

# Introducing Linked Data And The Semantic Web

<http://www.linkeddatatools.com/semantic-web-basics>

Linked Data와 Semantic Web이란 무엇인가? 원칙적으로 Semantic Web은 Web 3.0 이다; 즉, 시스템들 또는 엔티티(entities) 끼리 서로 데이터를 링크하는 방법이다. 따라서 웹 환경에서는 이것을 사용하여 데이터끼리 풍부하고, 자아-기술적(self-describing)인 관계성(interrelations)을 갖는다.

사실, SW는 HTML 문서에 포함되어 있는 데이터를 사람이 읽는 것이 아니라, 컴퓨터가 읽을 수 있어야 한다는 사고의 전환에서 비롯되었다. 다시 말해서, 컴퓨터가 인간을 위해 좀 더 많은 사고적(thinking) 업무를 수행할 수 있어야 한다는 생각에서 출발하였다.

## How Does It Differ From The Web As It Is Today?

오늘날 우리가 웹에서 얻는 많은 데이터는 웹 페이지의 형태로 우리에게 전달되는데, 이것들은 HTML 문서들이며, 서로 하이퍼링크로 연결되어 있다. 사람이나 기계 모두 이 같은 문서를 읽을 수 있다. 그렇지만, 사람은 전형적으로 페이지에서 키워드를 찾을 수 있지만, 기계가 이 문서에 들어 있는 의미를 스스로 발췌하기란 결코 쉽지 않다.

## Enter Linked Data - Liberating Web Databases From Their Old Chains

웹에는 많은 정보가 들어 있지만, 전형적으로 raw data 그 자체를 이용할 수는 없다. 만일 데이터베이스의 웹 사이트가 만들려 한다면, 그것의 HTML 문서들은 해당 데이터로만 만들어져야 한다.

따라서 SW는 이 같은 문제를 해결하기 위하여, 다음과 같은 여러 가지 방법으로 현재의 인터넷 모습을 변화시키고 있다:

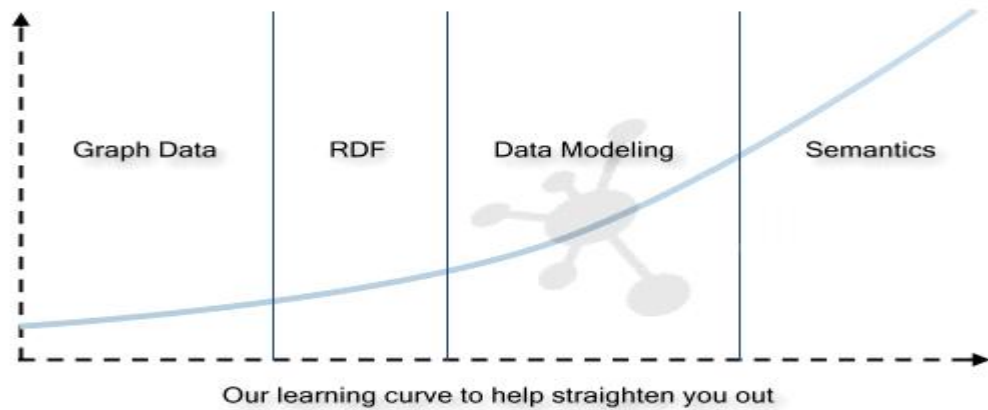
- 인공지능처리를 위하여 web of data를 개방하고 있다.(웹으로 하여금 우리를 위해 약간 좀 생각하도록 한다.)
- 회사, 조직, 개인들로 하여금 공개된 표준 포맷으로 자유롭게 자신들의 데이터를 출판하도록 한다.
- 기업에서 이미 웹에 있는 데이터를 이용할 수 있도록 한다(데이터 주고받기)

실제로, SW는 여러 곳에서 출판된 모든 HTML 정보를 수집한 다음, 그것을 마치 전에는 하나의 데이터베이스였던 것처럼, 다루고 조사하는 데이터 모델의 한 유형이다.

오늘날의 다양한 관련 도구와 소프트웨어를 비교해 볼 때, 이것이 인터넷을 사용하는 모든

data humanity 분야의 자동화 연구에 끼치는 이익은 엄청나다.

But Where Do I Start?



이 강의는 복잡하지 않다. 우리는 초보자의 시작을 돕기 위하여 개론으로 시작한 다음에, SW를 정의하고, SW와 현재의 웹과의 차이를 알아보며, 마지막으로 SW의 제작 기술에 대하여 깊이 있게 알아볼 것이다:

- ▶ Tutorial 1 Introduction To [Graph Databases](#) - gives a brief overview of the way in which the semantic web stores data.
- ▶ Tutorial 2 [RDF](#) - A Quick Start - an introductory look at Resource Description Framework (RDF), the format the semantic web uses to store data in graph databases.

## I. Introducing Graph Data

SW은 그렇게 잘 알려진 분야가 아니다. 초보자가 SW의 정의와 그것의 작동원리를 열심히 이해하려 한다면, 먼저 그것의 데이터 저장방법부터 이해해야 한다. 따라서 먼저, SW의 데이터 저장 모델인 graph database부터 알아보다.

여러분이 전통적인 IT 분야에 지식을 갖고 있고, 계층형(for example XML) 또는 관계형 데이터베이스(for example MySQL, MS SQL)의 데이터저장방법에 대하여 알고 있겠지만, Resource Description Framework(RDF)에 대해선 잘 알지 못할 수도 있다.

RDF란 SW 커뮤니티에서 사용하는 일반적인 두문자어이며, 시멘틱 데이터의 웹을 제작하는데 필요한 기본적인 빌딩블록들 중의 한 요소이다. 또한 이것이 define(정의)하는 것은 여러분이 익숙하지 않은 데이터베이스의 유형인 **graph database** 이다.

2) define이란?

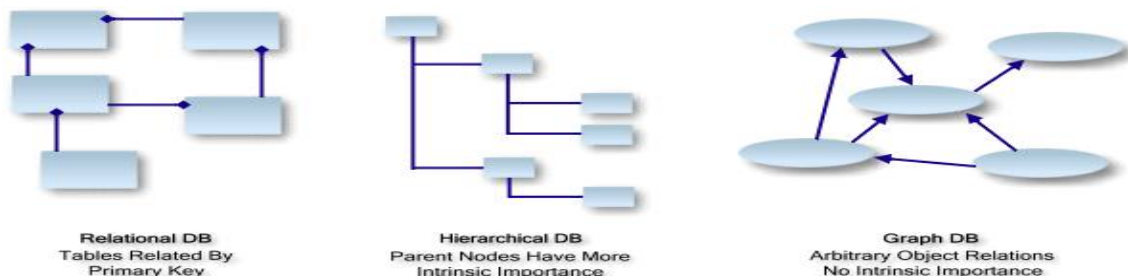
- To state the precise meaning of (a word or sense of a word, for example).
- To describe the nature or basic qualities of; explain:

비록 이 데이터베이스에 대해 여러분이 생소하더라도, 이것이 바로 전 세계적으로 SW를 제작하는데 사용하는 데이터베이스의 한 부류 이다.

이제 graph database가 무엇이고, 어떻게 RDF가 그것을 정의하는지, 그리고 graph database를 시각화하는 방법이 무엇인지에 대하여 알아보기로 한다.

먼저 hierarchical, relational, 그리고 graph databases를 비교하여 이것들이 서로 어떻게 다른지를 살펴보기로 하자:

## 1.1 Introducing The Graph Database



대부분의 데이터 저장 유형들에서는, 자신들의 여러 가지 데이터 요소(elements) 중에는 다른 것보다 보다 더 많은 precedence나 importance를 갖고 있는 몇 가지의 요소들(예를 들어, data nodes 또는 data tables)에 대한 개념을 정의하고 있다.

예를 들어, XML 다큐먼트를 보자. 전형적으로 XML 다큐먼트에는 한 개의 parent node와 함께 여러 개의 정보 노드를 사용하고 있다. 따라서 이러한 다큐먼트의 root는 최상위의 노드이며, 이 노드는 어떠한 부모 노드도 갖지 않는다.

위의 그림을 살펴보자. 맨 오른쪽의 data graph에는 어떠한 roots(또는 계층)도 존재하지 않으며, 서로 연결된 자원들로만 구성되어 있다. 또한 이 그래프의 어떠한 자원도 특별하게 다른 자원보다 본질적으로(intrinsic) 더 중요하지 않다는 것을 나타내고 있다.

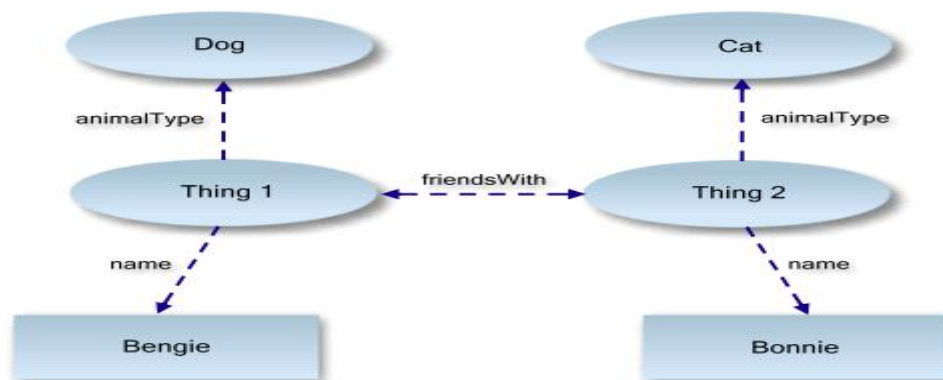
## An Example Of A Data Graph

사물들(things)이 서로 어떤 관계를 갖고 있는지를 기술하고 있는 지시문(statements)과 그 속에 있는 관계들을 RDF로 표현하는 방법을 알아보기에 앞서서, 먼저 그래프로 이들 사물간의 관계를 시각화해 보자. 이것은 매우 쉽다.

먼저 아래의 개(벤지)와 고양이(보니) 간의 관계를 설명하는 지시문을 살펴보자:

- Bengie는 개다.
- Bonnie는 양이다.
- Bengie와 Bonnie는 친구이다.

위의 간단한 3개의 지시문을 데이터 그래프로 그려보면, 아래와 같다:



위의 그래프에 나타나 있는 관계는 매우 직관적이므로, 우리는 이 관계들을 완전하게 이해할 수 있다: 즉, 우리는 "Thing 1" 과 "Thing 2" 라는 두 개의 사물이 있고, 이 사물들의 속성 이름이 하나는 "animalType"이고 나머지는 "friendsWith"라는 것을 알 수 있다.

또한, 우리는 "Thing 1"의 속성 "name"의 값이 "Bengie"이고, "Thing 2"의 속성 "name"의 값이 "Bonnie"라는 것도 알 수 있고, 물론 "Thing 1"은 개를, "Thing 2"는 고양이를 의미한다는 것도 알 수 있다. 끝으로, 이 두 개의 사물은 서로 친구(속성 "friendsWith"가 화살표로 양 방향 모두를 가리키고 있음)라는 것도 알 수 있다.

**핵심 포인트** - 위의 그래프에서 화살표는 속성(properties)을 나타낸다: 때론, 이것을 RDF 용어(terminology)로는 predicates라 한다. 이 두 용어는 호환적으로 사용되며, 그래프에서 표시된 화살표는 속성을 의미한다.

<Graph Model의 예: 지시문>

[http://dbpedia.org/resource/Billie\\_Jean](http://dbpedia.org/resource/Billie_Jean) has a **singer** whose value is **Michael Jackson**

- ▶ Subject: `http://dbpedia.org/resource/Billie_Jean` (URI)
- ▶ Predicate: `http://www.example.com/terms/singer` (URI)
- ▶ Object: `Michael_Jackson` (Literal)

공식적으로 RDF에 대해 알아보기 전에, 다음과 같은 간단한 예를 먼저 맛보기로 하자:

### 1.3 A Starting Example Of RDF

```

01. <?xml version="1.0" encoding="UTF-8"?>
02.
03. <rdf:RDF
04.     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
05.     xmlns:dc="http://purl.org/dc/elements/1.1/"
06.     xmlns:region="http://www.country-regions.fake/">
07.
08.     <rdf:Description rdf:about="http://en.wikipedia.org/wiki/Oxford">
09.         <dc:title>Oxford</dc:title>
10.         <dc:coverage>Oxfordshire</dc:coverage>
11.         <dc:publisher>Wikipedia</dc:publisher>
12.         <region:population>10000</region:population>
13.         <region:principaltown rdf:resource="http://www.country-regions.fake/oxford"/>
14.     </rdf:Description>
15.
16. </rdf:RDF>

```

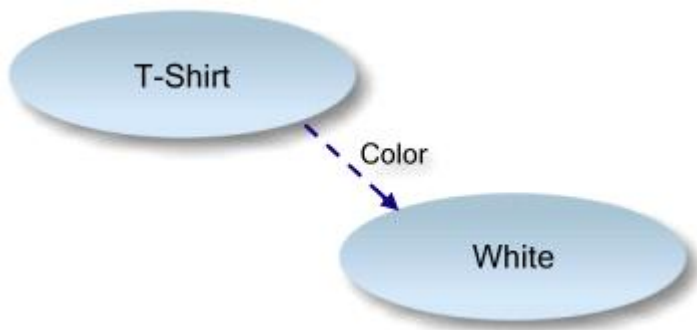
지금 위의 내용에 대해 걱정하지 마라. 우리는 나중에 이것을 다시 검토할 것이다. 지금은 단지 위와 같은 것이 RDF/XML - the XML form of RDF라는 것만을 이해하면 된다. RDF를 레코딩하는 여러 가지 방법들이 있지만, 우리는 단지 RDF/XML만을 살펴볼 것이다.

### 1.3 The RDF Statement (Triple)

위의 콘텐츠에서 붉은 색으로 표시한 RDF/XML(<rdf:Description> tags 사이)를 RDF 지시문, 또는 때때로 RDF triple이라 부른다. 이 두 가지 이름 중에서 triple이 우리가 RDF를 이해하는데 가장 도움이 되는 용어인데, 그 이유는 이것이 지시문을 3가지의 요소로 지시문이 구성되어 있다는 것을 의미하기 때문이다: 즉, 지시문의 subject, predicate, 그리고 object.

그래프의 형태로 이 3가지의 구성 용어들을 나타내 보자. 예를 들어, 티-셔츠의 색깔을 표현하고 있는 데이터에 대한 다음의 그래프를 살펴보자:

위의 간단한 그래프에서는 3개의 구성요소가 표현되어 있다:



- Subject: T-shirt;
- Predicate(또는 property): color;
- Object: white.

핵심 포인트 - RDF는 SW의 데이터 구조를 정의하는 토대이지만, 데이터에 숨어있는 어의나 의미를 스스로 기술하지는 못한다. 이것은 나중에 RDFS (RDF Schema) 와 OWL (Web Ontology Language)을 배울 때 다루기로 한다. 지금은 이것들에 대하여 걱정하지 마라. 먼저, RDF가 데이터간의 관계구축방법이 이미 잘 알려져 있는 기존의 데이터저장방법과 어떻게 다른지에 대해 배워야 한다. 또한 여러분은 계층형이나 관계형 데이터 모델에서 벗어나 graph model로 추세가 바뀌고 있음을 깨닫는 것이 중요하다.

간단한 RDF/XML 지시문을 통해, 이 구성요소들에 대하여 알아보자:

```

01. <?xml version="1.0" encoding="UTF-8"?>
02.
03. <rdf:RDF
04.     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
05.     xmlns:feature="http://www.linkeddatatools.com/clothing-features#">
06.
07.     <rdf:Description rdf:about="http://www.linkeddatatools.com/clothes#t-shirt">
08.
09.         <feature:color rdf:resource="http://www.linkeddatatools.com/colors#white"/>
10.
11.     </rdf:Description>
12.
13. </rdf:RDF>

```

공식적으로 우리는 이 지시문을 다음 강의에서 분석하겠지만, 여러분은 먼저 위의 RDF에서 subject, predicate, object를 어떻게 기술하는지에 대하여 감을 잡아야 한다.

이제 이장을 마감한다. 이제 여러분은 다음과 같은 것을 이해하고 있을 것이다:

- What a data graph is.
- That the semantic web is a giant, global data graph defined in RDF (Resource

Description Framework).

- The all-important shift in thinking from storing data in relational, or hierarchical models to a storing in graph models.
- The subject, predicate and object in terms of basic data graphs and RDF statements.
- A basic familiarity with the layout of an RDF document.

## II. Introducing RDF/XML

graph data model이 SW에서 데이터를 저장하는 모델이라면, RDF는 그것을 만드는 포맷이다. 앞 장에서, 우리는 graph data와 RDF를 살펴봤다. 이제 우리는 RDF/XML - 웹에서 가장 인기 있는 RDF 포맷들 중의 하나 - 을 사용하여 그래프 데이터를 작성하는데 필요한 기본 개념을 설명할 것이다.

이미 여러분은 graph database의 개념을 살펴봤고, 계층형과 관계형 데이터 모델과 같은 전통적인 데이터 저장 형태와 이것을 비교도 해 보았다.

이제 간단하게 RDF (Resource format)을 살펴보고, 주체(subject), 술어(predicate 또는 property), 그리고 객체(object)로 이루어진 지시문을 어떻게 정의하는지에 대해서도 알아보았으며, subject->predicate->object relationship를 triple이라 부른다는 것도 알았다. 또한 여러분은 RDF가 시멘틱 데이터의 웹을 구축하는 기본 포맷이라는 것도 배웠다.

이번 강좌에서, 여러분은 한 단계씩 여러분 자신의 RDF 지시문을 구축하는 방법을 배울 것이고, 그래프를 사용하여 그것들을 시각적으로 이해할 수 있게 될 것이다. 그런 다음에, 데이터에 어의나 의미를 추가하는 것에 대해 생각해 볼 것이고, 끝으로 이것이 제공하는 커다란 장점에 대해서도 이해하게 될 것이다.

위의 내용을 가장 잘 수행할 수 있도록, 먼저 간단한 RDF 다큐먼트의 예를 가지고 한 단계씩 알아보기로 한다.

### 2.1 Building An RDF document

#### >Add The RDF document Root Tag

먼저, RDF root node를 다음과 같이 추가해 보자:

##### 1. <rdf:RDF

```

2.   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
3.
4.   <!-- Body Code Omitted -->
5.
6. </rdf:RDF>

```

위의 예에 있는 두 번째 줄에서, 여러분은 표준화된 W3.org namespace인

**`http://www.w3.org/1999/02/22-rdf-syntax-ns#`**

라는 URI를 보게 될 것이다. 이 namespace는 다른 컴퓨터 리더(reader)에게 이 태그로 봉해져 있는(enclosing) 다큐먼트가 RDF 다큐먼트라는 것과 여기서 사용된 `rdf:RDF` tag들이 이 namespace에 포함되어 있다는 것을 알려주는 것이다.

위의 다큐먼트에서 namespace를 표현하고 있는 RDF node가 바로 이 RDF 다큐먼트의 roots node이다.

#### >Add A Statement

RDF 다큐먼트에는 하나 이상의 지시문이 포함될 수 있다. 간단하게 우리는 하나를 추가할 것이다. RDF/XML에서 주체(subject)를 정의하는 방법은 `<rdf:Description>` tag를 사용하는 것이다. 다음과 같이 붉은 색 지시문을 추가해 보자:

```

01. <rdf:RDF
02.   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
03.
04.   <rdf:Description rdf:about="http://www.linkeddatatools.com/clothes#t-shirt">
05.
06.       <!-- Statement Code Omitted -->
07.
08.   </rdf:Description>
09.
10. </rdf:RDF>

```

위에서 붉은 색으로 표현된 `<rdf:Description>` tag가 의미하는 것은 간단하게 말해서 다음과 같다:

"나는 객체 `t-shirt`에 대하여(about) 정의(describe)하며, 그것의 유일한 ID는 `http://www.linkeddatatools.com/clothes#t-shirt` 이다."

#### 주목!!!

`<rdf:Description>`은 `about` 속성에 의해 지정된 resource에 대한 URI 정보를 갖고 있는 container 이다. 다음의 예에서 각각의 resources는 books이고, 그것의 식별자(URI)는 서명



이다. 그리고 각각의 책에는 한명의 저자와 페이지 수만 표현되어 있다.

좀 이해가 됐나요? 계속 배워봅시다.

## >Add Predicates

여러분이 주체에 대해 정의하면서 그것의 고유한 ID를 지정해 놓았지만, 그 주체의 특성에 대해서는 어떠한 것도 정의하지 않았다. RDF 지시문에서는 RDF 전문용어로 속성을 의미하는 properties나 predicates를 사용하여 해당 주체의 특성들을 정의한다.

간단하게, T-shirt의 속성 중 하나인 size를 다음과 같이 추가해 보자:

```
01. <rdf:RDF
02.   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
03.   xmlns:feature="http://www.linkeddatatools.com/clothing-features#">
04.
05.   <rdf:Description rdf:about="http://www.linkeddatatools.com/clothes#t-shirt">
06.
07.       <feature:size>12</feature:size>
08.
09.   </rdf:Description>
10.
11. </rdf:RDF>
```

위의 7번째 줄을 보자. 간단하게 말해서 이 주체는 <feature:size> 태그이름으로 기술된 한 개의 속성을 갖고 있으며, 이 속성의 문자 값은 12이다. RDF 전문용어로 이것이 바로 지시문(statement)이다.

그리고 3번째 줄에 이 속성이름뿐만 아니라 이 객체에서 사용되는 속성이름의 namespace에 대한 URI가 표시되어 있다는 것을 확인하라.

끝으로, T-shirt의 색깔인 color 속성을 하나 더 다음과 같이 추가해 보자:

```
01. <rdf:RDF
02.   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
03.   xmlns:feature="http://www.linkeddatatools.com/clothing-features#">
04.
05.   <rdf:Description rdf:about="http://www.linkeddatatools.com/clothes#t-shirt">
06.
07.       <feature:size>12</feature:size>
08.       <feature:color rdf:resource="http://www.linkeddatatools.com/colors#white"/>
09.
10.   </rdf:Description>
```

11.

12. `</rdf:RDF>`

이 `<feature:color>` 속성의 표현은 `<feature:size>`와 다르다는 것을 알았을 것이다. 지난 번 속성에서는 문자 값 12가 있었던 반면에, 이번 것에는 또 다른 지시문의 `subject(ID)`를 참조하도록 지정해 놓았다. 이러한 표현이 옳은 것이다. RDF에서 `object`는 다른 지시문에 있는 `subject`를 참조할 수 있다.

!!! 속성 `rdf:resource`는 다른 `rdf:Description` element에서 정의하고 있는 resource에 about한 정보를 입력하는데 사용될 수 있다.

다시 본론으로 돌아가서, `<feature:color>`의 속성은 3번째 줄에 있는 `xmlns:feature`에 이미 포함되어 있다는 것도 이해하여야 한다. 다시 말해서, `xmlns:feature`의 이름리스트에는 이미 `size`와 `color`라는 속성이름이 포함되어 있다는 것이다.

!!! RDF 다크먼트에 있는 주체 또한 다른 RDF 지시문에서 속성의 객체처럼 참조될 수 있다. 이것은 RDF 초보자들에게는 개념적으로 혼란스러울 수 있다.

따라서 이것은 간단하게 "이 주체는 ID가 `http://www.linkeddatatools.com/colors#white`인 지시문을 참조하고 있는 객체와 더불어 `feature:color`이라는 이름의 한 개의 속성을 가지고 있다"라고 말하고 있다.

## 2.2 Breaking Down The Statement

이제 간단한 예의 RDF 다크먼트를 살펴보고, 우리가 배운 것을 근거로 지시문의 구성부분을 분석해 보자:

1. `<rdf:Description rdf:about="subject">`
2.     `<predicate rdf:resource="object" />`
3.     `<predicate>literal value</predicate>`
4. `</rdf:Description>`

이미 여러분은 지시문의 주체(what the statement is about), 그리고 두 가지 형태의 속성 (다른 RDF 지시문을 참고하도록 하는 resources 그리고 문자값과 에 대하여 알게 되었다.

주목 - `rdf:Description` element는 단일 container 안에 하나 이상의 지시문을 집단화할 수 있게 허용하고 있다. 위와 같은 일반적인 형태에는 사실상 한 개의 속성과 한 개의 객체 즉, `literal`과 `resource`, 두 가지가 함께 동일한 주체를 참조하는 두 개의 지시문이 포함되어 있다.

## 2.3 A More Thorough Example

앞에서 그래프 데이터에서 다루었던 보다 복잡한 예로 되돌아가 보자:

```
01.<?xml version="1.0" encoding="UTF-8"?>
02.
03.<rdf:RDF
04.xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
05.xmlns:dc="http://purl.org/dc/elements/1.1/"
06.xmlns:region="http://www.country-regions.fake/">
07.
08.<rdf:Description rdf:about="http://en.wikipedia.org/wiki/Oxford">
09.<dc:title>Oxford</dc:title>
10.<dc:coverage>Oxfordshire</dc:coverage>
11.<dc:publisher>Wikipedia</dc:publisher>
12.<region:population>10000</region:population>
13.<region:principaltown rdf:resource="http://www.country-regions.fake/oxford"/>
14.</rdf:Description>
15.
16.</rdf:RDF>
```

여러분의 이해력을 테스트하고 보완할 분야를 알기 위하여, 위의 RDF 다큐먼트를 아래와 같이 구분할 수 있는지를 스스로 확인해 보라:

- Subject of the statement?
- Predicates of the statement(including whether they are resources or literals)?
- Objects referenced by the **resource** predicates?

이제 RDF 다큐먼트에 대해 이해했고, 그것이 데이터 그래프와 어떤 관계가 있는지를 알았다면, 여러분은 RDF graph data로 시멘틱즈를 모델링하는 방법에 대하여 알 준비가 끝난 것이다.

## 2.4 A Quick Recap Of URIs And XML Namespaces

앞에서 사용된 <http://www.linkeddatatools.com/clothes#t-shirt>처럼 우리가 이제까지 사용해 왔던 고유한 IDs를 Uniform Resource Identifiers, 또는 줄여서 URIs라 부른다. 우리는 이제까지 아무런 설명없이 지시문의 주체, 속성, 객체에 고유한 IDs를 제공하기 위하여 URIs를 사용해 왔다.

URIs는 전 세계적으로 데이터 교환을 가능하게 만들자는 RDF의 목적을 달성하는데 너무나 중요하기 때문에, 우리는 이제 신속하게 URIs에 대해 다시 살펴보자.

## >XML Namespace URIs

앞에 있던 RDF 다큐먼트 예로 다시 가 보자. T-shirt size 속성은 아래의 7번째 줄에서 `<feature:size>`란 이름을 가지고 있다는 것을 알 수 있다:

```
01.<rdf:RDF
02.xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
03.xmlns:feature="http://www.linkeddatatools.com/clothing-features#">
04.
05.<rdf:Description rdf:about="http://www.linkeddatatools.com/clothes#t-shirt">
06.
07.<feature:size>12</feature:size>
08.<feature:color rdf:resource="http://www.linkeddatatools.com/colors#white"/>
09.
10.</rdf:Description>
11.
12.</rdf:RDF>
```

그리고 3번째 줄에서, 여러분은 우리가 XML namespace feature를 정의하기 위하여 그것에 URI `http://www.linkeddatatools.com/clothing-features#`라는 namespace를 지정하였다는 것도 주목해야 한다.

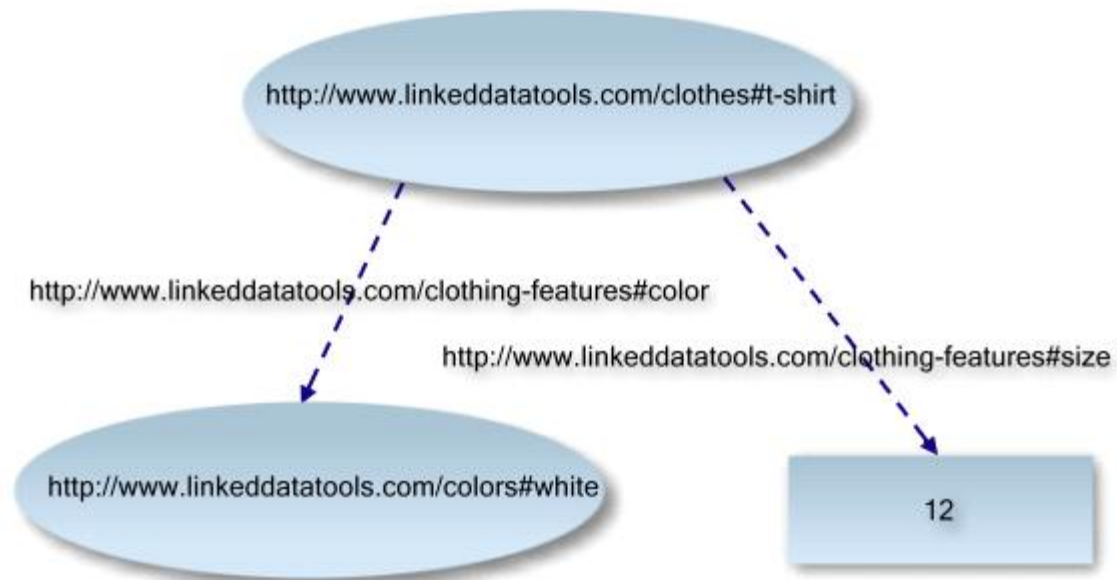
이 namespace의 목적은 단지 동일한 이름의 태그로 인하여 발생하는 이름간의 충돌을 피하기 위한 것이다: 그렇지만 “size”란 이름의 태그가 다른 namespace URIs로 정의된다면, RDF reader는 비록 그것들이 동일한 태그이름라 하더라도 서로 다른 속성들이라는 것을 알게 된다.

또한 추가로 feature:size용으로 완전한 자격을 갖춘 URI를 얻기 위하여, 간단하게 prefix feature를 그것의 완전 이름인 namespace URI인 `http://www.linkeddatatools.com/clothing-features#size`로 대체할 수 있다.

**Note** - RDF에서 XML namespace URIs는 동일한 (태그) 이름을 가진 속성들을 구분하는데 사용된다. 충분히 자격이 있는 URI를 얻기 위하여, 간단하게 namespace prefix를 그것의 namespace URI로 대체할 수 있다.

이제 우리는 완전하게 자격을 갖춘 URIs에 대해 말할 수 있다:

보시다시피, 이것을 RDF graph diagrams으로 완전하게 URIs를 표현하는 것이 항상 쉽지도



편리하지도 않다. 종종 간략 버전(shorthand versions)이 대신 사용된다(다시 말해서, namespace prefix만을 사용하는 것).

이 번 강의를 마치자. 여러분은 다음의 방법에 대해 이해하여야 한다:

- How to write your own basic RDF documents in RDF/XML
- Understand how and why RDF uses URIs to identify subjects, predicates and objects
- How to relate an RDF document to a corresponding data graph with fully qualified URIs

### III. RDF 맞보기

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="rdf.xsl"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">

  <rdf:Description rdf:about="http://en.wikipedia.org/wiki/Oxford">

    <dc:title>Oxford</dc:title>
    <dc:coverage>Oxfordshire</dc:coverage>
    <dc:publisher>Wikipedia</dc:publisher>
  
```

</rdf:Description>

</rdf:RDF>

#### RDF 보기:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body style="font-family:Arial;font-size:12pt;background-color:#EEEEEE">
        <xsl:for-each select="rdf:RDF/rdf:Description">
          <div style="background-color:gold;color:blue;padding:20px">
            <xsl:value-of select="dc:title"/>
            <p/>
            <xsl:value-of select="dc:coverage"/>
            <p/>
            <xsl:value-of select="dc:publisher"/>
          </div>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

#### Qualified DC의 예

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dcterms="http://purl.org/dc/terms/">

  <rdf:Description>
    <dcterms:abstract>The paper resolves the issues of the data model
      draft.
    </dcterms:abstract>
  </rdf:Description>
```

```
<rdf:Description rdf:about="http://purl.org/dc/terms/abstract">
  <rdfs:subPropertyOf
    rdf:resource="http://purl.org/dc/elements/1.1/description"/>
</rdf:Description>
</rdf:RDF>
```

**FIN**