

# Inserción de Datos

Tenemos una tabla creada: prueba2. Vamos ahora a introducir datos en la misma. Lo haremos mediante un nuevo programa. IMPORTANTE si utilizamos el mismo programa (al final haremos uno con todo lo visto) hay que tener en cuenta que o bien manejamos la excepción que nos dará al intentar crear una tabla que ya existe, o mejor, añadiremos un IF NOT EXISTS en la sentencia CREATE

El mecanismo es prácticamente el mismo que para crear una tabla, solo que utilizando sentencias INSERT en lugar de CREATE. Pero para hacerlo algo más útil, los datos a insertar los pediremos por teclado, con lo que habrá que construir la sentencia mezclando SQL con las variables java.

Acordaros de añadir la Librería MySQL en el proyecto.

Conectamos con la BD igual que hicimos en el ejercicio anterior:

```
public static void main(String[] args) {
    Connection conn;
    String user = "root";
    String pass = "";
    String url = "jdbc:mysql://localhost/java";

    try {
        conn = DriverManager.getConnection(url, user, pass);
        System.out.println("Conexión establecida.");
    } catch (SQLException e) {
        System.out.println("Error en la conexión. Finalizando programa\n"
            + e.getMessage());
        return;
    }
}
```

Si no se puede crear la conexión, abandonamos el programa.

Ahora vamos a realizar un bucle donde pediremos datos por teclado hasta que el usuario deje en blanco el primer campo:

```

Scanner tec = new Scanner(System.in);
Statement stmt;
try {
    stmt = conn.createStatement();
    while (true) {
        System.out.print("Introduce nombre: ");
        String nombre = tec.nextLine();
        if (nombre.isEmpty()) {
            break;
        }
        System.out.print("Introduce apellido: ");
        String apellido = tec.nextLine();
        System.out.print("Introduce email: ");
        String email = tec.nextLine();
    }
}

```

Y a continuación escribimos la sentencia. Como el primer campo, la clava primaria, es auto incrementable, tenemos dos opciones:

1. Poner su valor a null
2. Especificar los campos y sus valores. Así nos saltamos dicho campo.

Vamos a utilizar la segunda opción:

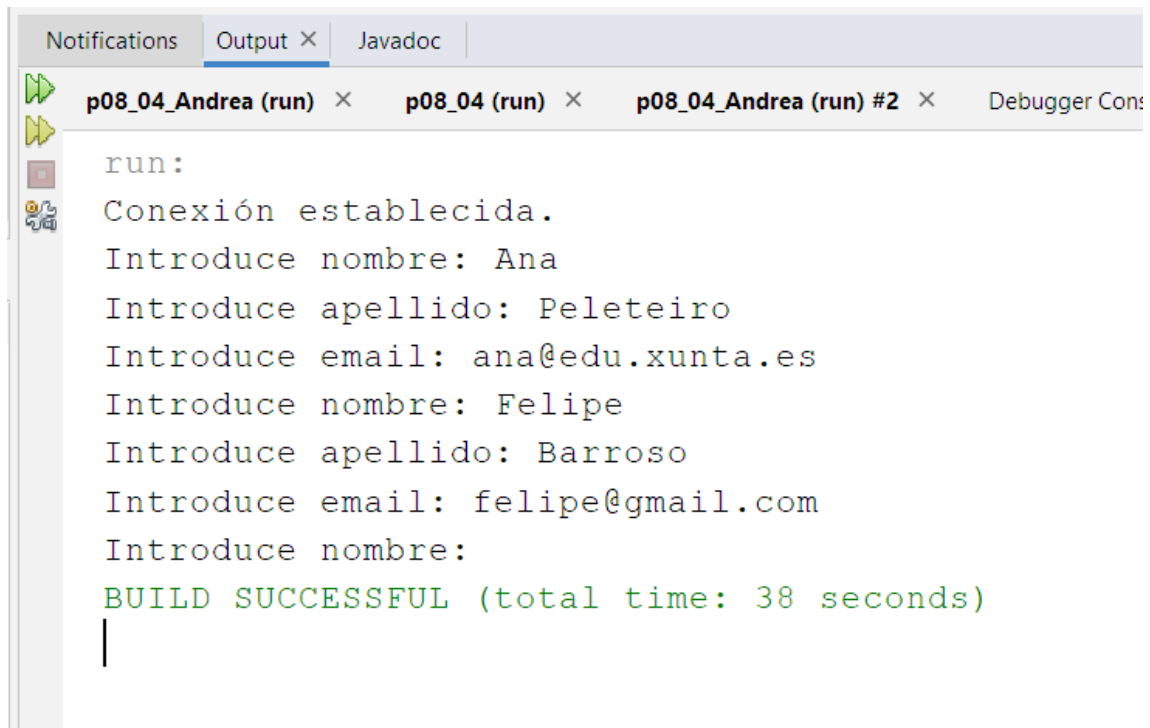
```

        stmt.executeUpdate(
            "INSERT INTO prueba2 (nombre, apellido,email)"
            + "VALUES('" + nombre
            + "', '" + apellido
            + "', '" + email
            + "')"
        );
    }
} catch (SQLException ex) {
    System.out.println("Error escribiendo en la BD. Abortando\n"
        + ex.getMessage());
    return;
}

```

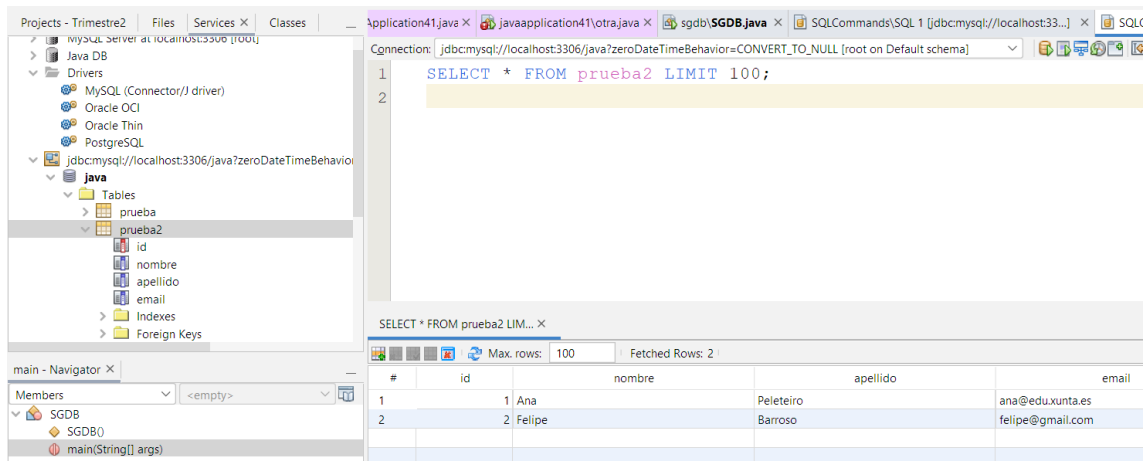
**IMPORTANTE:** fijarse en las comillas simples que preceden a cada uno de los campos. En SQL necesitamos entrecomillar los textos. Si los campos fueran numéricos no habría que ponerlas.

Ejecutamos el programa:



```
run:
Conexión establecida.
Introduce nombre: Ana
Introduce apellido: Peleteiro
Introduce email: ana@edu.xunta.es
Introduce nombre: Felipe
Introduce apellido: Barroso
Introduce email: felipe@gmail.com
Introduce nombre:
BUILD SUCCESSFUL (total time: 38 seconds)
```

Y comprobamos en la pestaña **Servicios** que los datos se han introducido correctamente:



Connection: jdbc:mysql://localhost:3306/java?zeroDateTimeBehavior=CONVERT\_TO\_NULL [root on Default schema]

```
1 SELECT * FROM prueba2 LIMIT 100;
2
```

SELECT \* FROM prueba2 LIM... X

#	id	nombre	apellido	email
1	1	Ana	Peleteiro	ana@edu.xunta.es
2	2	Felipe	Barroso	felipe@gmail.com

NOTA: no es necesario poner en mayúsculas las sentencias SQL. Las pongo así para distinguirlas más claramente.