1.Tujuan

- Dapat mengerti penerapan EclipseLink dalam implementasinya pada aplikasi
- Dapat membuat contoh sederhana dengan menggunakan EclipseLink dalam hal implementasi JPA

2.Latar Belakang

Dalam hal pemrograman berbasis object yang telah berkembang dewasa ini, framework untuk memudahkan pemrograman telah banyak digunakan. Dan merupakan suatu usaha tersendiri untuk melakukan suatu migrasi sumber data.

Pengembang di dunia mulai merasakan pentingnya suatu aplikasi yang bersifat dinamis terhadap sumber data. EclipseLink di bangun berdasarkan konsep sumber data dinamis. Yaitu dapat berupa Database dengan implementasi JPA, XML, WebService, dll. Dengan menggunakan EclipseLink kita tidak perlu menghawatirkan perubahan sumber data karena dapat dengan mudah dilakukan.

3.Percobaan

Library yang dibutuhkan:

1. eclipselink.jar
2. persistence.jar
3. mysql.jar

Issue.java

```
package comics.model;


import static javax.persistence.FetchType.LAZY;


import java.io.Serializable;
import java.math.BigDecimal;


import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.ManyToOne;


@Entity

public class Issue implements Serializable {

    private static final long serialVersionUID = 8916573949567829954L;


    @Id

    private int id;

    private String comments;

    private String condition;

    private int copies;

    private int issueNum;

    private double pricePaid;

    @ManyToOne(fetch=LAZY)

    private Title title;

    private BigDecimal value;


    protected Issue() {

        super();

    }


    public Issue(int id, Title title, int issueNum, double pricePaid,
            BigDecimal value) {
        super();
        this.id = id;
        this.issueNum = issueNum;
        this.pricePaid = pricePaid;
        this.title = title;
        this.value = value;
    }

    public String getComments() {
        return this.comments;
    }

    public String getCondition() {
        return this.condition;
    }

    public Integer getCopies() {
        return this.copies;
    }

    public int getId() {
        return this.id;
    }

    public Integer getIssueNum() {
        return this.issueNum;
    }

    public double getPricePaid() {
        return this.pricePaid;
    }

    protected Title getTitle() {
        return this.title;
    }

    public BigDecimal getValue() {
        return this.value;
    }

    public void setComments(String comments) {
        this.comments = comments;
    }

    public void setCondition(String condition) {
        this.condition = condition;
    }

    public void setCopies(Integer copies) {
        this.copies = copies;
    }
```

```
public void setId(int id) {
    this.id = id;
}

public void setIssueNum(Integer issueNum) {
    this.issueNum = issueNum;
}

public void setPricePaid(double pricePaid) {
    this.pricePaid = pricePaid;
}

public void setTitle(Title title) {
    this.title = title;
}

public void setValue(BigDecimal value) {
    this.value = value;
}

/* (non-Javadoc)
 * @see java.lang.Object#toString()
 */
public String toString() {
    return getTitle().getName() + "\t" + getIssueNum();
    }
}
```

Publisher.java

```
package comics.model;


import java.io.Serializable;

import java.util.List;


import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.OneToMany;


import org.eclipse.persistence.annotations.JoinFetch;


@Entity

public class Publisher implements Serializable {

    private static final long serialVersionUID = -5629875995814616811L;


    @Id

    private int id;

    @Column(name="PUB_NAME")

    private String name;

    @JoinFetch
    @OneToMany(mappedBy="publisher")
    private List<Title> titles;

    public Publisher() {
    }

    public Publisher(int id, String name) {
        this.id = id;
        this.name = name;
    }

    /**
     * @return Returns the name.
     */
    public String getName() {
        return name;
    }

    /**
     * @param name The name to set.
     */
    public void setName(String name) {
        this.name = name;
    }

    /**
```

```
     * @return Returns the id.
     */
    public int getId() {
        return id;
    }

    public void setId(int id) {
         this.id = id;
    }

    public String toString() {
        return "Publisher(" + getId() + ", " + getName() + ")";
    }

    public List<Title> getTitles() {
        return titles;
    }

    public void setTitles(List<Title> titles) {
        this.titles = titles;
    }

    public void addTitle(Title aTitle) {
        getTitles().add(aTitle);
        aTitle.setPublisher(this);
    }
```

```
}
```

Title.java

```
package comics.model;


import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;


import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.ManyToOne;

import javax.persistence.OneToMany;

import static javax.persistence.FetchType.LAZY;


@Entity

public class Title implements Serializable {

    private static final long serialVersionUID = 4425594997718097 60L;


    @Id

    private int id;

    @ManyToOne(fetch=LAZY)

    private Publisher publisher;

    @OneToMany(mappedBy="title")
    private List<Issue> issues = new ArrayList<Issue>();
    private String name;
    private String format;



    public Title() {
        setFormat("ongoing");
    }

    public Title(int id, String name) {
        this();
        this.id = id;
        this.name = name;
    }

    public Title(int id, String name, Publisher publisher) {
        this(id, name);
        setPublisher(publisher);
    }

    public String getFormat() {
        return this.format;
    }
    public int getId() {
        return this.id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return this.name;
    }

    public void setFormat(String format) {
        this.format = format;
    }

    public void setName(String title) {
        this.name = title;
    }

    public String toString() {
        return getName() + "\t" + getFormat();
    }
    public Publisher getPublisher() {

        return publisher;

    }


    public void setPublisher(Publisher publisher) {

        this.publisher = publisher;

    }


    public List<Issue> getIssues() {

        return issues;

    }


    public void setIssues(List<Issue> issues) {

        this.issues = issues;

    }


    public void removeIssue(Issue anIssue) {

        this.issues.remove(anIssue);

        anIssue.setTitle(null);

    }


    public void addIssue(Issue anIssue) {

        this.issues.add(anIssue);

        anIssue.setTitle(this);
    }
}
```

persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0"
    xmlns="http://java.sun.com/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
    <persistence-unit name="comics">
        <class>comics.model.Issue</class>
        <class>comics.model.Publisher</class>
        <class>comics.model.Title</class>
        <properties>
            <!-- MySql Client Login -->
            <property name="eclipselink.jdbc.driver"
                value="com.mysql.jdbc.Driver" />
            <property name="eclipselink.jdbc.url"
                value="jdbc:mysql://localhost:3306/eclipselink_sample" />
            <property name="eclipselink.jdbc.user" value="root" />
            <property name="eclipselink.jdbc.password" value="tulalit" />
            <!-- EclipseLink Properties -->
            <property name="eclipselink.jdbc.bind-parameters"
```

```
                value="true" />
            <property name="eclipselink.logging.level" value="FINE" />
            <property name="eclipselink.logging.timestamp"
                value="false" />
            <property name="eclipselink.logging.thread" value="false" />
            <property name="eclipselink.logging.session" value="false" />
        </properties>
    </persistence-unit>
</persistence>
```

JpaDemo

```
package comics;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import comics.model.Publisher;
import comics.model.Title;

public class JpaDemo {
    @SuppressWarnings("unchecked")
    public static void main(String[] args) throws Exception {

        EntityManagerFactory emf = Persistence

                .createEntityManagerFactory("comics");

        EntityManager em = emf.createEntityManager();


        List<Publisher> publishers = em

            .createQuery("select p from Publisher p where p.name like :name
order by p.name")

            .setParameter("name", "%M%").getResultList();

        for (Publisher publisher : publishers) {

            System.out.println(publisher.getName());

            for (Title title : publisher.getTitles()) {

                System.out.println("\t" + title.getName());

            }

        }

        em.close();

        emf.close();

    }

}
```