# ../../../../../Desktop/Screen%20Shot%202017-11-14%20at%202.Programming assignment:

# A video game with a database-backed leaderboard

Next we continue with the creation of the database and inserting scores. Every time a user plays the game, when the game is over, a new record must be added to the database that includes the player’s name, the score, and what hero they used. Implementing this in Java and SQL is the goal of Task 2.

**INVADERS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **player** | **hero** | **gametime** | **score** |
| 1 | crono | 1 | 2017-11-14 12:17:01 | 50 |
| 2 | crono | 2 | 2017-11-14 12:17:01 | 470 |
| 3 | marle | 3 | 2017-11-14 12:17:01 | 320 |
| 4 | lucca | 2 | 2017-11-14 12:17:01 | 5630 |
| 5 | crono | 1 | 2017-11-14 12:17:01 | 480 |

## Task 2: Create and insert

1. You need to create the table on the database server. You can use PhpMiniAdmin or PhpAdmin. As you can see from the example above, there are 5 columns.
   1. We need a data type for each column. I think you know how to figure most of them out. The only new one is gametime. The correct type for this is called TIMESTAMP.
   2. We will rely on the DBMS to provide the value for gametime each time a new record is added. To do that, we provide a default value using CURRENT\_TIMESTAMP.
   3. We will also rely on the DMBS to provide the value for the id column. We use a feature called AUTO\_INCREMENT. It automatically creates ID numbers every time you add a record. Since we haven’t seen this before, I’ll show you how to do it:

CREATE TABLE invaders (

id INT NOT NULL AUTO\_INCREMENT,

player VARCHAR(20) NOT NULL,

hero INT NOT NULL,

gametime TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

score INT NOT NULL,

PRIMARY KEY (id)

) ENGINE=InnoDB

1. Now when you insert, you only need to specify the player, the hero, and the score. So your insert commands will look something like this:

INSERT

INTO invaders (player, hero, score)

VALUES ('crono', 1, 50)

You can try it out in PhpMiniAdmin if you want to see how auto increment and the timestamp work.

1. Now comes the time for Java code.
   1. Open the NetBeans project that you downloaded in Task 1. Play the game once. When the game is over, a message is printed to the console. (You may need to close the game window to see it.)
   2. That message is printed in the updateDatabase method in the ScoreUpdater class in the edu.sxu.databases.invaders.backend package. Open it up and find the spot.
   3. Notice that the JdbcTest class is also in the package. It is the code from Lab 6, and it is there for your reference. Copy the code from that class that loads the database driver and opens the connection to the database. You will have to change the database from “hafh” to your net ID. Hard code your username and password. Test the program at this point to make sure there are no errors.
   4. Finally, you need to run an INSERT command using a prepared statement. The code is like this:

//Record new score in database

String command = "INSERT INTO invaders (player, hero, score) " +

"VALUES (?, ?, ?)";

PreparedStatement stmt;

stmt = conn.prepareStatement(command);

//set parameters (student code)

stmt.executeUpdate();

System.out.println("Score inserted into database.");

* 1. The comment line above that says “student code” needs to be filled in. You need to call setString and setInt on the stmt object to fill in the values for the question marks. We did this in the lab, and there is one example in the JdbcTest example code. Please ask for help if you are stuck on this part.

1. Play the game a few times, using the same screenname and different screennames. Try different heroes. Then check the database table using PhpMiniAdmin. Make sure that in all cases, the database is updated correctly.
2. Upload the ScoreUpdater.java file to Canvas by the posted due date.