1. *Object*

Object adalah “benda”, secara fisik atau konseptual yang dapat ditemui di sekeliling kita, misalnya dokumen, *hardware*, *software*, manusia. Contoh: mekanik, mobil.

2. *Class*

Class adalah definisi umum (pola, *template* atau cetak biru) untuk himpunan objek sejenis, kelas menetapkan spesifikasi perilaku dan objek-objek tersebut.

3. *Attributes* (Atribut)

Attributes merupakan karakteristik suatu objek dan biasanya ditandai dengan kata sifat dan “*frase*” milik.

4. *Method* (Metode)

Method merupakan subprogram yang tergantung yang bersama–sama dengan atribut.

Class dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*. Hubungan antar *class* yaitu [9]:

1. Asosiasi

Hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah query antar *class*.

2. Agregasi

Hubungan yang menyatakan bagian (“terdiri atas..”).

3. Pewarisan

Hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.

4. Hubungan Dinamis




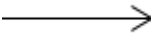

Rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada *class* lain.

2.2.6.5 State Diagram

State menggambarkan perubahan kondisi objek sebagai tanggapan atas perubahan situasi / kejadian. Diagram *state* memerlukan [9]:

1. Objek.
2. Kondisi / Status Objek.
3. Transisi yang menyebabkan perubahan kondisi objek.
4. Aktifitas yang terkait dengan kondisi tertentu.

Tabel 2.4 Simbol *State Diagram* [9]

NO	GAMBAR	NAMA	KETERANGAN
1		<i>State</i>	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		<i>Initial Pseudo State</i>	Bagaimana objek dibentuk atau diawali
3		<i>Final State</i>	Bagaimana objek dibentuk dan dihancurkan
4		<i>Transition</i>	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.

2.2.7 PHP

PHP sendiri sebenarnya merupakan singkatan dari *Hypertext Preprocesso*”, yang merupakan sebuah bahasa scripting tingkat tinggi yang dipasang pada dokumen HTML. Sebagian besar sintaks dalam PHP mirip dengan bahasa C, Java dan Perl,

namun pada PHP ada beberapa fungsi yang lebih spesifik. Sedangkan tujuan utama dari penggunaan bahasa ini adalah untuk memungkinkan perancang web yang dinamis dan dapat bekerja secara otomatis [10].

PHP pertama kali dibuat oleh Rasmus Lerdroft, seorang programmer C. Pada waktu itu PHP masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan script yang digunakan untuk mengolah data form dari web. Jadi semula PHP digunakannya untuk menghitung jumlah pengunjung di dalam webnya. Kemudian Rasmus Lerdroft mengeluarkan *Personal Home Page Tools* versi 1.0 secara gratis. Versi ini pertama kali keluar pada tahun 1995. Isinya adalah sekumpulan *script* PERL yang dibuatnya untuk membuat halaman webnya menjadi dinamis. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI, kependekan dari *Hypertext Preprocessing/Form Interpreter*.

PHP/FI merupakan akronim dari *Personal Home Page/Forms Interpreter*. Pada awal penyusunan, PHP/FI hanya mempunyai fungsi dasar dari PHP yang ada sekarang ini karena ketika pertama kali dibuat dengan menggunakan Perl maka PHP/FI juga mempunyai susunan dan karakter pemograman yang sama dengan Perl.

Pada tahun 1997, dikeluarkan PHP/FI versi 2.0. Fungsi-fungsi pada PHP/FI ditulis dengan menggunakan bahasa C. karena telah memiliki fungsi khusus untuk mengakses database, maka pada tahun yang sama terdapat kurang lebih 50.000 domain yang menggunakan PHP/FI sebagai bahasa pemograman untuk website, atau sekitar 1 % dari total domain yang ada pada waktu itu.

Dengan perilisan kode sumber ini menjadi *open source*, maka banyak programmer yang tertarik untuk ikut mengembangkan PHP. Kemudian pada tahun 1996 Rasmus Lerdroft mengeluarkan PHP versi 2.0 yang kemampuannya telah dapat mengakses database dan dapat terintegrasi dengan HTML. Pada rilis ini interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan. Pada tahun 1998 tepatnya pada tanggal 6 Juni 1998 keluarlah

PHP versi 3.0 yang dikeluarkan oleh Rasmus sendiri bersama kelompok pengembang *softwaranya*.

PHP versi 4.0 keluar pada tanggal 22 Mei 2000 merupakan versi yang lebih lengkap lagi dibandingkan dengan versi sebelumnya. Perubahan yang paling mendasar pada PHP 4.0 adalah terintegrasinya *Zend Engine* yang dibuat oleh Zend Suraski dan Andi Gutmans yang merupakan penyempurnaan dari PHP *scripting engine*. Yang lainnya adalah *build in HTTP session*, tidak lagi menggunakan *library* tambahan seperti pada PHP. Tujuan dari bahasa *scripting* ini adalah untuk membuat aplikasi-aplikasi yang dijalankan di atas teknologi web. Dalam hal ini, aplikasi pada umumnya akan memberikan hasil pada web *browser*, tetapi prosesnya secara keseluruhan dijalankan web *server*.

PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi. Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

PHP memiliki 8 (delapan) tipe data, yaitu [10]:

1. Integer

Tipe ini meliputi semua bilangan bulat dengan range -2,147,483,648 sampai +2,147,483,647 pada platform 32bit. PHP juga akan mengkonversi secara otomatis bila suatu bilangan berada diluar range tersebut ke dalam tipe data floating point. Tipe ini juga dapat dinyatakan dalam bentuk oktal (berbasis 8), desimal (berbasis 10), heksadesimal (berbasis 16).

2. Floating Point

Tipe ini biasa digunakan dalam bilangan pecahan namun bisa juga bilangan desimal. Tipe ini memiliki range $1.7E-308$ sampai $1.7E+308$. Dapat dinyatakan dalam bentuk bilangan desimal atau dalam bentuk pangkat.

3. String

Tipe data string dinyatakan dengan mengapitnya menggunakan tanda petik tunggal (' ') atau tanda petik ganda (" "). Perbedaan dari penggunaan keduanya adalah dengan tanda petik tunggal kita tidak dapat menggunakan variable dan escape sequence handling bersama dalam suatu kalimat.

4. Boolean

Tipe data boolean digunakan untuk menyimpan nilai true atau false. Biasanya tipe data ini mayoritas digunakan untuk melakukan pengecekan kondisi pada php.

5. Null

Tipe data yang tidak memuat apapun. Setiap Variabel yang diset menjadi Tipe Data NULL ini akan menjadikan Variabel tersebut kosong.

6. Array

Tipe ini dapat mengandung satu atau lebih data juga dapat diindeks berdasarkan numerik atau string. Ia juga mendukung multidimensi dan membolehkan semua datanya berbeda tipe data.

7. Object

Tipe data object dapat berupa bilangan, variable atau fungsi. Object dibuat dengan tujuan agar para programmer terbiasa dengan OOP, meski fasilitas ini masih minim.

8. Resource

Tipe Data Spesial yang satu ini di khususkan untuk menyimpan resource, sumber atau alamat. Variabel tersebut hanya dapat diciptakan oleh suatu fungsi khusus yang mengembalikan nilai berupa resource seperti penggunaan fungsi fopen, opendir, mysql_connect, mysql_query dan sebagainya.

Beberapa kelebihan PHP dari bahasa pemrograman *web*, antara lain:

1. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.

2. *Web Server* yang mendukung PHP dapat ditemukan dimana-mana, mulai dari apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis-milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.2.8 MySQL

MySQL adalah relational database management system (RDBMS) yang didistribusikan secara gratis dibawah licensi GPL (*General Public License*). MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian database terutama untuk pemilihan/seleksi dan pemasukan data yng memungkinkan pengoperasian data dikerjakan dengan mudah dan secara otomatis [11].

Keandalan suatu sistem database dapat diketahui dari cara kerja optimizernya dalam melakukan proses perintah-perintah SQL, yang dibuat oleh user maupun program-program aplikasinya. Sebagai database server, MySQL dapat dikatakan lebih unggul dibandingkan database server lainnya dalam query data. Hal ini terbukti untuk query yang dilakukan oleh single user, kecepatan query MySQL dapat sepuluh kali lebih cepat dari PostgreSQL dan lima kali lebih cepat dibandingkan Interbase.

2.2.8.1 DDL (Data Definition Language)

DDL adalah sebuah metode *query* SQL yang digunakan untuk mendefinisikan data pada sebuah database.

1. CREATE

Digunakan untuk melakukan pembuatan tabel maupun database.

2. DROP

Digunakan untuk melakukan penghapusan tabel maupun database.

3. ALTER

Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat baik menambah *field*, mengganti nama *field* ataupun menamakannya kembali serta menghapus *field*.

2.2.8.2 DML (Data Manipulation Language)

DML adalah sebuah metode *query* yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari *query* ini adalah untuk melakukan manipulasi database yang telah ada atau telah dibuat sebelumnya.

1. INSERT

Digunakan untuk melakukan penambahan data pada tabel database.

2. UPDATE

Digunakan untuk melakukan perubahan data yang ada pada tabel database.

3. DELETE

Digunakan untuk melakukan penghapusan data pada tabel. Penghapusan ini dapat dilakukan secara sekaligus maupun hanya beberapa data.