SOEN 387 Web-based Enterprise Application Design

Stuart Thiel

Concordia University
Department of Computer & Software Engineering

Fall, 2014

Outline

Does ACIDity Fix Everything

- Does it fix Inconsistent Reads?
- . . . not exactly
- Does it fix Lost Updates?
- . . . not alone

Inconsistent Reads and Transactions

- It prevents you reading data made stale by someone else
- It does not prevent you reading data made stale by your current thread

Lost Updates and Transactions

- Alone it does not help anything at all
- But it's the focus of the remainder of this lecture
- . . . and we have a reliable and easy solution

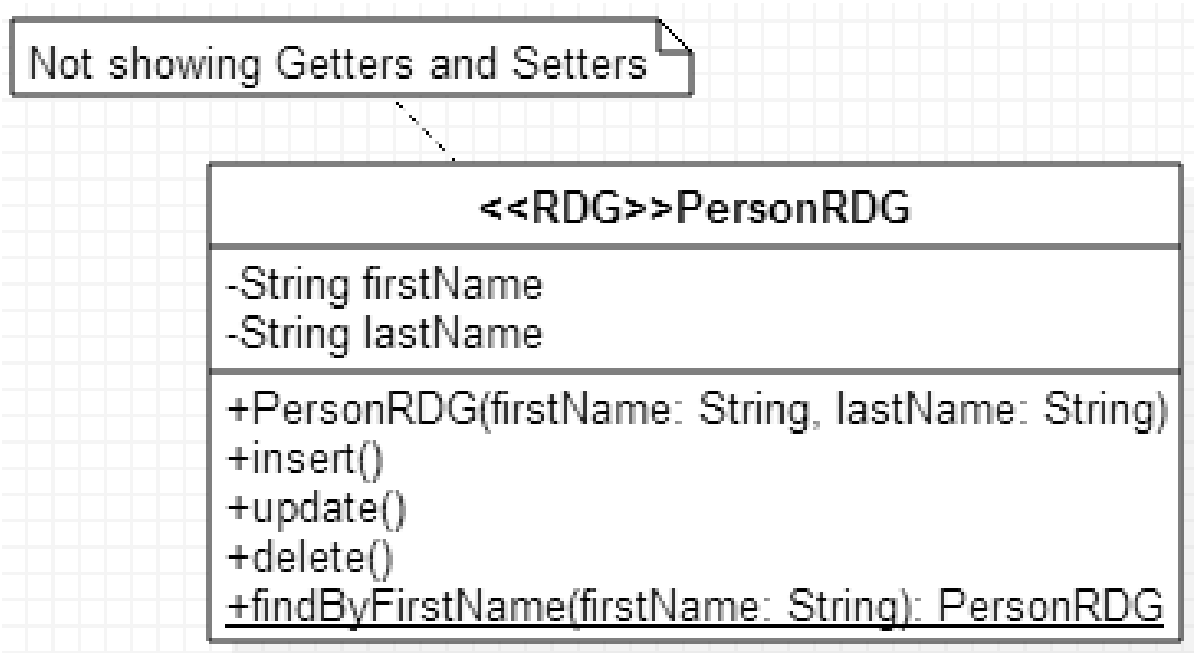
Did Anyone Notice Anything Fishy About Last Tutorial

- . . . inconvenient firealarm is a new SOEN pattern
- Did anyone try to make the same name twice?
- Was that allowed?
- Why is this relevant to concurrency?
- Database concurrency solutions rely on identity
- Identity Field is the pattern “identified” by Fowler

Identity Field

- Each thing has its own id
- id should be unrelated to anything in the item
- Do all things of one type share an id space?
- Do all things share one big id space?
- Where to get ids?

Initial PersonRDG



- Our basic RDG
- Two people with same first name?

PersonRDG Code

```
package org.soen387.tutorial;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class PersonRDG {

        String firstName;
        String lastName;
        public String getFirstName() {
                return firstName;
        }
        public void setFirstName(String firstName) {
                this.firstName = firstName;
        }
        public String getLastName() {
                return lastName;
        }
        public void setLastName(String lastName) {
                this.lastName = lastName;
        }
        public PersonRDG(String firstName, String lastName) {
                super();
                this.firstName = firstName;
                this.lastName = lastName;
        }
```

PersonRDG Code

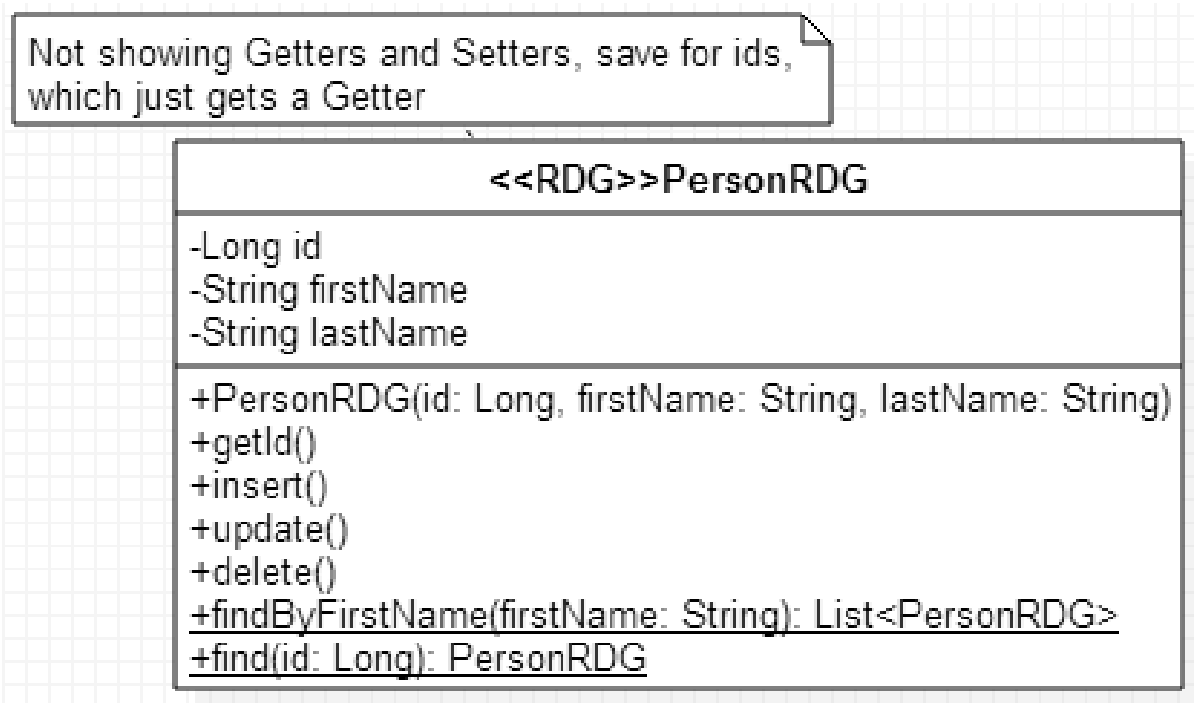
```
public int insert() throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "INSERT INTO t2person (first_name, last_name) " +
                       "VALUES (?,?);";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ps.setString(2, lastName);
        int count = ps.executeUpdate();
        ps.close();
        return count;
}

public int update() throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "UPDATE t2person SET last_name=? " +
                       "WHERE first_name=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, lastName); //note that lastname comes first
        ps.setString(2, firstName);
        int count = ps.executeUpdate();
        ps.close();
        return count;
}
```

PersonRDG Code

```
public static PersonRDG findByFirstName(String firstName)
                throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "SELECT first_name, last_name FROM t2person " +
                        "WHERE first_name=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ResultSet rs = ps.executeQuery();
        PersonRDG p = null;
        if(rs.next()) {
                p = new PersonRDG(rs.getString("first_name"),
                                rs.getString("last_name"));
                rs.close();
                ps.close();
        } else {
                return null;
        }
        return p;
}
```

PersonRDG with Identity Field



- Add ids
- finding by First Name now returns a list (possibly empty)

PersonRDG with Identity Field Code

```
public class PersonRDG {

        Long id;
        String firstName;
        String lastName;
        public Long getId() {
                return id;
        }

        public String getFirstName() {
                return firstName;
        }
        public void setFirstName(String firstName) {
                this.firstName = firstName;
        }
        public String getLastName() {
                return lastName;
        }
        public void setLastName(String lastName) {
                this.lastName = lastName;
        }

        public PersonRDG(Long id, String firstName, String lastName) {
                super();
                this.id = id;
                this.firstName = firstName;
                this.lastName = lastName;
        }
```

PersonRDG with Identity Field Code

```
public int insert() throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "INSERT INTO t2person (id, first_name, last_name) " +
                       "VALUES (?,?,?);";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setLong(1, id);
        ps.setString(2, firstName);
        ps.setString(3, lastName);
        int count = ps.executeUpdate();
        ps.close();
        return count;
}

public int update() throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "UPDATE t2person SET first_name=?, last_name=? " +
                       "WHERE id=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ps.setString(2, lastName);
        ps.setLong(3, id);
        int count = ps.executeUpdate();
        ps.close();
        return count;
}
```

PersonRDG with Identity Field Code

```
public static PersonRDG find(Long id)  throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "SELECT id, first_name, last_name FROM t2person " +
                       "WHERE id=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setLong(1, id);
        ResultSet rs = ps.executeQuery();
        PersonRDG p = null;
        if(rs.next()) {
                p = new PersonRDG(rs.getLong("id"), rs.getString("first_name"),
                                  rs.getString("last_name"));
                rs.close();
                ps.close();
        } else {
                return null;
        }
        return p;
}
```

PersonRDG with Identity Field Code

```
public static List<PersonRDG> findByFirstName(String firstName)
                throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "SELECT id, first_name, last_name FROM t2person " +
                        "WHERE first_name=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ResultSet rs = ps.executeQuery();
        List<PersonRDG> people = new ArrayList<PersonRDG>();
        while(rs.next()) {
                people.add(new PersonRDG(rs.getLong("id"),
                                rs.getString("first_name"),
                                rs.getString("last_name")));
        }
        rs.close();
        ps.close();
        return people;
}
```

Optimistic Offline Lock

- Once you have a way to identify a record
- You get row-level locking, but no temporal identity
- So we add version as well

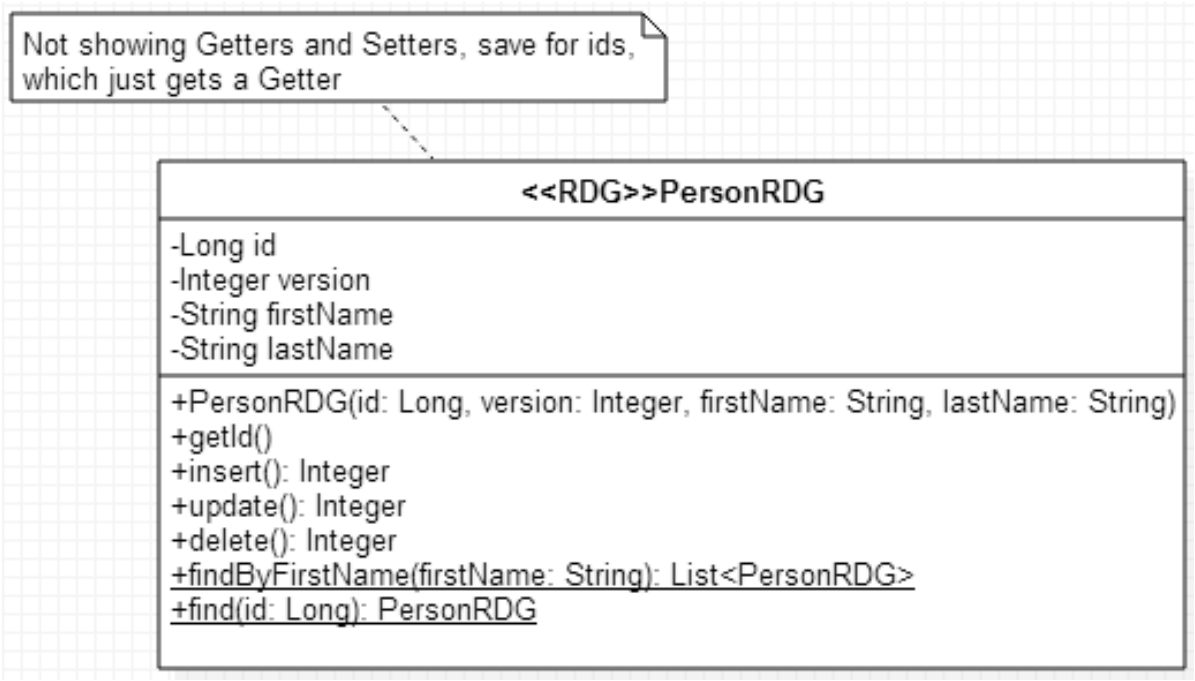
What is a Pessimistic Lock?

- That's a valid solution in some applications
- That's where you lock on behaviour, not identity of record
- Maybe we'll talk more about this later
- Not popular in WEA these days at all

versions

- So, the record gets a version and id in the db, two more columns
- The RDG gets two more fields
- Update on id and version
- Most importantly pass in the version that you are updating along with the id
- Update also increments version
- Need to communicate if you can't match version
- We'll call it a LostUpdateException
-

PersonRDG with Identity Field



- Add version
- update/delete return ints

PersonRDG with Identity Field, Optimistic Offline Lock Code

```
public class PersonRDG {
        Long id;
        Integer version;
        String firstName;
        String lastName;
        public Long getId() {return id;}
        public Long getVersion() {return version;}
        public void setVersion(Integer version) {
                this.version = version;
        }
        public String getFirstName() {return firstName;}
        public void setFirstName(String firstName) {
                this.firstName = firstName;
        }
        public String getLastName() {return lastName;}
        public void setLastName(String lastName) {
                this.lastName = lastName;
        }

        public PersonRDG(Long id, Integer version,
                        String firstName, String lastName) {
                super();
                this.id = id;
                this.version = version;
                this.firstName = firstName;
                this.lastName = lastName;
        }
```

PersonRDG with Identity Field Code

```
public int insert() throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "INSERT INTO t2person (id, version, " +
                       "first_name, last_name) VALUES (?,1,?,?);";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setLong(1, id);
        ps.setString(2, firstName);
        ps.setString(3, lastName);
        int count = ps.executeUpdate();
        ps.close();
        return count;
}

public int update() throws SQLException, LostUpdateException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "UPDATE t2person SET version=version+1, " +
                       "first_name=?, last_name=? WHERE id=? AND version=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ps.setString(2, lastName);
        ps.setLong(3, id);
        ps.setInteger(4, version);
        int count = ps.executeUpdate();
        version++;
        ps.close();
        return count;
}
```

PersonRDG with Identity Field Code

```
public static PersonRDG find(Long id) throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "SELECT id, version, first_name, " +
                       "last_name FROM t2person WHERE id=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setLong(1, id);
        ResultSet rs = ps.executeQuery();
        PersonRDG p = null;
        if(rs.next()) {
                p = new PersonRDG(rs.getLong("id"),
                                  rs.getInteger("version"),
                                  rs.getString("first_name"),
                                  rs.getString("last_name"));
                rs.close();
                ps.close();
        } else {
                return null;
        }
        return p;
}
```

PersonRDG with Identity Field Code

```
public static List<PersonRDG>
        findByFirstName(String firstName) throws SQLException {
        Connection con = HelloWorldTSWithRDG.myCon.get();
        String query = "SELECT id, version, first_name, " +
                       "last_name FROM t2person WHERE first_name=?;";
        PreparedStatement ps = con.prepareStatement(query);
        ps.setString(1, firstName);
        ResultSet rs = ps.executeQuery();
        List<PersonRDG> people = new ArrayList<PersonRDG>();
        while(rs.next()) {
                people.add(new PersonRDG(rs.getLong("id"),
                                rs.getInteger("version"),
                                rs.getString("first_name"),
                                rs.getString("last_name")));
        }
        rs.close();
        ps.close();
        return people;
}
```

Who Takes Care of Lost Update Exception

- Transaction Script
- RDG
- Template View?
- Who takes care of incrementing the version on success for an update?

New Responsibilities

- Starting Transactions
- Ending Transactions
- Dealing with Lost Update Exceptions
- Guess what, we need more patterns to sensibly isolate these

TITLE

-