INSERT and UPDATE

executeUpdate

Inserting a new row using JDBC

```
public void testInsert(){
    String SQL = "INSERT INTO municipalities(Name, URL, HTTPS, Server) " +
            "VALUES('Ingalunda stad', 'http://ingalunda.se', 0, 'IIS')";
    Statement stm=null;
    try{
        stm = con.createStatement();
        stm.executeUpdate(SQL);
    }catch(Exception e){
        System.err.println("Exception inserting: " + e.getMessage());
    }finally{
        closeIt(stm);
    }
}
```

Updating a row

```
public int updateHTTPS(String name, boolean https){
    String SQL = "UPDATE municipalities SET HTTPS=" +
                 (https?"1":"0") +
                 " WHERE Name='" + name + "'";
    Statement stm = null;
    int rows = -1;
    try{
        stm  = con.createStatement();
        rows = stm.executeUpdate(SQL);
    }catch(Exception e){
        System.err.println("Exception updating https for " + name
                           + " " + e.getMessage());
    }finally{
        closeIt(stm);
        return rows;
    }
}
```

Updating a row - was anything updated?

```
public int updateHTTPS(String name, boolean https){
     String SQL = "UPDATE municipalities SET HTTPS=" +
         (https?"1":"0") +
         " WHERE Name='" + name + "'";
     Statement stm = null;
     int rows = -1;
     try{
         stm  = con.createStatement();
         rows = stm.executeUpdate(SQL);
     }catch(Exception e){
         System.err.println("Exception updating https for " + name
                            + " " + e.getMessage());
     }finally{
         closeIt(stm);
         return rows;
     }
 }
```

Deleting a row

```
public int deleteCity(String name){
     String SQL = "DELETE FROM municipalities" +
         " WHERE Name='" + name + "'";
     Statement stm = null;
     int rows = -1;
     try{
         stm  = con.createStatement();
         rows = stm.executeUpdate(SQL);
     }catch(Exception e){
         System.err.println("Exception deleting " + name
                            + " " + e.getMessage());
     }finally{
         closeIt(stm);
         return rows;
     }
 }
```

Running a few tests

```
rikard@T530:~/iths/database$ javac db/main/Main.java && java -cp
.:driver/sqlite-jdbc-3.8.11.2.jar db.main.Main
Running db.Main...
We have a connection
Ale kommun HTTP only
Alingsås kommun HTTPS support
Alvesta kommun HTTP only
Aneby kommun HTTP only
Arboga kommun HTTP only
==================
Ingalunda stad - http://ingalunda.se - only http - IIS
Setting HTTPS to true for Ingalunda
1 rows were updated
Ingalunda stad - http://ingalunda.se - https support - IIS
==================
Deleting Ingalunda.
1 rows were deleted
Fetching Ingalunda
No such city: Ingalunda stad
```

A few words on design

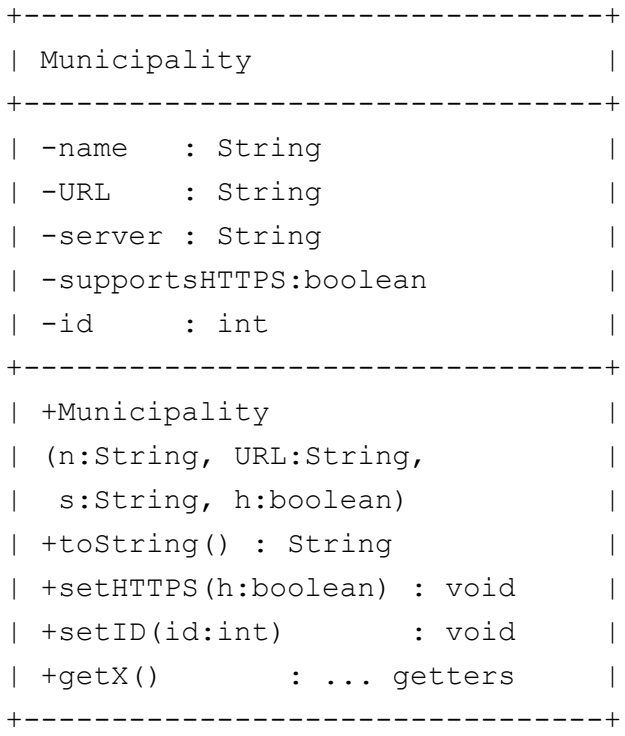
The database class, how much should it know about the tables? About the queries?

The Main class, how much should it know about the database?

What would we much rather do, using Java?

Layers (light version)

Let’s create a class Municipality:



Let’s create a db util class

```
public static DBUtils getInstance()
public boolean hasConnection()
public ResultSet executeQuery(String sql)
public int executeUpdate(String sql)
public void closeIt(AutoCloseable it)
```

Next, let's create an interface MunicipalityDB

```
public interface MunicipalityDB{
    public List<Municipality>getAllCities();
    public void updateCity(Municipality m);
    public void deleteCity(Municipality m);
    public void addCity(Municipality m);
    public Municipality getByName(String name);
}
```

Why an interface? We can change to a different implementation later on, e.g. if the database changes.

Finally, let's create an implementation

```
public class MyMunicipalities implements MunicipalityDB{
    public List<Municipality>getAllCities(){...}
    public void updateCity(Municipality m){...}
    public void deleteCity(Municipality m){...}
    public void addCity(Municipality m){...}
    public Municipality getByName(String n){...}
}
```

Let's look at getByName()...

```
public Municipality getByName(String name){
  String SQL="SELECT * FROM municipalities WHERE name='"+name+"'";
  System.out.println("DEBUG: SQL: " + SQL);
  ResultSet rs = db.executeQuery(SQL);
  Municipality m = null;
  try{
    if(rs.next()){
      m = new Municipality(rs.getString("Name"),   rs.getString("URL"),
                           rs.getString("Server"), rs.getBoolean("HTTPS"));
      m.setID(rs.getInt("MunicipalityID"));
    }
    return m;
  }catch(Exception e){
    System.err.println("getByName: " + e.getMessage());
  }finally{
    db.closeIt(rs);
  }
  return null;
}
```

What could possibly go wrong with getByName()...

```
public Municipality getByName(String name){
  String SQL="SELECT * FROM municipalities WHERE name='"+name+"'";
  System.out.println("DEBUG: SQL: " + SQL);
  ResultSet rs = db.executeQuery(SQL);
  Municipality m = null;
  try{
    if(rs.next()){
      m = new Municipality(rs.getString("Name"),   rs.getString("URL"),
                           rs.getString("Server"), rs.getBoolean("HTTPS"));
      m.setID(rs.getInt("MunicipalityID"));
    }
    return m;
  }catch(Exception e){
    System.err.println("getByName: " + e.getMessage());
  }finally{
    db.closeIt(rs);
  }
  return null;
}
```

What if we feed the getByName method this?

```
"' or 1=1 --"
```

The resulting name variable will then refer to this:

```
SELECT * FROM municipalities WHERE name='' or 1==1 --'
```

Discussion: “--” means comment. Why do we need to end the attack with --?

What if we inject code in an update statement?

```
"'; drop table municipalities; --"
```

What would happen? ;-)

Let’s say we have a method

```
public int updateHTTPSbyName(String name, boolean https)
```

to which we pass a String with the “name” of the city.

What if we inject code in an update statement?

Instead of a proper name, an attacker sends:

```
"'; drop table municipalities; --"
```

What would happen?

The query would translate to:

```
UPDATE municipalities SET HTTPS=0 WHERE name='' ;drop table municipalities;--'
```

What if we inject code in an update statement?

Instead of a proper name, an attacker sends:

```
"' OR 1=1--"
```

What would happen?

The query would translate to:

```
UPDATE municipalities SET HTTPS=0 WHERE name='' OR 1=1--'
```

How many rows would be affected?

What about foreign key constraints from Java?

Foreign key constraints need to be turned on, also from Java:

```
import org.sqlite.SQLiteConfig; // Dependency
public static final String DB_URL = "jdbc:sqlite:database.db";
public static final String DRIVER = "org.sqlite.JDBC";
public static Connection getConnection() throws ClassNotFoundException {
    Class.forName(DRIVER);
    Connection connection = null;
    try {
        SQLiteConfig config = new SQLiteConfig();
        config.enforceForeignKeys(true);
        connection =
DriverManager.getConnection(DB_URL,config.toProperties());
    } catch (SQLException ex) {}
    return connection;
}
```

Next: PreparedStatement - protect against attacks

Next lecture we'll look at a way to protect ourselves from SQL injection attacks.

Read

http://www.w3schools.com/sql/sql_injection.asp

https://www.owasp.org/index.php/Preventing_SQL_Injection_in_Java