

Asignatura

Programación



UNIVERSAE
Instituto Superior de FP

Asignatura

Programación

UNIDAD 14

Gestión de base de datos relacionales



UNIVERSAE
Instituto Superior de FP

Base de datos relacionales

Características

- Las más usadas.
- Los datos se representan en tablas y relaciones.
- Los campos se representan como columnas
- Los registros se representan como filas
- Existen claves primarias por cada tabla

Gestores de base de datos

- Microsoft SQL Server
- Oracle
- MariaDB
- MySQL
- PostgreSQL



ORACLE

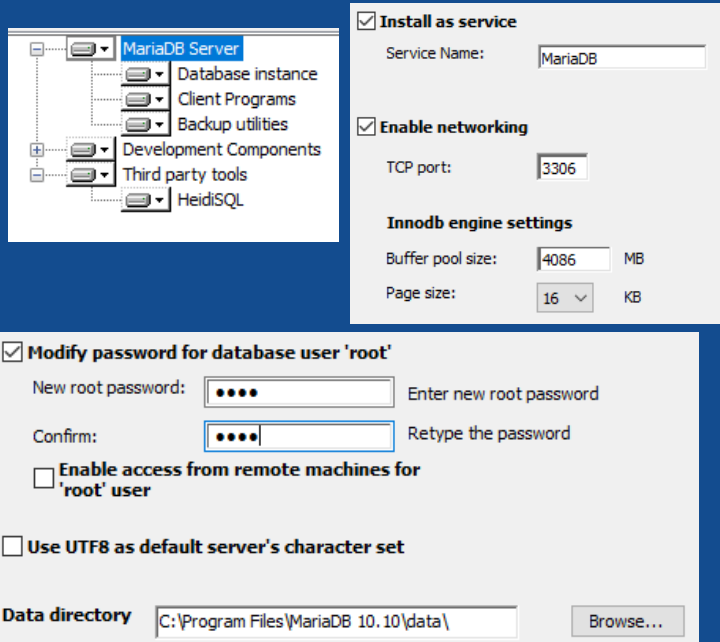




MariaDB. Crear la base de datos

Instalación

- Descargar el paquete instalador
<https://mariadb.org/download>
- Seguir el asistente de instalación
 - Especificar que se va a instalar (Server, Instancia, Client, HeidiSQL, etc.)
 - Indicar contraseña para el usuario root
 - Especificar la ubicación
 - Definir el nombre del servicio y el puerto



The screenshot shows the MariaDB Server installation wizard. On the left, a tree view shows the installation components: MariaDB Server, Database instance, Client Programs, Backup utilities, Development Components, and Third party tools. The 'MariaDB Server' component is selected. On the right, the 'Install as service' section is active, showing the 'Service Name' as 'MariaDB'. The 'Enable networking' checkbox is checked, and the 'TCP port' is set to '3306'. The 'InnoDB engine settings' section shows 'Buffer pool size' as '4086 MB' and 'Page size' as '16 KB'. At the bottom, the 'Data directory' is set to 'C:\Program Files\MariaDB 10.10\data\'. The 'Modify password for database user 'root'' section is also visible, with fields for 'New root password' and 'Confirm'.

Crear la base de datos

- Usar MySQL Client o HeidiSQL

```
C:\Program Files\MariaDB 10.10\bin>mysql -u root -p
Enter password: ****
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 6
Server version: 10.10.2-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

```
CREATE DATABASE agenda;
USE agenda;

CREATE TABLE Contactos (
  id          INT AUTO_INCREMENT PRIMARY KEY,
  nombre      VARCHAR(255) NOT NULL,
  apellido1   VARCHAR(255),
  apellido2   VARCHAR(255),
  telefono    INT,
  email       VARCHAR(255)
);

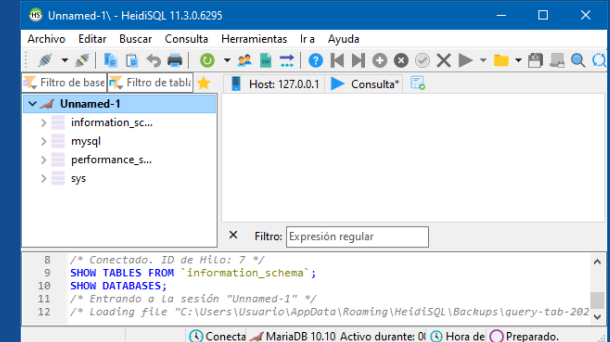
CREATE TABLE tareas (
  id          INT AUTO_INCREMENT PRIMARY KEY,
  nombre      VARCHAR(255) NOT NULL,
  descripcion  VARCHAR(255)
);

CREATE TABLE historico (
  id          INT AUTO_INCREMENT PRIMARY KEY,
  id_contacto INT NOT NULL,
  id_tarea    INT NOT NULL,
  fecha      DATE,
  finalizada BOOLEAN NOT NULL DEFAULT FALSE,
  FOREIGN KEY (id_contacto) REFERENCES contactos (id),
  FOREIGN KEY (id_tarea) REFERENCES tareas (id)
);
```

```
INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email)
VALUES ('Antonio','Gonzalez','Garcia','700800900','ant@email.com');
INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email)
VALUES ('Laura','Miró','Carreño','600600600','laura@email.com');
INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email)
VALUES ('Carolina','Miras','Miras','666677685','caro@email.com');
INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email)
VALUES ('Raúl','Miranda','Gomez','808606606','ramigo@email.com');

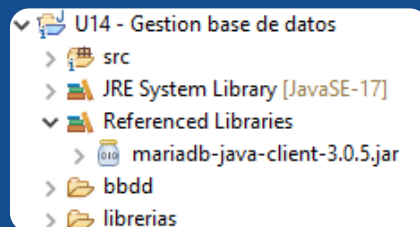
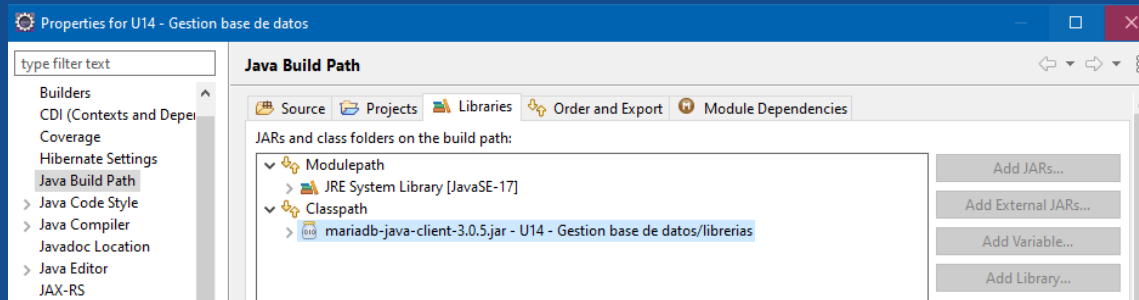
INSERT INTO tareas (nombre,descripcion)
VALUES ('Enviar email','Tarea de envio de emails');
INSERT INTO tareas (nombre,descripcion)
VALUES ('Recepcion correo postal','Tarea de recepción de correo postal');
INSERT INTO tareas (nombre,descripcion)
VALUES ('Comprobar calendario','Tarea de revisar calendario');
INSERT INTO tareas (nombre,descripcion)
VALUES ('Añadir evento','Tarea de añadir un evento');

INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (1,1,'2022/06/02',true);
INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (1,2,'2022/06/03',false);
INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (1,1,'2022/06/05',false);
INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (2,2,'2022/06/05',false);
INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (3,3,'2022/06/06',true);
INSERT INTO historico (id_contacto,id_tarea,fecha,finalizada) VALUES (3,4,'2022/06/06',true);
```



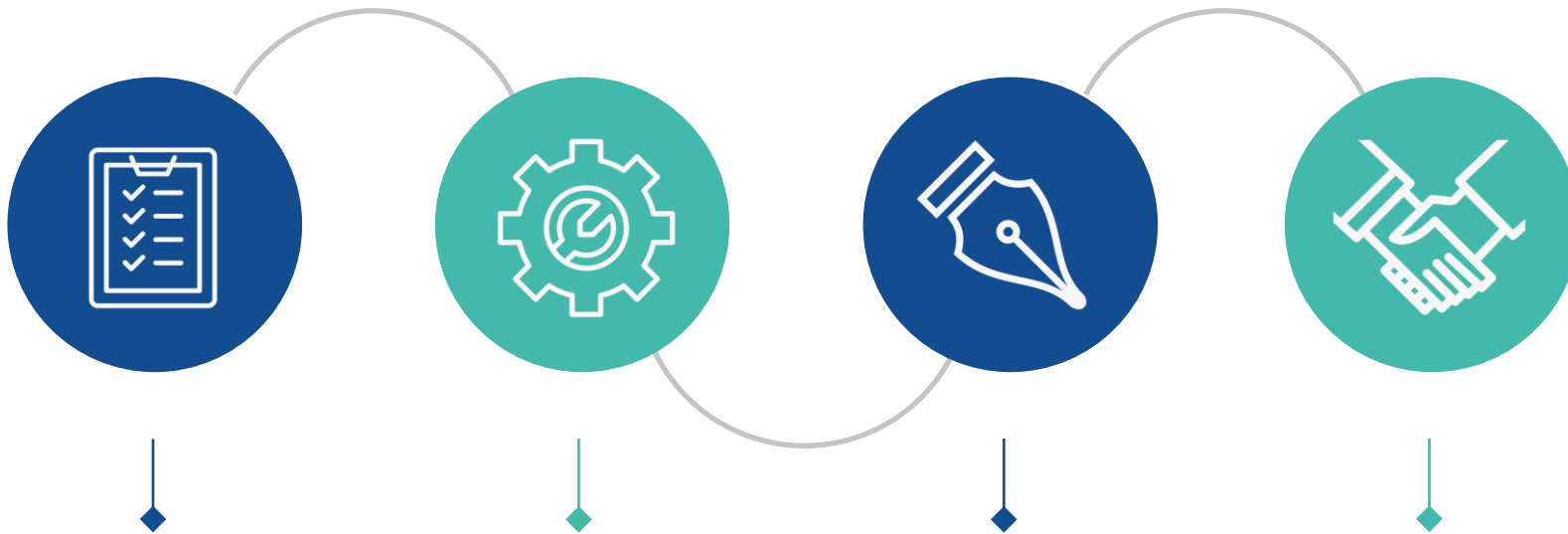
Crear el proyecto

- Descargar librería mariadb-java-client-3.0.5.jar
<https://mariadb.com/downloads/connectors/connectors-data-access/java8-connector>
- Crear un nuevo proyecto
- Añadir la librería al proyecto
 - Botón secundario sobre el proyecto/properties
 - Ubicarnos en Java Build Path y añadir JARs externos
- Debe de aparecer la librería referenciada



Procedimiento API JDBC

- Uso del paquete *java.sql*



Apertura

Abrir la conexión con la base de datos.

- `Class.forName(DRIVER);`
- `getConnection(url, user, password)`

Preparar consulta

Crear sentencia SQL

- `createStatement()`
- `prepareStatement()`

Obtener resultados

La consulta da como resultado un conjunto de filas o el número de filas afectadas

- `execute()`
- `executeQuery()`
- `executeUpdate()`

Cierre

Cerrar la conexión y otros elementos con la base de datos

- `close()`

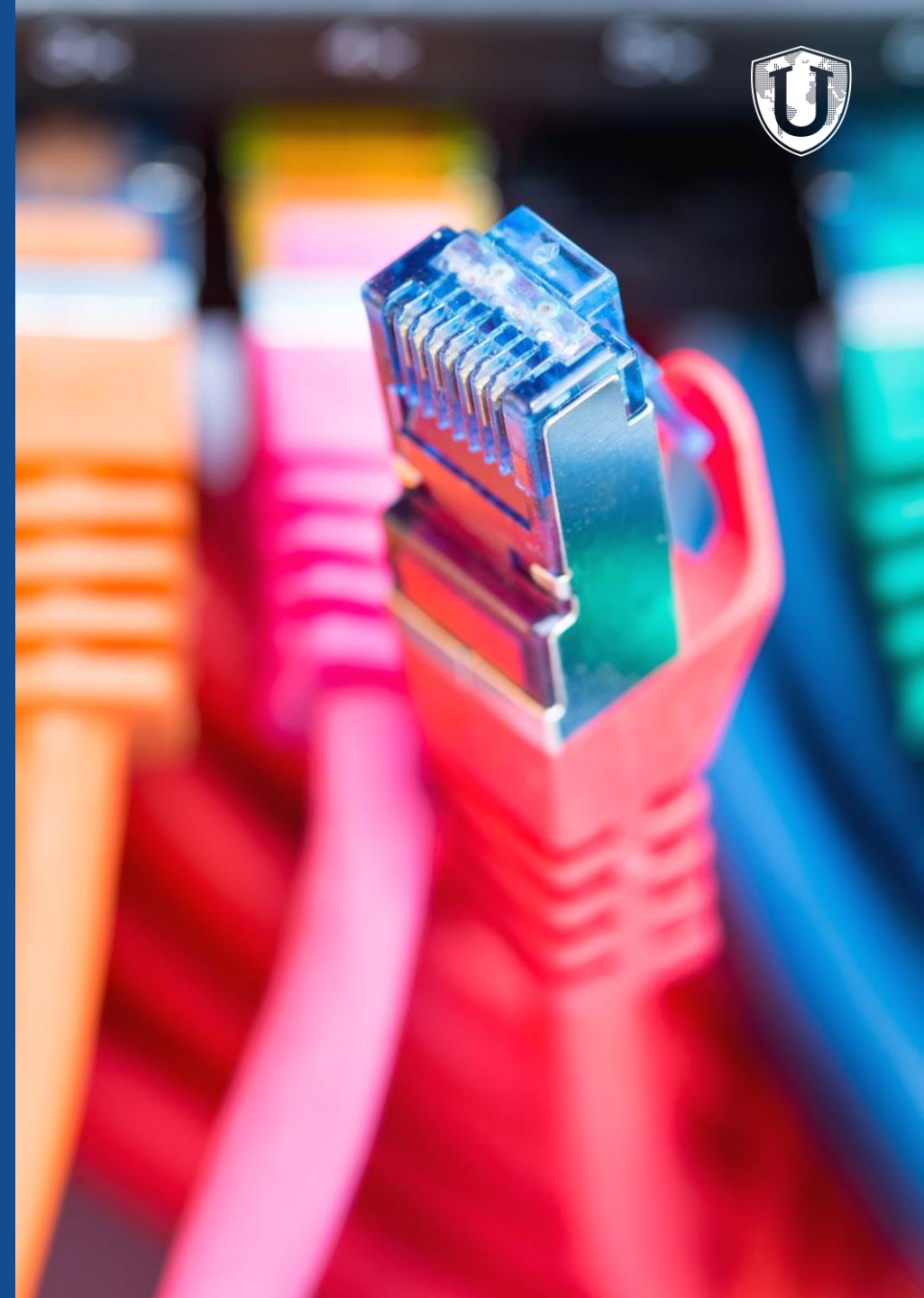


Establecer la conexión

- Uso del paquete `java.sql`
- Procedimiento
 1. Registrar el driver
 2. Montar la cadena de conexión
 3. Crear la conexión con `DriverManager`
 4. Realizar las acciones con la base de datos
 5. Cerrar la conexión

```
try {  
    Class.forName("org.mariadb.jdbc.Driver");  
    String url = "jdbc:mariadb://localhost:3306/agenda";  
    String usuario = "root";  
    String contraseña = "1234";  
    Connection conexion = DriverManager.getConnection (url, usuario, contraseña);  
  
    System.out.println("Conexión establecida");  
    DatabaseMetaData metadatos = conexion.getMetaData();  
    System.out.println("Nombre de la base de datos: " + metadatos.getDatabaseProductName());  
    System.out.println("Version del driver: " + metadatos.getDriverVersion());  
    System.out.println("Usuario conectado: " + metadatos.getUserName());  
    System.out.println("Soporta transacciones: " + metadatos.supportsTransactions());  
  
    conexion.close();  
}  
catch (Exception e) {  
    System.err.print(e.getCause());  
    System.err.print(e.getMessage());  
    e.printStackTrace();  
}
```

```
Conexión establecida  
Nombre de la base de datos: MariaDB  
Version del driver: 3.0.5  
Usuario conectado: root  
Soporta transacciones: true
```



Consultas



Statement

```
Connection conexion = null;
Statement declaracion = null;

try {
    // Conexión
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    // Consulta
    String query = "SELECT * FROM contactos";
    declaracion = conexion.createStatement();

    // Obtener resultados
    ResultSet resultados = declaracion.executeQuery(query);

    System.out.println("id" + ", " + "nombre" + ", " + "apellido1" + ", " +
        "apellido2" + ", " + "telefono" + ", " + "email");
    while (resultados.next()) {
        int id = resultados.getInt("id");
        String nombre = resultados.getString("nombre");
        String apellido1 = resultados.getString("apellido1");
        String apellido2 = resultados.getString("apellido2");
        int telefono = resultados.getInt("telefono");
        String email = resultados.getString("email");
        System.out.println(id + ", " + nombre + ", " + apellido1 + ", " +
            apellido2 + ", " + telefono + ", " + email);
    }
} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```

PreparedStatement

```
Connection conexion = null;
PreparedStatement declaracion = null;
try {
    // Conexión
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    // Consulta
    String query = "SELECT * FROM contactos WHERE id=?";
    declaracion = conexion.prepareStatement(query);
    declaracion.setInt(1, 2);

    // Obtener resultados
    ResultSet resultados = declaracion.executeQuery(query);
    System.out.println("id" + ", " + "nombre" + ", " + "apellido1" + ", " +
        "apellido2" + ", " + "telefono" + ", " + "email");
    while (resultados.next()) {
        int id = resultados.getInt("id");
        String nombre = resultados.getString("nombre");
        String apellido1 = resultados.getString("apellido1");
        String apellido2 = resultados.getString("apellido2");
        int telefono = resultados.getInt("telefono");
        String email = resultados.getString("email");
        System.out.println(id + ", " + nombre + ", " + apellido1 + ", " +
            apellido2 + ", " + telefono + ", " + email);
    }
} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```


Insertar, Actualizar y Borrar



Insert

```
Connection conexion = null;
PreparedStatement declaracion = null;
try {
    // Conexion
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    // Sentencia insert
    String query = "INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email) "
        + "VALUES (?, ?, ?, ?, ?)";
    declaracion = conexion.prepareStatement(query);

    declaracion.setString(1, "Pascual");
    declaracion.setString(2, "Lozano");
    declaracion.setString(3, "García");
    declaracion.setInt(4, 615754236);
    declaracion.setString(5, "pascual@email.com");

    // Ejecutar
    declaracion.executeUpdate();

} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```

Update

```
Connection conexion = null;
Statement declaracion = null;
try {
    // Conexion
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    // Sentencia update
    String query = "UPDATE contactos SET email='pas.lo@email.com' WHERE id=3";
    declaracion = conexion.createStatement();

    // Ejecutar
    int registrosActualizados = declaracion.executeUpdate(query);
    System.out.println(registrosActualizados);

} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```

Delete

```
Connection conexion = null;
Statement declaracion = null;
try {
    // Conexion
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    // Sentencia delete
    String query = "DELETE FROM contactos WHERE id=5";
    declaracion = conexion.createStatement();

    // Ejecutar
    int registrosActualizados = declaracion.executeUpdate(query);
    System.out.println(registrosActualizados);
} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```

Transacciones

```
Connection conexion = null;
Statement declaracion = null;

try {
    // Conexión
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection(url, usuario, contraseña);

    //Establecer el control de las transacciones
    conexion.setAutoCommit(false);

    // Sentencias
    declaracion = conexion.createStatement();

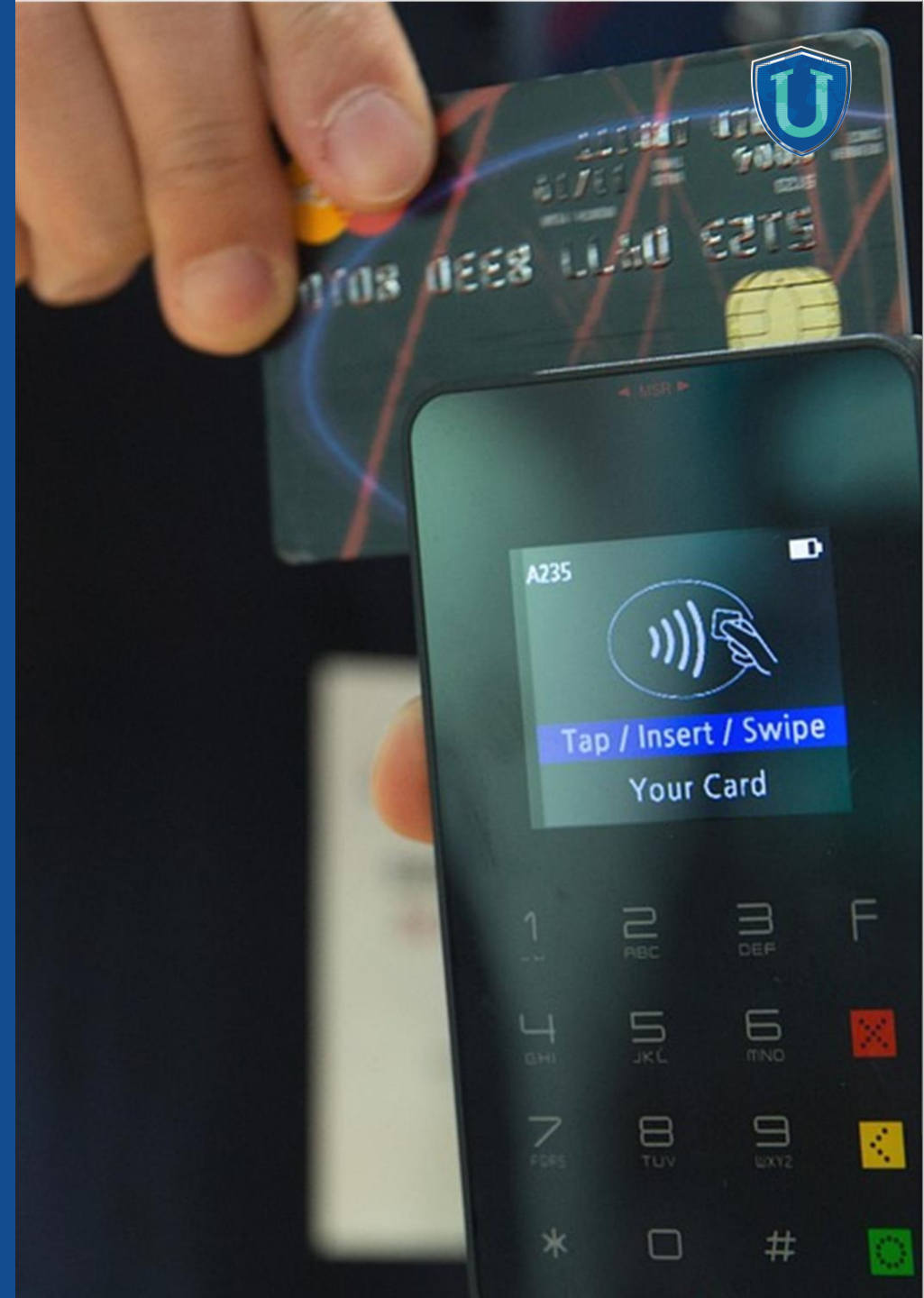
    String query1 = "INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email) "
        + "VALUES ('Amparo','Carrion','Gonzalez','656112233','amparo@email.com')";

    String query2 = "INSERT INTO HISTORICO (id_contacto,id_tarea,fecha,finalizada) "
        + "VALUES ("
        + "      (SELECT id FROM contactos WHERE nombre='Amparo' and apellido1='carrion'),"
        + "      1,'2022/06/05',false)";

    // Ejecutar sentencias
    // executeUpdate devuelve el numero de registros afectados
    int resultado = declaracion.executeUpdate(query1);
    if(resultado != 0)
        resultado = declaracion.executeUpdate(query2);

    if(resultado == 0)
        conexion.rollback();
    else
        conexion.commit();

} catch (SQLException | ClassNotFoundException e) {
    e.printStackTrace();
    try { if(conexion != null) conexion.rollback(); } catch (SQLException e1) { e1.printStackTrace(); }
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```



Ejecución en bloque



```
Connection conexion = null;
Statement declaracion = null;
try {
    // Conexión
    Class.forName("org.mariadb.jdbc.Driver");
    String url = "jdbc:mariadb://localhost:3306/agenda";
    String usuario = "root";
    String contraseña = "1234";
    conexion = DriverManager.getConnection (url, usuario, contraseña);
    conexion.setAutoCommit(false);

    // Sentencias
    String query1 = "INSERT INTO CONTACTOS (nombre,apellido1,apellido2,telefono,email) "
        + "VALUES ('Amparo','Carrion','Gonzalez','656112233','email@email.com)";
    String query2 = "UPDATE contactos SET email='email@email.com' WHERE id=4";

    declaracion = conexion.createStatement();

    declaracion.addBatch(query1);
    declaracion.addBatch(query2);

    // Ejecución
    int[] registrosAfectados = declaracion.executeBatch();
    System.out.println(registrosAfectados);

    if(registrosAfectados.length != 0)
        conexion.commit();
    else
        conexion.rollback();

} catch (SQLException | ClassNotFoundException e) {
    System.err.print(e.getMessage());
    try { if(conexion!= null) conexion.rollback(); } catch (SQLException e1) { e1.printStackTrace();}
} finally {
    // Cierre
    try {
        if(declaracion != null) declaracion.close();
        if(conexion != null) conexion.close();
    } catch (SQLException e) { e.printStackTrace(); }
}
```



Metainformación

DatabaseMetaData

- Obtiene información de la base de datos
- Diferentes métodos para obtener información.
- Usuario, bases de datos, tablas, etc.

Uso de sentencias de SQL

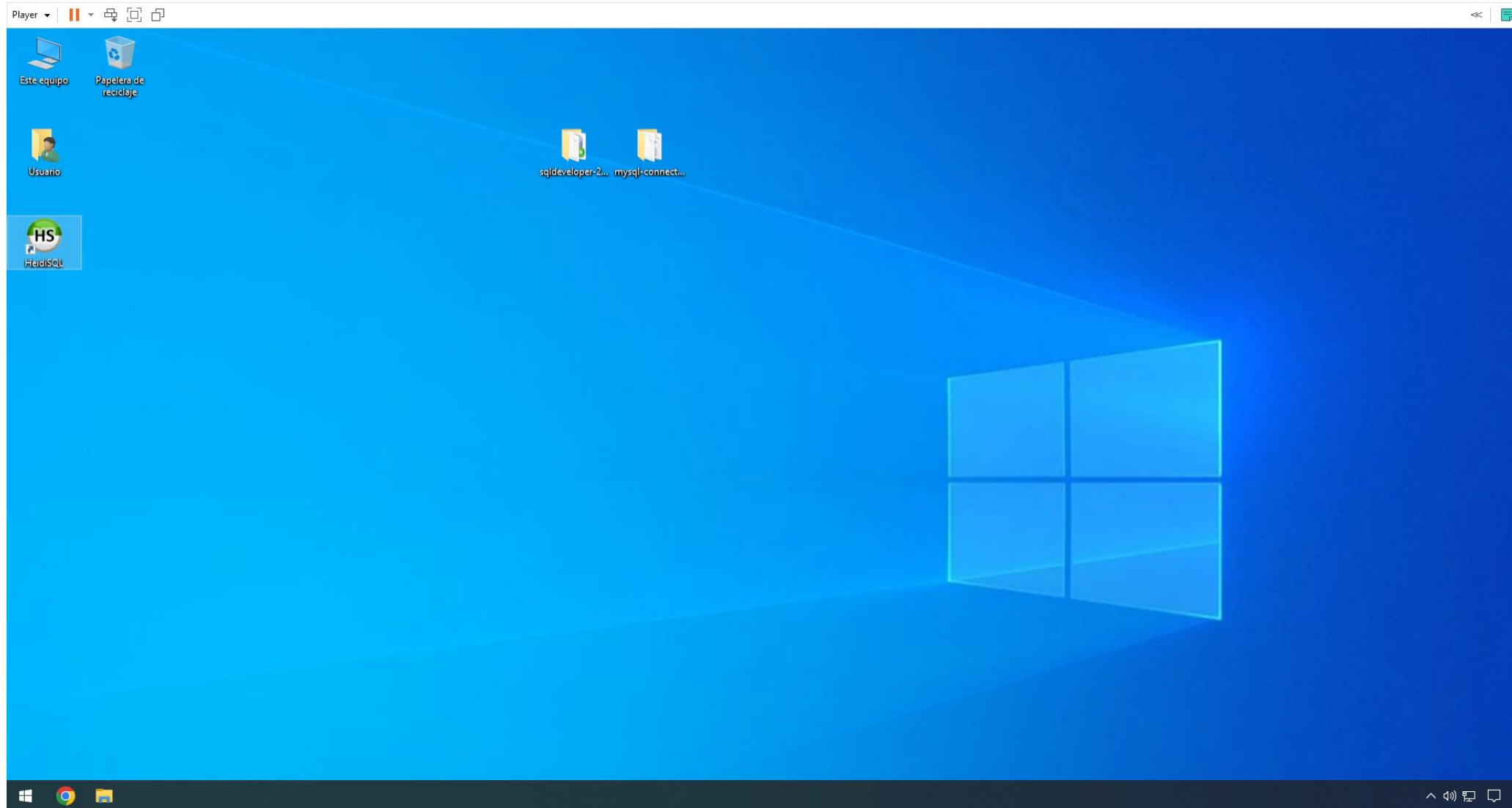
- SHOW DATABASES;
- SHOW TABLES;
- SELECT * FROM mysql.user
- SELECT table_name
FROM INFORMATION_SCHEMA.TABLES
WHERE table_schema = 'agenda';
- SELECT *
FROM INFORMATION_SCHEMA.CHARACTER_SETS;

```
try {  
    // Conexion  
    Class.forName("org.mariadb.jdbc.Driver");  
    String url = "jdbc:mariadb://localhost:3306/agenda";  
    String usuario = "root";  
    String contraseña = "1234";  
    Connection conexion = DriverManager.getConnection (url, usuario, contraseña);  
  
    // Uso de DatabaseMetaData  
    DatabaseMetaData metadatos = conexion.getMetaData();  
    ResultSet resultados = metadatos.getCatalogs();  
  
    System.out.println("- Obtener esquemas o base de datos:");  
    while (resultados.next()) {  
        String basededatos = resultados.getString("TABLE_CAT");  
        System.out.println(basededatos);  
    }  
  
    // Obtener metadatos directamente  
    Statement declaracion = conexion.createStatement();  
    String query = "SELECT DISTINCT user FROM mysql.user";  
    ResultSet res = declaracion.executeQuery(query);  
  
    System.out.println("- Obtener usuarios de la base de datos:");  
    while (res.next()) {  
        String user = res.getString("USER");  
        System.out.println(user);  
    }  
  
    declaracion.close();  
    conexion.close();  
} catch (SQLException | ClassNotFoundException e) {  
    System.err.print(e.getMessage());  
}
```

```
- Obtener esquemas o base de datos:  
agenda  
information_schema  
mysql  
performance_schema  
sys  
- Obtener usuarios de la base de datos:  
root  
mariadb.sys
```

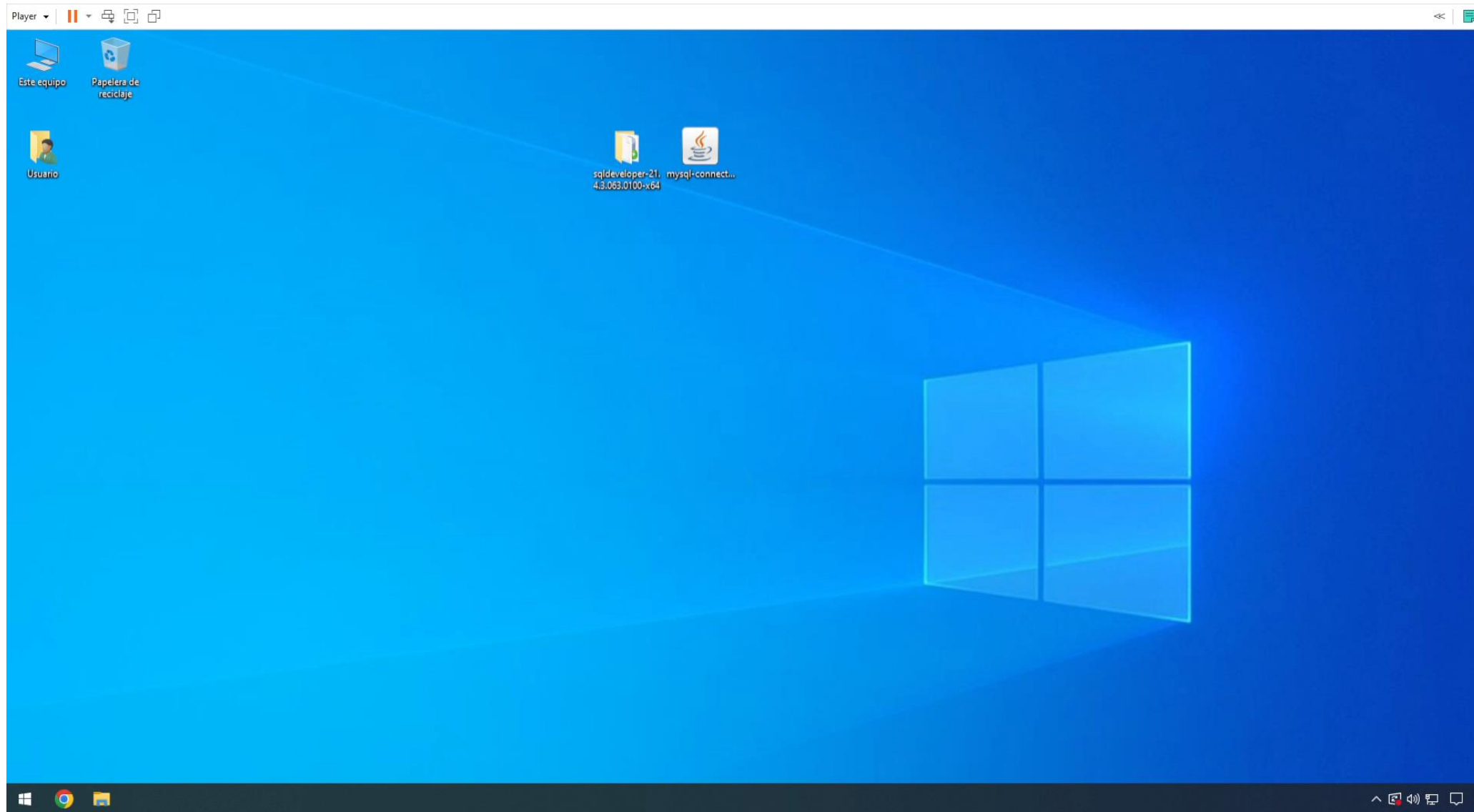



Cientes de bases de datos. Heidi SQL





Cientes de bases de datos. SQL Developer





Resumen

1. Base de datos relacionales
2. MariaDB. Crear la base de datos
3. Crear el proyecto
4. Procedimiento API JDBC
5. Establecer la conexión
6. Consultas
7. Insertar, actualizar y borrar
8. Transacciones
9. Ejecución en bloque
10. Metainformación
11. Clientes de base de datos. HeidiSQL
12. Clientes de base de datos. SQL Developer

The background is a solid blue color. Overlaid on this are several faint, light-blue geometric patterns. These include a grid of small squares that form larger, irregular shapes, and numerous small, light-blue arrows pointing in various directions. The overall effect is a sense of movement and digital connectivity.

UNIVERSAE

— CHANGE YOUR WAY —