# BAB II
# LANDASAN TEORI

## 2.1 Ontologi

### 2.1.1 Pengertian Ontologi

Ontologi sangat penting untuk mendeskripsikan tentang sesuatu. Secara teknis, ontologi dapat direpresentasikan dalam bentuk objek, properti dari objek, dan relasi diantara setiap objek (Chandrasekaran, Josephson, and Benjamins,1999) [5].

Menurut Tijerino *et al* (2003) Ontologi adalah suatu konseptual yang formal dari sebuah domain tertentu yang dipakai bersama oleh kelompok orang [6]. Pendapat lain mengatakan bahwa ontologi terbentuk oleh 4 *tuple* (C,R,I,A). Adalah *concept,* R adalah *relation,* I adalah *instance*, dan A adalah *axiom. Axiom* digunakan untuk menyediakan informasi mengenai *class* dan *properties,* sebagai contoh batasan pada *properties* [7].

### 2.1.2 Manfaat Ontologi

*Eliza* (2006) Mengemukakan bneberapa manfaat dalam menggunakan ontologi, yaitu[8]:

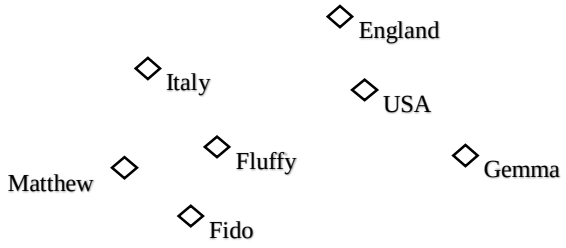
1. Menjelaskan suatu domain pengetahuan secara eksplisit. Memberikan struktur hierarki dari konsep untuk menjelaskan sebuah domain dan bagaimana mereka berhubungan.
2. Berbagi pemahaman dari informasi yang terstruktur. Sebagai contoh beberapa web yang berbeda memiliki informasi medis. Jika web tersebut dipakai bersama dan dipublikasikan dengan dasar ontologi yang sama maka perangkat lunak dapat mengekstrak dan mengumpulkan informasi dari situs yang berbeda.
3. Penggunaan ulang domain pengetahuan. Apabila ingin membangun ontologi yang luas dapat mengembangkan ontologi yang telah ada sebelumnya dan mengintegrasikan dengan beberapa ontologi lainnya yang relevan dengan ontologi yang ingin dibangun.

### 2.1.3 Komponen Ontologi

Komponen pada ontologi web semantik terdiri atas *instance*, *property*, *class* dan *axiom* (Matthew *et al*, 2004) [9]. Berikut ini akan dijelaskan secara singkat elemen dasar pembentuk ontologi web semantik.

a. *Instance*

*Instance* atau disebut juga *individual* adalah anggota (member) dari *classes*. *Instance* ini dapat dipandang sebagai objek yang ada pada domain yang dibahas.




**Gambar 2.1** Representasi *instance* (*individual*)

Dilihat dari gambar 2.1 di atas, terdapat *instance* England, Italy, USA, Fluffy, Gemma, Matthew, dan Fido, dimana *instance-instance* tersebut tidak didefinisikan relasi antara satu dengan lainnya.

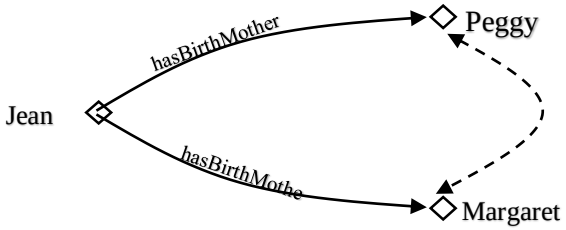
b. *Properties*

*Properties atau slot* merupakan *binary relation*. Ada dua jenis *properties*, yaitu *object properties* dan *data properties*. *Object properties* digunakan untuk menghubungkan *instance* dengan *instance* lainnya sedangkan *data properties* digunakan untuk menghubungkan *instance* dengan *datatype value* seperti *text*, *string* atau *number*. Pada *object properties,* terdapat jenis *properties*, yaitu :

1. *Functional Properties*

*Functional properties* adalah sebuah individu yang berhubungan hanya dengan satu individu. *Functional property* disebut juga sebagai *single valued property* atau *feature*. Sebagai contoh pada gambar 2.2 menunjukkan individu Jean hasBirthMother Peggy dan individu Jean hasBirthMother Margaret, hasBirthMother merupakan *functional*,

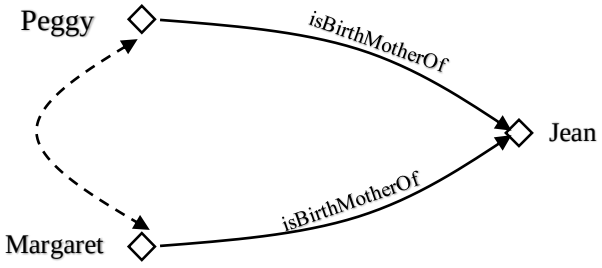
dapat disimpulkan bahwa Peggy dan Margaret adalah individu yang sama karena Jean hanya memiliki satu *BirthMother.*




**Gambar 2.2** Contoh *Functional*

2. *Inverse Functional Properties*

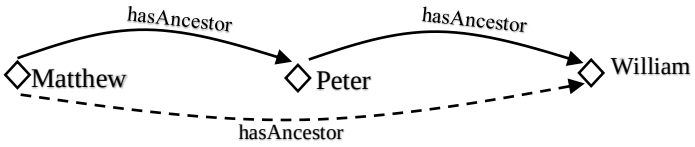
Jika sebuah properti adalah *inverse functional*, maka *inverse property* tersebut adalah *functional*, sebuah individu berhubungan hanya dengan satu individu. Pada gambar 2.3 menunjukkan dimiliki *inverse functional* isBirthMotherOf yang merupakan inverse dari hasBirthMother.




**Gambar 2.3** Contoh *Inverse Functional*

3. *Transitive*

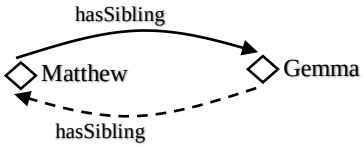
Jika sebuah properti adalah *transitive,* properti maka menghubungkan individu *x* dengan individu *y* serta menghubungkan individu *y* dengan individu *z*, maka dapat disimpulkan bahwa individu *x* berhubungan dengan individu *z* melalui properti *p*. Dapat dilihat pada gambar 2.4 menunjukkan jika individu Matthew mempunyi *ancestor* Peter, Peter mempunyai *ancestor* William, maka dapat disimpulkan Matthew mempunyai *ancestor* William.




**Gambar 2.4** Contoh *Transitive*

4. *Symmetric*

Properti *p* adalah *symmetric* jika properti tersebut menghubungkan *x* ke individu *y* kemudian menghubungkan *y* ke individu *x*. Seperti pada gambar 2.5, individu Matthew berhubungan dengan individu Gemma melalui properti hasSibling. Dapat disimpulkan Matthew bersaudara dengan Gemma




**Gambar 2.5** Contoh *Symmetric*

5. *Asymmetric*

Properti *p* adalah a*symmetric* jika properti tersebut menghubungkan *x* ke individu *y* namun tidak menghubungkan *y* ke individu *x*.

6. *Reflexive*

Bahwa sebuah properti bersifat refleksif menyebabkan setiap individu untuk berhubungan dengan dirinya sendiri.

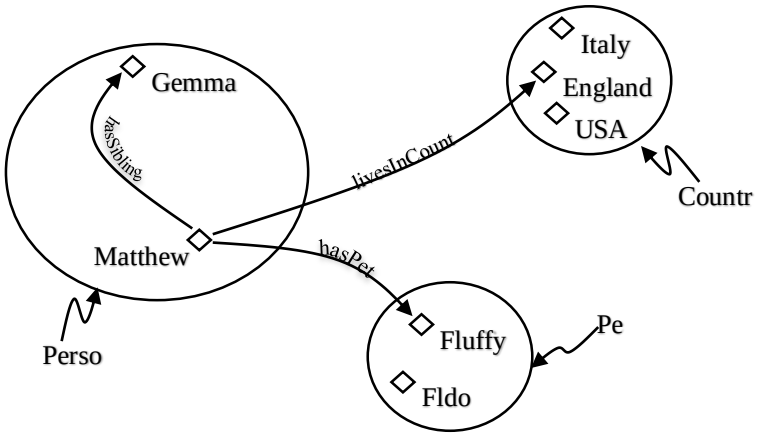
7. *Irreflexive*

Bahwa sebuah properti bersifat *irreflexive* menyebabkan setiap individu tidak dapat berhubungan dengan dirinya sendiri melalui.

c. *Class*

*Class* merupakan titik pusat ontologi. *Class* menjelaskan sebuah konsep dalam suatu domain yang terdiri dari beberapa *instance*. *Class* juga dikenal sebagai *concept, object* dan *categories*. Sebuah *class*

memiliki *subclass* yang ditujukan untuk menyatakan *concept* lebih spesifik dari *superclass.*




**Gambar 2.6** Representasi *class*

Dari gambar 3 di atas dapat dilihat bahwa *class* Penyakit memiliki *instance* DBD dan Malaria.

### 2.1.4 Pembangunan Ontologi

Terdapat beberapa tahapan yang dilakukan dalam pengembangan ontology yaitu (Natalya *et al.,* 2004) [10]:

a. Menentukan domain dalam batasan ontologi

Dalam mengembangkan ontologi dimulai dengan mendefinisakn domain dan batasan dengan menjawab pertanyaan berikut:

1. Domain apa yang akan melingkupi ontologi?
2. Mengapa ontologi digunakan?
3. Apa jenis pertanyaan terhadap informasi dalam ontologi sehingga perlu menyediakan jawaban?

Salah satu cara menentukan batasan dalam ontologi adalah dengan membuat daftar pertanyaan yang harus dapat dijawab oleh *knowledge base* atau yang biasa disebut *competency questions* (Natalya *et al.,* 2004).

b. Mempertimbangkan penggunaan ontologi yang sudah ada.

Ontologi yang sudah ada dapat diperhalus dan diperluas untuk domain dan *task* yang akan dibuat. Pengguaan otology yang sudah ada merupakan persyaratan apabila system yang akan dibuat akan berinteraksi dengan aplikasi lainnya yang telah dilakukan pada suatu ontology atau perbendaharaan kata yang dikontrol. Banyak ontologi yang telah tersedia dan dapat dimasukkan dalam pengembangan ontologi yang dilakukan.

c. Menentukan istilah yang akan dijelaskan *user.* Sebagai contoh pada ontology minuman anggur, istilah yang penting meliputi minuman anggur, anggur, tempat membuat anggur, warna minuman anggur, bentuk, cita rasa, dan kadar gula. Pada dasarnya, hal ini dibutuhkan untuk mendapat daftar istilah yang menyeluruh tanpa khawatir tumpang tindih antara *concept*, hubungan antara istilah dan properti dari *concept* atau *concept* tersebut termasuk *class* atau slot.

d. Mendefinisikan *class* ontologi dan menyusun *class* dalam hirarki taksonomi *(subclass – superclass*).

Terdapat beberapa pendekatan dalam pengembangan hirarki *class* (Matthew et al., 2004) yaitu [9]:

1. Proses pengembangan *top-down* dimulai dengan mendefinisikan *concept* umum dalam domain dilanjutkan dengan *concept* yang lebih spesifik.

2. Proses pengembangan *bottom-up* dimulai dengan mendefinisikan *class* yang paling spesifik kemudian dikelompokkan menjadi *class* dengan konsep yang lebih umum.

3. Proses pengembangan *combination* adalah sebuah kombinasi antara pendekatan *top-down* dan *bottom-up*. Mendefinisikan konsep yang menonjol terlebih dahulu kemudia menggeneralisasi dan mengkhususkan konsep tersebut.

e. Mendefinisakn *slot* atau *properties* dan menjabarkan nilai dari *slot* tersebut.

f. Mendefinisikan *facets* pada *slots*.

*Slot* dapat memiliki *facet* yang berbeda dalam menggambarkan tipe nilai, nilai tersebut dapat berupa *cardinality* (sejumlah nilai) dan fitur lainnya.

Beeikut merupakan beberapa *facet* yang umum dgunakan:

1. *Slot cardinality*

   *Slot cardinality* mendefinisikan sejumlah nilai yang dimiliki oleh *slot.* Beberapa system hanya membedaka *single cardinality* (mempunyai satu nilai) dengan *multiple cardinality* (mempunyai beberapa nilai).

2. *Slot value type*

   Sebuah tipe *facet* menggambarkan tipe dari nilaiyang dapat mengisi *slot.* Berikut merupakan tipe nilai yang umum digunakan:

   a. *String*, tipe nilai yang paling sederhana yang biasa digunakan untuk *slot.*

   b. *Number*, terkadang tipe nilai yang lebih spesifik seperti *Float* dan *integer* yang digunakan dalam penggambaran *slot* dengan nilai numeric.

   c. *Boolean*

      Pada *Boolean slot* mempunyai *yes-no flag*

   d. *Enumerated*

      *Enumerated slot* merinci daftar dari spesifik nilai yang diperbolehkan untuk *slot.*

   e. *Instance*

      *Instance* membuat hubungan antara individual.

g. Membuat *instances*

   Langkah terakhir adalah membuat individual atau *instances* pada *class* dalam suatu hirarki. Dalam mendefinisakn individual *instance* pada *class* harus diperhatikan persyaratan berikut:

   1. Pemilihan *class*
   2. Pembuatan individual *instances* pada c*lass tersebut.*

*Knowledge base* selanjutnya dapat dibuat dengan mendefinisikan individual *instances* dari *class* yang terisi pada nilai spesifik *slot* dan *slot* batasan tambahan (*facets*).

3. Mengisi nilai *slot.*

## 2.2 Semantik

### 2.2.1 Pengertian Semantik

Semantik (Bahasa Yunani : semantikos, memberikan tanda, penting, dari kata *sema*, tanda) adalah cabang linguistic yang mempelajari makna yang terkandung pada suatu bahasa, kode atau jenis representasi lain (Semantic Web) [11]. Semantik biasanya dikontraskan dengan ekspresi makna sintaksis yaitu pembentukan symbol kompleks dari simboll yang lebih sederhana.

Web semantik merupakan pengembangan dari *world wide web* dimana *content* web yang ditampilkan tidak hanya dalam bahasa format manusia yang umum tetapi juga dalam format yang dapat dibaca dan digunakan oleh mesin. Web semantik memiliki informasi yang dimiliki oleh mesin yang memiliki kecerdasan buatan sehingga mampu menemukan dan mengintegrasikan informasi dengan mudah. Tujuan dari web semantik adalah mengatur informasi dan prosedur. Fundamental dalam pembangunan semantik web adalah kreasi dan semantik metadata(Semantic Web).

Metadata terdiri dari dua bagian, yaitu :

a. Penggambaran sebuah dokumen.

Contohnya adalah halaman web ataubagian dari suatu dokumen seperti sebuah paragraf

b. Penggambaran entitas didalam suatu dokumen.

Contohnya adalah sesorang atau sebuah perusahaan. Pada semua kasus, yang terpenting bahwa metadata adlah semantik, yang menggambarkan semua isi dari dokumen tersebut.

### 2.2.2 Komponen dalam Semantik WEB

Pembuatan *semantic* web dimungkinkan dengan adanya seumpulan standar yang dikoordinasu oleh *world wide web consortium* (W3C). Standar yang paling penting dalam pembangunan *semantic* web adlah XML, XML *Schema*, RDF, RDF *Schema* dan OWL. Komponen – komponen *semantic* web ini yangmemungkinkan dan interaksi pada level mesin (Niko, 2007) [12].

a. XML dan XML *Schema*

*Extensible markup language* (XML) merupakan *markup* yang didesain untuk menjadi sarana yang mudah dalam mengirimkan dokumen melalui web. Berbeda dengan *hypertext Markup Language* (HTML), XML memungkinkan penggunaannya untuk mendefinisikan *custom tag.*

XML *Schema* merupakan bahasa yang digunakan untuk mendefinisikan sekumpulan aturan (*schema*) yang harus dipatuhi oleh dokumen XML. Struktur dari dokumen XML yang dibuat harus sesuai dengan *schema* yang telah didefinisikan tersebut.

b. RDF dan RDF *Schema*

*Resource Description Framework* (RDF) adalah spesifikasi yang dibuat oleh W3C sebagai metode umum untuk memodelkan informasi dengan menggunakan sekumpulan format sintaks. Ide dasar dari RDF adalah bagaimana kita dapat membuat pertayaan mengenai sebuah *resource web* dalam bentuk ekspresi subjek (S), predikat (P), objek (O). Dalam terminology RDF, SPO ini seringkali disebut denganistilah N-triple. Subjek mengacu pada *resource* yang ingin dideskripsikan. Predikat merupakan komposisi yang menerangkan sudut padang dari subjek yang dijelaskan objek, sementara subjek dan objek merupakan entitas. Objek di dalam RDF dapat menjdai subjek yang diterangkan oleh objek lainnya.dengan inilah objek dapat berupa masukan yang dapat diterangkan secara jelas dan detail, sesuai dengan keingina pengguna yang memberikan masukan. Dengan menggunakan RDF, *website* dapat

menyimpan dan melakukan pertukaran informasi antar web. RDF telah digunakan pada aplikasi-aplikasi berikut :

1. RDF *Site Summary* (RSS)

RSS memberikan informasi update sebuah *website* tanpa pengunjung perlu mengunjungi *website* tersebut.

2. *Friend of a Friend* (FOAO)

Didesain untuk mendeskripsikan orang-orang, ketertarikan dan hubungan mereka.

3. *Semantically-Interlinked Online Communities* (SIOC)

menerangkan komunitas *online* dan menciptakan koneksi antara diskusi berbasis internet seperti *message board, blog* maupun *mailing list.* RDF *schema* dapat dipandang sebagai kamus data atau *vocabulary* untuk mendeskripsikan *properties* dan *classes* dari *resources* RDF.

c. *Ontology Web Language* (OWL)

OWL adalah suatu bahasa yang dapat digunakan oleh aplikasi-aplikasi yang bukan sekedar menampillkan informasi tersebut pada manusia melainkan juga yang perlu memproses isi informasi. Ontologi sendiri dapat didefinisikan sebagai suatu cara untuk mendeskripsikan arti dan relasidari istilah-istilah tersebut dengan cara yang lebih mudah atau dengan pengertian lain adalah representasi istilah beserta hubungannya. Ketika informasi yang ada dalam dokumen perlu untuk diproses oleh aplikasi atau mesin, OWL dapat digunakan untuk merepresentasikan makna suatu istilah secara eksplisit sekaligus hubungan antara istilah-istilahtersebut.

Dengan menggunkan OWL, kita dapat menambah *vocabulary* tambahan disamping *semantic* formal yang telah dibuat sebelumnya menggunakan XML, RDF dan RDF Schema. Hal ini sangat membantu penginterpretasian mesin yang lebih baik terhadap isi web. Untuk mendeskripsikan *properties* dan *classes,* OWL menambahkan *vocabulary* seperti:

1. "*among others"*
2. Relasi antar *classes* (misalnya : "*disjointnes*s")
3. *Kardinalitas*
4. Kesamaan (*equality*)
5. Karakteristik *property*
6. *Enumerated classes*

OWL memiliki tiga *sub-language* yaitu :

1. OWL *Lite*

   Mendukung pengguna yang memerlukan klasifikasi dan dalam batasan yang sederhana.

2. OWL DL

   Mendukung konstruksi seluruh OWL, tetapi hanya dapat digunakan pada batasn tertentu.

3. OWL *Full*

   Diperuntukkan bagi yang menginginkan maksimum pengguna dan kebebasan sintaksis

d. SPARQL (SPARQL Protocol and RDF Query Language)

SPARQL (SPARQL Protocol and RDF Query Language) adalah standar yang dikeluarkan oleh W3C guna melakukan query untuk memperoleh data dari sumber daya web (web resource) yang terdapat pada dokumen RDF dan OWL. SPARQL query terdiri atas triple pattern yang sama seperti RDF triple yaitu subject, predicate, dan object dimana masing-masing dari subject, predicate, dan object dapat menjadi variabel pada SPARQL. Query SPARQL didasarkan pada pencocokan pola triple pada RDF.

Klausa yang digunakan dalam query SPARQL, diantaranya :

1. PREFIX

Statemen PREFIX merupakan sebuah metode yang digunakan sebagai penunjuk yang membawa informasi dalam suatu *resource* yang dalam hal ini diwakili oleh URI (*Uniform Resource Identifier*). Pada dasarnya PREFIX digunakan untuk menyingkat sebuah *resource*.

2. SELECT

*Statement* SELECT mendefinisikan sebuah daftar variabel-variabel yang akan dikembalikan sebagai hasil dari eksekusi *query*. Setiap variabel diawali dengan notasi "?".

3. WHERE

*Statement* WHERE mendefinisikan sederetan *triple pattern* yang harus dimiliki oleh setiap hasil *query* yang valid. Seluruh pola yang merepresentasikan suatu kalimat RDF harus sesuai dengan RDF *triples*, yaitu terdiri dari *subject*, *predicate*, dan *object*. RDF *triple* tersebut dapat direpresentasikan oleh URI atau sebuah variable dan nilai literal.

### 2.2.3 Keuntungan Semantic

Keuntungan yang dimiliki oleh *semantic* web adalah sebagai berikut:

a. Waktu yang diperlukan untuk mendapatkan informasi yang dibutuhkan lebih spesifik dan efisien.

b. Pekerjaan pencarian yang dilakukan manusia dapat digatikan oleh mesin.

## 2.3 Tool Ontologi

### 2.3.1 Protege

Protege merupakan *tool ontology* dengan *platform open source* untuk membangun domain model dan aplikasi *knowledge* dan dapat memvisualisasikan hasil ontologi dalam berbagai format. Sebuah ontologi menggambarkan *concept* dan hubungan-hubugan yang penting dalamdomain yang khusu, yang menyediakan kosa kata dalamdomain tersebut. Dalam beberapa tahun terkhir, ontologi telah dadopsi dalam

bisnis dan komnitas ilmiah seperti *electronic commerce* dan *semantic web service.*

Pada protege terdapat dua cara dalam pemodelan ontologi, yaitu:

a. Protege Frame Editor

User dapat membangun ontologi dalam *frame-based* dengan kesepakatan dengan *Open Knowledge Base Connectivity Protocol* (OKBC) pada model ini, sebuah ontologi terdiri dari seperangkat class yang terorganisir pada suatu hirarki yang merepresentasikan sbuah domain, seperangkat *slot* yang berhubungan dengan *class* serta *instance* dari tiap *class* tersebut.

b. Protege OWL

Protege-OWL editor merupakan kelanjutan dari protege yang mendukung *Ontology Web language* (OWL). OWL merupakan pengembangan mutakhir standar bahasa ontologi yang disahkan oleh *World Wide Web Consortium* (W3C) untuk mempopulerkan *semantic web vision.* Protege_OWL editor memungkinkan user untuk :

1. Mengambil dan menyimpan OWL dan RDF ontologi.
2. Mengubah dan memvisualisasikan *class, properties* dan *Semantic Web Rule Language* (SWRL).
3. Menjabarkan karakteristik *class* secara logis sebagai ekspresi OWL.
4. Mengeksekusi penalaran seperti *description logic classfier.*
5. Mengubah OWL individual untuk web semantik.

Protege-OWL berhubungan secara erat dengan Jena dan mempunyai *open source* Java API untuk pengambangan *Semantic Web Service.* Jena merupakan *framework* berbasis Java untuk mengkonstruksi aplikasi semantik web. *Framework* ini menyediakan lingkungan pemograman RDF, RDF Schema OWL dan SPARQL.

Protege-OWL *Application Programming Interface* (API) adalah sebuah *java Library open source* untuk *Web Otology Language* (OWL)

dan RDF. API menyediakan metode untuk mengambil dan menyimpan file OWL, menanyakan OWL data model, menjalakan penalaran berdasarkan *Description Logic* engines *dan sebagai* graphical user interface. API didesain untuk dapat digunakan dalam dua konteks, yaitu :

1. Pengambangan komponen yang dieksekusi dalam Protege-OWL editor *user interface.*
2. Pengembangan aplikasi *stand-alone* (contoh: aplikasi Swing, Servlet dan *plug-in Ecipse*).

Pada *object property* terdapat tampilan karakteristik Objek Properti menampilkan karakteristik yang dinyatakan untuk properti objek yang dipilih. Karakteristik ditampilkan sebagai daftar kotak centang. Jika kotak dicentang untuk karakteristik tertentu yang berarti bahwa beberapa ontologi dalam penutupan impor ontologi aktif menegaskan karakteristik itu. Jika kotak tidak dicentang untuk karakteristik tertentu yang berarti bahwa karakteristik tersebut tidak ditegaskan sama sekali dalam setiap ontologi dalam penutupan impor ontologi aktif.

### 2.3.2 Jena Fuseki

Jena adalah sebuah *framework* Java untuk membangun aplikasi web semantik. Jena menyediakan perpustakaan Java yang luas untuk membantu *developer* mengembangkan kode yang menangani RDF, RDFS, RDFa, OWL dan SPARQL sesuai dengan rekomendasi dari W3C yang diterbitkan. Jena juga menyertakan *rule-based inference engine* untuk melakukan penalaran berdasarkan ontologi OWL dan RDFS, serta berbagai strategi penyimpanan dalam penyimpanan tiga kali lipat RDF pada memori atau *disk*.

Jena awalnya dikembangkan oleh para peneliti di HP Labs, dimulai di Bristol, Inggris, pada tahun 2000. Jena selalu menjadi proyek *open-source*, dan telah banyak digunakan dalam berbagai aplikasi web semantik dan aplikasi demo. Pada tahun 2009, HP memutuskan untuk memfokuskan kembali aktivitas pengembangan dari dukungan

langsung pengembang Jena, meskipun tetap mendukung tujuan proyek. Akhirnya, tim proyek berhasil mengajikan permohonan agar Jena diakusisi oleh Apache Software Foundation pada November 2010 [13].

## 2.4 Penyakit Tropis

Penyakit tropis yang terjadi di Indonesia menjadi tanggungan yang harus dituntaskan, dari penyakit menular, tidak menular, dan penyakit baru. Penyakit tropis yang diselesaikan dan menjadi batasan masalah adalah penyakit tropis yang mudah dikenali dari gejalanya. Penyakit tropis berikut sesuai dengan buku referensi Widoyono (2011) yang menjelaskan penyakit tropis epidemiologi, penularan, pencegahan, dan pemberantasannya [14].

A. Tuberkulosis (TBC)

Penyakit tuberkulosis disebabkan oleh kuman *Mycobacterium tuberculosis* ditularkan melalui udara saat pasien batuk.

Gejala dan tandanya terdapat batuk berdahak, batuk berdarah, sesak napas, nyeri dada, demam, berkeringat di malam hari, meriang, penurunan berat badan.

B. Difteria

Difteria adalah penyakit akut yang disebabkan oleh toksin dari bakteri *Corynebacterium diphteriae.* Penyakit ini menyerang saluran pernafasan. Penularannya terjadi melalui droplet saat penderita (atau *carrier*) batuk, bersi, dan berbicara.

Gejala dan tandanya terdapat peradangan pada tenggorok, demam yang tidak tinggi, dan pembengkakan leher, serta terjadi pembentukan membran keputihan pada tenggorok atau tonsil yang mudah berdarah apabila dilepas.

C. Tyfus atau demam tifoid

Demam tifoid adalah infeksi akut pada saluran pencernaan yang disebabkan *Salmonella typhi*.

Gejala dan tandanya terdapat demam berkepanjangan, gangguan sistem pencernaan, gangguan kesadaran, batuk, sembelit, diare, mual muntah, nyeri perut, kembung, higga nafsu makan kurang.

D. Leptospirosis

Leptospirosis adalah infeksi akut yang disebabkan oleh bakteri *Leptospira.* Penyakit ini menyerang peredarah darah. Penularan pada manusia dapat melalui jika kntak dengan air, tanah, dan lumpur yang tercemar bakteri. Kontak degan organ, darah, dan urin hewan terinfeksi. Mengkonsumsi makanan yang terinfeksi.

Gejala dan tandanya demam tinggi mendadak, malaise, nyeri otot, ikterus, sakit kepala, dan nyeri perut akibat gangguan hati, ginjal, dan meningitis

E. Demam berdarah dengue (DBD)

Penyakit ini disebabkan virus dengue dari kelompok *Arbovirus B.* Gejala dan tandanya demam tinggi mendadak, sakit kepala, mual muntah, sulit bernafas, nyeri otot, terdapat bintik merah.

F. Campak

Campak adalah suatu penyakit akut yang sangat menular yang disebabkan oleh virus. Campak disebut juga *rubeola*, *morbilli*, atau *measles.* Penularannya melalui droplet diudara sehingga gampang menular.

Gejala dan tandanya demam, batuk, pilek, dan konjungtivitis. Diikuti dengan bercak kemerahan dibagian kulit.

G. Hepatitis

Penyakit ini disebut juga hepatitis infeksiosa. Hepatitis A berupa infeksi hati akut.

Gejala dan tandanya demam, kekuningan pada kulit dan mata, mual muntah, sering BAK, kesadaran turun, nyeri perut, feses pucat.

H. Varisela

Varisela adalah infeksi virus akut yang ditandai dengan adanya vesikel pada kulit yang sangat menular. Penyakit ini disebut juga

dengan *chicken pox*, cacar air, atau varisela zoster. Penularannya melalui kontak langsung dengan penderita.

Gejala dan tandanya timbul demam, malaise, anoreksia, dan nyeri kepala. Kemudian akan muncul ruam, dan vesikel seperti tetesan air.

I. Polio paralisis

Penyakit ini menyerang sistem saraf perifer yang disebabkan oleh virus polio. Gejala utama penyakit ini yaitu kelumpuhan yang bisa saja menyebabkan kecacatan permanen.

J. Polio nonparalisis

Polio nonparalisis tidak menyebabkan kelumpuhan. Dengan gejala demam, leher kaku, mual muntah, batuk berdahak, nyeri kepala, nyeri punggung, nyeri sendi hingga otot.

K. Malaria

Malaria disebabkan oleh parasit *sporozoa plasmodium* yang ditularkan melalui gigitan nyamuk *anopheles* betina.

Gejalanya demam lebih dari dua hari, menggigil, berkeringat, gangguan kesadaran, nyeri otot, mual muntah.

L. ISPA (Infeksi saluran pernapasan akut)

ISPA adalah penyakit saluran pernapasan yang bersifat akut dengan berbagai macam gejala. Seperti batuk, ruam pada kulit, demam, lemas, sakit tenggorokan, menggigil, kesadaran turun, kejang.

M. Pneumonia

Pneumonia adalah penyakit infeksi yang menyerang paru, sehingga menyebabkan kantung udara di dalam paru meradang dan membengkak.

Gejala umumnya sering batuk, demam, berkeringat, menggigil, nyeri dada, susah bernapas, nafsu makan turun, dan detak jantung terasa cepat. Gejala yang jarang meliputi sakit kepala, lemas, mual dan muntah, nyeri sendi dan otot, batuk disertai darah.

N. Usus buntu

Usus buntu adalah penyakit yang tidak menular. Gejalanya demam disertai perut bengkak, nyeri perut, sembelit, kembung, hingga diare.

O. Muntaber

Muntaber adalah penyakit yang menyerang sistem pencernaan. Gejalanya demam, nafsu makan berkurang, mual muntah, nyeri perut, diare hingga dehidrasi.

P. Amandel

Amandel memiliki gejala demam, batuk kering, sakit tenggrokan, leher kaku, nafas bau hingga bercak ke abu-abuan pada mulut.

Q. Meningitis

Meningitis penyakit yang meyerang otak dengan gejala demam, nyeri kepala, menggigil, kesadaran turun, leher kaku, silau jika terkena cahaya, napas cepat, mual muntah.

R. Rubella

Rubella memiliki gejala demam, nyeri kepala, mata merah higga berkunang-kunang, hidung tersumbat, nyeri otot dan sendi.

S. Sinusitis

Sinusitis penyakit yang tidak menular memiliki gejala nyeri bagian wajah, nyeri kepala dan telinga, sakit tenggorokan, indra penciuman buruk, nafas bau, sakit gigi, ingus hijau, dan mual muntah.

T. Zika

Zika penyakit yang dapat ditularkan melalui nyamuk. Degan gejala demam, nyeri sendi dan otot, lemas dan nyeri kepala hingga kesadaran turun.