**CS 54100 – Project**

# BIBA INTEGRITY MODEL

Daniele MIDI – dmidi@purdue.edu

## TASK BREAKDOWN

The project was developed over the course of multiple iterations. In every iteration, the code was refactored and expanded, including both the development of new functionalities, as well as the enhancement in flexibility and "best practices" of the existing ones. Here is the breakdown of tasks and the subsequent iterations in design and development:

### ITERATION 1

- Biba policies research and learning.
- Choice of the policies to be designed and implemented.

### ITERATION 2

- Design and implementation of the database tables.
- Design and implementation of the connection to the Oracle database.
- Design and implementation of the basic interactive console environment.
- Implementation of the basic console commands to impersonate a subject and execute a query.

### ITERATION 3

- Design and implementation of the basic domain entities: Subject, Object, IntegrityLevel.
- Definition of the 5 integrity levels, hard-coded in the program.
- Design and implementation of the SubjectManager and ObjectManager, responsible for loading the predefined subjects and objects into the system runtime. In this iteration, those are hard-coded in the program.
- Design and first draft implementation of the Security engine, able to parse simple queries, extract relevant objects and enforce the basic Biba Strict policy.

### ITERATION 4

- Design and implementation of the database tables for storing subjects, objects and integrity levels.
- Refactoring of SubjectManager and ObjectManager to read subjects and objects from the database (no longer hard-coded).
- Introduction of the IntegrityLevelManager to load the integrity levels into the system runtime (no longer hard-coded).

### ITERATION 5

- Design of the BibaPolicy interface, to define a common interface for multiple policies and allow extensibility and introduction of new policies.
- Design and implementation of the Ring policy and the Subject Low-Watermark policy.
- Refactoring of the Security engine to delegate the policy enforcement to a pluggable BibaPolicy.
- Implementation of the console command to switch policy.

### ITERATION 6

- Design and implementation of the advanced Query Parser, to analyze the queries in depth, extract relevant objects and manage subqueries, read+write queries and complex syntaxes.

# DATABASE DESIGN

## DATA TABLES

```
CREATE TABLE zipcodes (
    zip INT,
    city VARCHAR(30),
    PRIMARY KEY(zip),
    CHECK (zip>=0 AND zip<=99999)
);

CREATE TABLE employees (
    eno INT,
    ename VARCHAR(30),
    zip INT,
    hdate DATE,
    PRIMARY KEY(eno),
    FOREIGN KEY(zip) REFERENCES zipcodes,
    CHECK (zip>=0 AND zip<=99999),
    CHECK (eno>=0 AND eno<=9999)
);

CREATE TABLE parts (
    pno INT,
    pname VARCHAR(30),
    qoh INT,
    price NUMBER(6,2) NOT NULL,
    olevel INT,
    PRIMARY KEY(pno),
    CHECK (pno>=0 AND pno<=99999),
    CHECK (qoh>=0),
    CHECK (price>=0.0)
);

CREATE TABLE customers (
    cno INT,
    cname VARCHAR(30),
    street VARCHAR(30),
    zip INT,
    phone VARCHAR(12),
    PRIMARY KEY(cno),
    FOREIGN KEY(zip) REFERENCES zipcodes,
    CHECK (cno>=0 AND cno<=99999),
    CHECK (zip>=0 AND zip<=99999)
);

CREATE TABLE orders (
    ono INT,
    cno INT,
    eno INT,
    received DATE,
    shipped DATE,
    PRIMARY KEY(ono),
    FOREIGN KEY(cno) REFERENCES customers ON DELETE CASCADE,
    FOREIGN KEY(eno) REFERENCES employees ON DELETE CASCADE,
    CHECK (ono>=0 AND ono<=99999),
    CHECK (cno>=0 AND cno<=99999),
    CHECK (eno>=0 AND eno<=9999)
);

CREATE TABLE odetails (
    ono INT,
    pno INT,
    qty INT,
    PRIMARY KEY(ono, pno),
    FOREIGN KEY(ono) REFERENCES orders ON DELETE CASCADE,
```

```
    FOREIGN KEY(pno) REFERENCES parts ON DELETE CASCADE,
    CHECK (ono>=0 AND ono<=99999),
    CHECK (pno>=0 AND pno<=99999),
    CHECK (qty>0)
);
```

### BIBA META-TABLES

```
CREATE TABLE integritylevels (
    il INT,
    label VARCHAR(30) NOT NULL,
    PRIMARY KEY(il)
);
CREATE TABLE subjects (
    username VARCHAR(50),
    description VARCHAR(50) NOT NULL,
    il INT NOT NULL,
    istrusted CHAR(1) NOT NULL,
    PRIMARY KEY(username),
    FOREIGN KEY(il) REFERENCES integritylevels ON DELETE CASCADE
);
CREATE TABLE objects (
    oname VARCHAR(50),
    il INT NOT NULL,
    PRIMARY KEY(oname),
    FOREIGN KEY(il) REFERENCES integritylevels ON DELETE CASCADE
);
```

## CONCEPTUAL DESIGN

### Integrity levels

The integrity levels available are, in increasing order of integrity:

1. Very Low
2. Low
3. Medium
4. High
5. Very High

### Objects

The objects are the tables defined in the previous section. Each object is assigned an initial integrity level, as follows:

| OBJECT | INTEGRITY LEVEL |
|---|---|
| **zipcodes** | Low |
| **employees** | High |
| **parts** | Very High |
| **customers** | High |
| **orders** | Medium |
| **odetails** | High |

### Subjects

We define several subjects. Each subject is assigned an initial integrity level, which may change during the execution of the program according to the possible watermarking policies. The changes are applied only per-

session, i.e. are not permanent from execution to execution, to allow an easy test reproducibility. Moreover, some subjects are defined as trusted, and the Biba Ring policy will apply to them.

The subjects, their initial integrity levels and their trustworthiness are defined as follows:

| SUBJECT | DESCRIPTION | INTEGRITY LEVEL | TRUSTED |
|---|---|---|---|
| **admin** | CEO user account | Very High | Yes |
| **inventory** | Inventory management script | Very High | No |
| **customerservice** | Customer service representative user account | Medium | No |
| **hr** | Human Resources representative user account | High | Yes |
| **audit** | Auditing script, periodically scheduled | Very Low | No |
| **guest** | Authorized guest user account | Low | No |

# APPLICATION DESIGN

## SUPPORTED OPERATIONS

The application is able to interactively receive commands from the user, interpret and execute them. The available commands allow the user to:

- Impersonate a particular subject
- Set the desired Biba policy
- Execute SQL queries on the underlying Oracle database

The system loads integrity levels, subjects and objects from the database at runtime and uses them throughout the lifetime of the current application session. The integrity levels associated with subject and objects are dynamic and can change according to the currently active Biba policy.

## POLICIES

The Biba policies implement a common interface in order to make the system extensible by simply defining new policies and plugging them into the system. The built-in policies that are currently implemented are the Strict policy, the Subject Low-watermark policy and the Ring policy.

The integrity constraints are enforced at the database relation granularity level.

## ADVANCED QUERY SUPPORT

The system supports complex queries. In particular, it supports complex SELECTs, INSERTs, UPDATEs, DELETEs and any combination of them, through nested queries. We are able to parse and analyze a very flexible syntax for SQL queries and always extract the relevant objects. For example, our system is capable of dealing with queries that range from syntaxes as simple as:

```
SELECT * FROM Table1
```

to queries as complex as:

```
SELECT T1.a, T2.b
FROM Table1 T1 LEFT OUTER JOIN Table2 T2 ON T1.c=T2.d
WHERE T1.f NOT IN
    (SELECT T3.f FROM Table3 T3 GROUP BY f HAVING SUM(a)>=10)
```

Thanks to the support for nested queries, the system also supports queries that are both read and write queries at the same time. An example of such query can be the following:

```
INSERT INTO Table1
```

```
SELECT * FROM Table2
```

The Query Parser will detect this query as performing a read operation on the object `Table2`, and a write operation on the object `Table1`. The integrity levels of all the involved objects will be checked and validated according to the currently active policy.

The query will be processed and actually executed against the backing database if and only if ALL the integrity checks are successfully passed. This means that if, according to the current Biba policy, the current subject is allowed to read from `Table2` but not to write to `Table1`, then the whole query will be aborted.

In order to keep the parser simple, all the subqueries are detected only if enclosed in parenthesis. Therefore the previous sample complex query will be correctly parsed and analyzed only if it is expressed in the following way:

```
INSERT INTO Table1
(SELECT * FROM Table2)
```

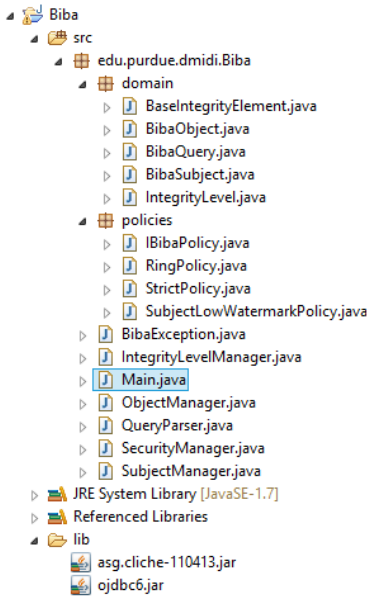
This is however not a big limitation in the versatility and expressiveness of the queries that our system is able to deal with.

### DISCUSSION AND FUTURE WORK

Even though the system is able to process complex read+write queries involving different objects, and enforce different read and write restrictions, it is not currently able to understand the internal database model relationships between the various database relations. This means for example that, if because of a CASCADE constraint, a deletion on an object will trigger a deletion on another object, only the integrity level of the first object will be checked, as the system does not have a knowledge of the side effect triggered by this action on the second object. Future extensions of this project might tackle this issue.

## IMPLEMENTATION

The application is organized in different packages.




The base package is `edu.purdue.dmidi.Biba`. It contains the main functionality classes of the application:

- `IntegrityLevelManager`: loads the integrity levels from the database and makes them available to the rest of the application
- `ObjectManager`: loads the objects on which to enforce integrity policies from the database, and makes them available to the rest of the application
- `SubjectManager`: loads the subjects on which to enforce integrity policies from the database, and makes them available to the rest of the application
- `QueryParser`: is responsible for parsing complex queries (also managing nested queries via recursion), extracting the relevant objects for every possible operation (SELECT, INSERT, UPDATE, DELETE)
- `SecurityManager`: is the component responsible for orchestrating the other component. It is able to execute the security commands issued by the user, such as setting a certain Biba policy or validating a query.

- `Main`: is the entry point for the application. It loads all the components and compose them via manual dependency injection. For the interactive command line console, we leverage the open source “Cliche” library. As a database drive, we use the official OJDBC driver.

The subpackage `edu.purdue.dmidi.Biba.domain` contains the domain classes. The `BibaObject` and `BibaSubject` classes represent objects and subjects, respectively. They both inherit from the abstract base type `BaseIntegrityElement`. They both have an initial integrity level, and a current integrity level (that captures the dynamic evolution of the integrity enforcement).

Last, the `edu.purdue.dmidi.Biba.policies` package contains the common interface to be implemented by all the policies (in order to use polymorphism to allow extensibility to the application), as well as the built-in Biba policies that we already implemented (Strict, Subject Low-Watermark, Ring).

# TEST EXECUTION

```
STARTING SYSTEM INITIALIZATION...

Loaded integrity level Very Low
Loaded integrity level Low
Loaded integrity level Medium
Loaded integrity level High
Loaded integrity level Very High
Loaded Subject 'admin' with integrity level Very High (trusted)
Loaded Subject 'inventory' with integrity level Very High
Loaded Subject 'customerservice' with integrity level Medium
Loaded Subject 'hr' with integrity level High (trusted)
Loaded Subject 'audit' with integrity level Very Low
Loaded Subject 'guest' with integrity level Low
Loaded Object 'zipcodes' with integrity level Low
Loaded Object 'employees' with integrity level High
Loaded Object 'parts' with integrity level Very High
Loaded Object 'customers' with integrity level High
Loaded Object 'orders' with integrity level Medium
Loaded Object 'odetails' with integrity level High

SYSTEM INITIALIZATION COMPLETE.

Biba policy enforcement

> l customerservice
Current subject: customerservice.

> q 'SELECT * FROM orders'
SELECT * FROM orders
SELECT:
 - orders
Reading Integrity level: Medium

--> Query read IL: Medium; Query write IL: <none>; Subject IL: Medium
ONO             CNO             ENO             RECEIVED SHIPPED
1020            1111            1000            1994-12-10 00:00:00.01994-12-12 00:00:00.0
1021            1111            1000            1995-01-12 00:00:00.01995-01-15 00:00:00.0
1022            2222            1001            1995-02-13 00:00:00.01995-02-20 00:00:00.0
1023            3333            1000            1997-06-20 00:00:00.0null
******************


> q 'SELECT zip, city FROM zipcodes'
SELECT zip, city FROM zipcodes
SELECT:
 - zipcodes
Reading Integrity level: Low

--> Query read IL: Low; Query write IL: <none>; Subject IL: Medium
ACCESS DENIED.
******************
```

```
> q 'SELECT cno FROM customers'
SELECT cno FROM customers
SELECT:
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Medium
CNO
1111
2222
3333
******************


> q 'SELECT C.cno FROM customers C JOIN parts P'
SELECT C.cno FROM customers C JOIN parts P
SELECT:
 - parts
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Medium
ORA-00905: missing keyword
******************


> l admin
Current subject: admin.

> q 'SELECT * FROM orders'
SELECT * FROM orders
SELECT:
 - orders
Reading Integrity level: Medium

--> Query read IL: Medium; Query write IL: <none>; Subject IL: Very High
ONO               CNO               ENO               RECEIVED SHIPPED
1020              1111              1000              1994-12-10 00:00:00.01994-12-12 00:00:00.0
1021              1111              1000              1995-01-12 00:00:00.01995-01-15 00:00:00.0
1022              2222              1001              1995-02-13 00:00:00.01995-02-20 00:00:00.0
1023              3333              1000              1997-06-20 00:00:00.0null
******************


> q 'SELECT zip, city FROM zipcodes'
SELECT zip, city FROM zipcodes
SELECT:
 - zipcodes
Reading Integrity level: Low

--> Query read IL: Low; Query write IL: <none>; Subject IL: Very High
ZIP                                      CITY
67226                                    Wichita
60606                                    Fort Dodge
50302                                    Kansas City
54444                                    Columbia
66002                                    Liberal
61111                                    Fort Hays
******************


> q 'SELECT cno FROM customers'
SELECT cno FROM customers
SELECT:
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Very High
CNO
1111
2222
3333
******************
```

```
> q 'SELECT C.cno FROM customers C JOIN parts P'
SELECT C.cno FROM customers C JOIN parts P
SELECT:
 - parts
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Very High
ORA-00905: missing keyword
******************


> sp subjectwatermark
New policy enforced.

> l inventory
Current subject: inventory.

> q 'SELECT * FROM orders'
SELECT * FROM orders
SELECT:
 - orders
Reading Integrity level: Medium

--> Query read IL: Medium; Query write IL: <none>; Subject IL: Very High
[Subject's IL lowered to Medium]
ONO             CNO             ENO             RECEIVED SHIPPED
1020            1111            1000            1994-12-10 00:00:00.01994-12-12 00:00:00.0
1021            1111            1000            1995-01-12 00:00:00.01995-01-15 00:00:00.0
1022            2222            1001            1995-02-13 00:00:00.01995-02-20 00:00:00.0
1023            3333            1000            1997-06-20 00:00:00.0null
******************


> q 'SELECT zip, city FROM zipcodes'
SELECT zip, city FROM zipcodes
SELECT:
 - zipcodes
Reading Integrity level: Low

--> Query read IL: Low; Query write IL: <none>; Subject IL: Medium
[Subject's IL lowered to Low]
ZIP                                     CITY
67226                                   Wichita
60606                                   Fort Dodge
50302                                   Kansas City
54444                                   Columbia
66002                                   Liberal
61111                                   Fort Hays
******************


> q 'SELECT cno FROM customers'
SELECT cno FROM customers
SELECT:
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Low
CNO
1111
2222
3333
******************


> q 'SELECT C.cno, P.pno FROM customers C JOIN parts P'
SELECT C.cno, P.pno FROM customers C JOIN parts P
SELECT:
 - parts
 - customers
Reading Integrity level: High

--> Query read IL: High; Query write IL: <none>; Subject IL: Low
```

```
ORA-00905: missing keyword
******************


> q 'INSERT INTO parts (SELECT cno FROM customers)'
INSERT INTO parts (SELECT cno FROM customers)
SELECT:
 - customers
INSERT:
 - parts
Reading Integrity level: High
Writing Integrity level: Very High

--> Query read IL: High; Query write IL: Very High; Subject IL: Low
ACCESS DENIED.
******************


> q 'SELECT cno FROM customers WHERE cno IN (SELECT O.cno FROM orders O WHERE cno IN (SELECT O.cno
FROM orders O))'
SELECT cno FROM customers WHERE cno IN (SELECT O.cno FROM orders O WHERE cno IN (SELECT O.cno FROM
orders O))
SELECT:
 - orders
 - customers
Reading Integrity level: Medium

--> Query read IL: Medium; Query write IL: <none>; Subject IL: Low
CNO
2222
1111
3333
******************
```