TP 2 – PARTE A - Acceso a Datos JSON y Migración de la Información

1- Cree la tabla **Pais** en una base de datos tipo **SQL (MySQL, SQLServer, etc)** con los siguientes campos:

codigoPais PK numero entero
nombrePais varchar(50) not null
capitalPais varchar(50) not null
región varchar(50) not null
población entero largo not null
latitud numero decimal not null
longitud numero decimal not null

```sql
CREATE DATABASE tp2_db;
USE tp2_db;

CREATE TABLE Pais (
    codigoPais INT PRIMARY KEY,
    nombrePais VARCHAR(50) NOT NULL,
    capitalPais VARCHAR(50) NOT NULL,
    region VARCHAR(50) NOT NULL,
    poblacion BIGINT NOT NULL,
    latitud FLOAT NOT NULL,
    longitud FLOAT NOT NULL
);
```

2- Ejecute una llamada mediante al menos 2 lenguajes de programación (Java, PHP, ASP, JavaScript, Python, etc) a la siguiente URL tipo RESTful
https://restcountries.com/v2/callingcode/{callingcode}

Ejemplo
https://restcountries.com/v2/callingcode/54

RETORNA JSON:

[{"name":"Argentina", "topLevelDomain":[".ar"], "alpha2Code":"AR", "alpha3Code":"ARG", "callingCodes":["54"], "capital":"Buenos Aires", "altSpellings":["AR", "Argentine Republic", "República Argentina"], "region":"Americas", "subregion":"South America", "population":43590400, "latlng":[-34.0,-64.0], "demonym":"Argentinean", "area":2780400.0, "gini":44.5, "timezones":["UTC-03:00"], "borders":["BOL", "BRA", "CHL", "PRY", "URY"], "nativeName":"Argentina", "numericCode":"032", "currencies":[{"code":"ARS", "name":"Argentine peso", "symbol":"$"}], "languages":[{"iso639_1":"es", "iso639_2":"spa", "name":"Spanish", "nativeName":"Español"},{"iso639_1":"gn", "iso639_2":"grn", "name":"Guaraní", "nativeName":"Avañe'ẽ"}], "translations":{"de":"Argentinien", "es":"Argentina", "fr":"Argentine", "ja":"アルゼンチン", "it":"Argentina", "br":"Argentina", "pt":"Argentina", "nl":"Argentinië", "hr":"Argentina", "fa":"نیتنا ژرآ"},

"flag":"https://restcountries.eu/data/arg.svg", "regionalBlocs":[{"acronym":"USAN", "name":"Union of South American Nations", "otherAcronyms":["UNASUR", "UNASUL", "UZAN"], "otherNames":["Unión de Naciones Suramericanas", "União de Nações Sul-Americanas", "Unie van Zuid-Amerikaanse Naties", "South American Union"]}], "cioc":"ARG"}]

Obtenga la información y migre la misma a la tabla país creada anteriormente. El proceso debe ejecutarse para los códigos desde 1 hasta 300, contemplando que si alguno de los códigos no retorna datos se continúe con el siguiente.

El pseudocódigo a realizar seria el siguiente:

For(código = 1; código <= 300; código++){

DATOS JSON = https://restcountries.com/v2/callingcode/{callingcode}
if(hayDatos) {
nombrePais = DATOS JSON [name]
capitalPais = DATOS JSON [capital]
region = DATOS JSON [region]
poblacion = DATOS JSON [population]
latitud = DATOS JSON [latlng [0]]
longitud = DATOS JSON [latlng [1]]
codigoPais = DATOS JSON [codigoPais]
//busco país en base de datos filtrando por código
País = SELECT * FROM PAIS WHERE codigoPais = codigoPais
If(existePais){
//ejecuto un update a la tabla país
UPDATE país SET nombrePais =..... WHERE codigoPais = codigoPais
}**else**{
//ejecuto un insert a la tabla país
INSERT INTO país(campos) VALUES (valores)
}
}**else**{
continue
}
}

```
String url1 = "jdbc:mysql://localhost:3306/tp2_db ";
String username = "root";
String password = "44907252”
try {
    Connection conn = DriverManager.getConnection(url1,username,password);
    for(int codigo = 1 ; codigo <=300; codigo++) {
        URL url = new URL("https://restcountries.com/v2/callingcode/" + codigo);
        HttpURLConnection conectar = (HttpURLConnection) url.openConnection();
        conectar.setRequestMethod("GET");
        conectar.connect();
        int responseCode = conectar.getResponseCode();
        if (responseCode == HttpURLConnection.HTTP_OK) {
            StringBuilder information = new StringBuilder();
            Scanner scanner = new Scanner(url.openStream());
            while (scanner.hasNext()) {
                information.append(scanner.nextLine());
            }
            scanner.close();
            JSONArray jsonArray = new JSONArray(information.toString());
            JSONObject jsonObject = jsonArray.getJSONObject(0);
            String capitalPais;
            if (jsonObject.has("capital"))
                capitalPais = jsonObject.getString("capital");
            else
```

```
                capitalPais = "";
            String nombrePais = jsonObject.getString("name");
            String region = jsonObject.getString("region");
            int poblacion = jsonObject.getInt("population");
            double latitud = jsonObject.getJSONArray("latlng").getDouble(0);
            double longitud = jsonObject.getJSONArray("latlng").getDouble(1);
            PreparedStatement stmt = conn.prepareStatement("SELECT * FROM Pais WHERE
codigoPais =
                                ?");
                        stmt.setInt(1, codigo);
            ResultSet rs = stmt.executeQuery();
            if (rs.next()) {
                PreparedStatement updateStmt = conn.prepareStatement("UPDATE Pais SET
                nombrePais = ?, capitalPais = ?, region = ?, poblacion = ?, latitud = ?,
longitud = ? WHERE codigoPais = ?");
                updateStmt.setString(1, nombrePais);
                updateStmt.setString(2, capitalPais);
                updateStmt.setString(3, region);
                updateStmt.setInt(4, poblacion);
                updateStmt.setDouble(5, latitud);
                updateStmt.setDouble(6, longitud);
                updateStmt.setInt(7, codigo);
                updateStmt.executeUpdate();
            } else {
                PreparedStatement insertStmt = conn.prepareStatement("INSERT INTO Pais
                (codigoPais, nombrePais, capitalPais, region, poblacion, latitud,
longitud) VALUES (?, ?, ?, ?, ?, ?, ?)");
                insertStmt.setInt(1, codigo);
                insertStmt.setString(2, nombrePais);
                insertStmt.setString(3, capitalPais);
                insertStmt.setString(4, region);
                insertStmt.setInt(5, poblacion);
                insertStmt.setDouble(6, latitud);
                insertStmt.setDouble(7, longitud);
                insertStmt.executeUpdate();

            }
        }else{
            continue;
        }
    }
} catch (ProtocolException ex) {
    throw new RuntimeException(ex);
} catch (MalformedURLException ex) {
    throw new RuntimeException(ex);
} catch (SQLException ex) {
    throw new RuntimeException(ex);
} catch (IOException ex) {
    throw new RuntimeException(ex);
}
System.out.println("Finalizado");
```

TP 2 – PARTE B - Acceso a Datos JSON / MongoDB

1- Instale Mongo DB en su PC

2- Cree una base de datos mongo llamada **paises_db**.

3- Crear una colección **paises** para almacenar los documentos con los datos: codigoPais(callingCodes), nombrePais(name), capitalPais(capital), región(region), población(population), latitud(latlng), longitud(latlng), superficie(area)

4- Ejecute una llamada mediante el lenguaje de programación que prefiera a la siguiente URL tipo RESTful

https://restcountries.com/v2/callingcode/{callingcode}

Ejemplo

https://restcountries.com/v2/callingcode/**54**

RETORNA JSON:

[{"name":"Argentina", "topLevelDomain":[".ar"], "alpha2Code":"AR", "alpha3Code":"ARG", "callingCodes":["54"], "capital":"Buenos Aires", "altSpellings":["AR", "Argentine Republic", "República Argentina"], "region":"Americas", "subregion":"South America", "population":43590400, "latlng":[-34.0,-64.0], "demonym":"Argentinean", "area":2780400.0, "gini":44.5, "timezones":["UTC-03:00"], "borders":["BOL", "BRA", "CHL", "PRY", "URY"], "nativeName":"Argentina", "numericCode":"032", "currencies":[{"code":"ARS", "name":"Argentine peso", "symbol":"$"}], "languages":[{"iso639_1":"es", "iso639_2":"spa", "name":"Spanish", "nativeName":"Español"},{"iso639_1":"gn", "iso639_2":"grn", "name":"Guaraní", "nativeName":"Avañe'ẽ"}], "translations":{"de":"Argentinien", "es":"Argentina", "fr":"Argentine", "ja":"アルゼンチン", "it":"Argentina", "br":"Argentina", "pt":"Argentina", "nl":"Argentinië", "hr":"Argentina", "fa":"نیتنا ژرآ"}, "flag":"https://restcountries.eu/data/arg.svg", "regionalBlocs":[{"acronym":"USAN", "name":"Union of South American Nations", "otherAcronyms":["UNASUR", "UNASUL", "UZAN"], "otherNames":["Unión de Naciones Suramericanas", "União de Nações Sul-Americanas", "Unie van Zuid-Amerikaanse Naties", "South American Union"]}], "cioc":"ARG"}]

Obtenga la información y migre la misma a la colección paises creada anteriormente. El proceso debe ejecutarse para los códigos desde 1 hasta 300, contemplando que si alguno de los códigos no retorna datos se continúe con el siguiente.

<u>El pseudocódigo a realizar seria el siguiente:</u>

```
For(código = 1; código <= 300; código++){

    DATOS JSON = https://restcountries.com/v2/callingcode/{code}
    if(hayDatos) {
        nombrePais =  DATOS JSON [name]
        capitalPais =  DATOS JSON [capital]
        region = DATOS JSON [region]
        poblacion = DATOS JSON [population]
        latitud = DATOS JSON [latlng [0]]
        longitud = DATOS JSON [latlng [1]]
        codigoPais = DATOS JSON [codigoPais]
        //busco país en base de datos filtrando por código
        db.paises.find({ ..... });
        If(existePais){
            //ejecuto un update
            db.paises.update({ ..... });
        }else{
            //ejecuto un insert
            db.paises.insert({.....});
    }else{
        continue
    }
}
```

```
    String url1 = "jdbc:mysql://localhost:3306/países_db";
    String username = "root";
    String password = "44907252";
    try {
        Connection conn = DriverManager.getConnection(url1,username,password);
        MongoClient mongoClient = crearConexion();
        MongoDatabase database = mongoClient.getDatabase("paises_db");
        MongoCollection<Document> paises = database.getCollection("pises");
        for(int codigo = 1 ; codigo <=300; codigo++) {
            URL url = new URL("https://restcountries.com/v2/callingcode/" + codigo);
            HttpURLConnection conectar = (HttpURLConnection) url.openConnection();
            conectar.setRequestMethod("GET");
            conectar.connect();
            int responseCode = conectar.getResponseCode();
            if (responseCode == HttpURLConnection.HTTP_OK) {
                StringBuilder information = new StringBuilder();
                Scanner scanner = new Scanner(url.openStream());
                while (scanner.hasNext()) {
                    information.append(scanner.nextLine());
                }
                scanner.close();
                JSONArray jsonArray = new JSONArray(information.toString());
                JSONObject jsonObject = jsonArray.getJSONObject(0);
                String capitalPais;
                if (jsonObject.has("capital"))
                    capitalPais = jsonObject.getString("capital");
                else
                    capitalPais = "";
                String nombrePais = jsonObject.getString("name");
                String region = jsonObject.getString("region");
                int poblacion = jsonObject.getInt("population");
                double latitud = jsonObject.getJSONArray("latlng").getDouble(0);
                double longitud = jsonObject.getJSONArray("latlng").getDouble(1);
                Document query = new Document("codigoPais", codigo);
                Document existePais = paises.find(query).first();
                if (existePais != null) {
                    Document updatePaises= new Document("$set", new Document()
                            .append("nombrePais", nombrePais)
                            .append("capitalPais", capitalPais)
                            .append("región", region)
                            .append("población", poblacion)
                            .append("latitud", latitud)
                            .append("longitud", longitud));
                    paises.updateOne(query, updatePaises);
                } else {
                    Document nuevoPais= new Document("codigoPais", codigo)
                            .append("nombrePais", nombrePais)
                            .append("capitalPais", capitalPais)
                            .append("región", region)
                            .append("población", poblacion)
                            .append("latitud", latitud)
                            .append("longitud", longitud);
                    paises.insertOne(nuevoPais);
                }
            }
        }
        mongoClient.close();
    }catch (ProtocolException e) {
        throw new RuntimeException(e);
    } catch (MalformedURLException e) {
```

```
            throw new RuntimeException(e);
        } catch (SQLException e) {
            throw new RuntimeException(e);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
    private static MongoClient crearConexion() {
        MongoClient mongo = null;
        try {
            mongo = new MongoClient("localhost", 27017);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return mongo;
    }
```

5- Finalizada la migración de los datos a la colección de Países, codifique mediante el lenguaje de programación elegido y haciendo uso de las consultas de Mongo los siguientes métodos.

5.1. Codifique un método que **seleccione** los documentos de la colección **países** donde la **región** sea **Americas**. Muestre el resultado por pantalla o consola.

```
public static void paisesRegion(MongoCollection<Document> collection, String region) {
    Document query = new Document("región", region);
    FindIterable<Document> result = collection.find(query);
    MongoCursor<Document> cursor = result.iterator();
    while (cursor.hasNext()) {
        Document document = cursor.next();
        System.out.println(document.toJson());
    }
}
```

5.2. Codifique un método que **seleccione** los documentos de la colección **países** donde la **región** sea **Americas** y la **población** sea mayor a **100000000**. Muestre el resultado por pantalla o consola.

```
public static void paisesRegionPoblacion(MongoCollection<Document> collection, String
region,
                                          long poblacion) {
    Document query = new Document("región", region)
            .append("población", new Document("$gt", poblacion));
    FindIterable<Document> result = collection.find(query);
    MongoCursor<Document> cursor = result.iterator();
    while (cursor.hasNext()) {
        Document document = cursor.next();
        System.out.println(document.toJson());
    }
}
```

5.3. Codifique un método que **seleccione** los documentos de la colección **países** donde la **región** sea **distinto** de **Africa**. (investigue **$ne**). Muestre el resultado por pantalla o consola.

```
public static void regionPoblacion(MongoCollection<Document> collection, String region) {
    Document query = new Document("región", new Document("$ne", region));
    FindIterable<Document> result = collection.find(query);
    MongoCursor<Document> cursor = result.iterator();
    while (cursor.hasNext()) {
        Document document = cursor.next();
        System.out.println(document.toJson());
    }
}
```

5.4. Codifique un método que **actualice** el documento de la colección **países** donde el name sea Egypt,

cambiando el name a “Egipto” y la población a 95000000.

```
public static void updateNombre (MongoCollection<Document> collection, String nombre,
String nuevoNom, long pobla) {
    Document query = new Document("name", nombre);
    Document updateFields = new Document("$set", new Document()
            .append("name", nuevoNom)
            .append("population", pobla));
    UpdateResult result = collection.updateOne(query, updateFields);
}
```

5.5. Codifique un método que **elimine** el documento de la colección **países** donde el **código** del país sea **258.**

```
public static void deletePais(MongoCollection<Document> collection, String codigo) {
    Document query = new Document("codigoPais", codigo);
    DeleteResult result = collection.deleteOne(query);
}
```

5.6. Describa que sucede al ejecutar el método **drop()** sobre una colección y sobre una base de datos.

5.7. Codifique un método que seleccione los documentos de la colección **países** cuya **población** sea mayor a 50000000 y menor a 150000000. Muestre el resultado por pantalla o consola.

```
public static void rangoPoblacion(MongoCollection<Document> collection, long poblaMin,
long
        poblaMax) {
    Document query = new Document("población", new
            Document("$gt",poblaMin).append("$lt", poblaMax));
    FindIterable<Document> result = collection.find(query);
    MongoCursor<Document> cursor = result.iterator();
    while (cursor.hasNext()) {
        Document document = cursor.next();
        System.out.println(document.toJson());
    }
}
```

5.8. Codifique un método que seleccione los documentos de la colección **países** ordenados por nombre (name) en forma Ascendente. **sort().** Muestre el resultado por pantalla o consola.

```
public static void ordenarNombre(MongoCollection<Document> collection) {
    FindIterable<Document> result = collection.find().sort(Sorts.ascending("name"));
    MongoCursor<Document> cursor = result.iterator();
    while (cursor.hasNext()) {
        Document document = cursor.next();
        System.out.println(document.toJson());
    }
}
```

5.9. Describa que sucede al ejecutar el método **skip()** sobre una colección. Ejemplifique con la colección **países**.

5.10. Describa y ejemplifique como el uso de expresiones regulares en Mongo puede reemplazar el uso de la cláusula **LIKE** de SQL.

5.11. Cree un nuevo índice para la colección países asignando el campo código como índice. investigue **createIndex**())

5.12. Describa como se realiza un backup de la base de datos mongo **países_db**.