Evaluación Automatizada de la Asignatura Análisis y Diseño de Sistemas de Información

PROYECTO

presentado para optar

al Título de Grado en Ingeniería en Organización Industrial por

Sebastian Ricci Zaragoza

bajo la supervisión de

Nicolás Serrano

Donostia-San Sebastián, agosto de 2022

Tecnun
Universidad de Navarra
ESCUELA DE INGENIERÍA
INGENIARITZA ESKOLA
SCHOOL OF ENGINEERING

Tecnun Universidad de Navarra | ESCUELA DE INGENIERÍA INGENIARITZA ESKOLA SCHOOL OF ENGINEERING

Proyecto Fin de Grado

INGENIERIA EN ORGANIZACIÓN INDUSTRIAL

Evaluación Automatizada de la Asignatura Análisis y Diseño de Sistemas de Información

Sebastian Ricci Zaragoza

Donostia-San Sebastián, agosto de 2022

Pº Manuel Lardizabal, 13. 20018 Donostia-San Sebastián, Gipuzkoa
Tel. 943 219 877 · Fax 943 311 442 · www.tecnun.es

AGRADECIMIENTOS

En primer lugar, quiero agradecer a mi supervisor Nicolás Serrano por darme la oportunidad de trabajar en este proyecto, por guiarme y asesorarme en todo el proceso. También quiero agradecer a un compañero de alumno interno Daniel Adames por el trabajo previo que realizamos para este proyecto.

En una nota más personal, me gustaría agradecer a mi familia por su apoyo incondicional durante estos 4 años, a pesar de la distancia siempre estaremos juntos. Quiero agradecer a todos los profesores que he tenido, tanto en la universidad como fuera de ella, por todas las lecciones para formarme profesionalmente y más importantemente, como persona.

ÍNDICE DE CONTENIDOS

AGRADECIMIENTOS ..... **I**

ÍNDICE DE FIGURAS ..... **III**

ÍNDICE DE TABLAS ..... **IV**

ABREVIACIONES ..... **V**

RESUMEN ..... **VI**

1. INTRODUCCIÓN **1**

1.1 OBJETIVO 1
1.2 DEFINICIÓN DEL PROYECTO 1
1.3 FUNCIONALIDAD DEL SISTEMA 1

2. ESTADO DEL ARTE **3**

2.1 CODEX 3
2.2 ACCESS 3
2.3 JSON 4

3. DISEÑO Y DESARROLLO DEL SISTEMA **5**

3.1 METODOLOGÍA SEGUIDA 5
3.2 REQUERIMIENTOS 5
3.3 ESTRUCTURA DE DESCOMPOSICIÓN DEL TRABAJO 6
3.4 INTERFAZ GRÁFICA 7
3.5 RECOPILACIÓN DE DATOS 8
3.5.1 Tablas *9*
3.5.2 Consultas *11*
3.5.3 Formularios *13*
3.5.4 Informes *15*
3.5.5 Relaciones *17*
3.6 CONEXIÓN CON CODEX 19

4. RESULTADOS OBTENIDOS **21**

5. CONCLUSIONES Y RECOMENDACIONES **23**

6. REFERENCIAS **25**

7. ANEXO **27**

7.1 PRÁCTICA 1: ANÁLISIS Y DISEÑO DE BASES DE DATOS – INTRODUCCIÓN A TABLAS, RELACIONES, CONSULTAS, FORMULARIOS Y INFORMES 27

ÍNDICE DE FIGURAS

FIGURA 1: CONEXIÓN ENTRE ACCESS Y CODEX 2
FIGURA 2: EJEMPLO DE OBJETO EN FORMATO JSON 4
FIGURA 3: METODOLOGÍA DE DESARROLLO ÁGIL 5
FIGURA 4: INTERFAZ GRÁFICA DEL PROYECTO 7
FIGURA 5: DIAGRAMA DE INTERFAZ GRÁFICA 7
FIGURA 6: CÓDIGO DE RECOPILACIÓN DE TABLAS 9
FIGURA 7: DIAGRAMA DE RECOPILACIÓN DE TABLAS A JSON 10
FIGURA 8: CÓDIGO DE RECOPILACIÓN DE CONSULTAS 11
FIGURA 9: DIAGRAMA DE RECOPILACIÓN DE CONSULTAS A JSON 12
FIGURA 10: CÓDIGO DE RECOPILACIÓN DE FORMULARIOS 13
FIGURA 11:DIAGRAMA DE RECOPILACIÓN DE FORMULARIOS A JSON 14
FIGURA 12: CÓDIGO DE RECOPILACIÓN DE INFORMES 15
FIGURA 13: DIAGRAMA DE RECOPILACIÓN DE INFORMES A JSON 16
FIGURA 14: CÓDIGO DE RECOPILACIÓN DE RELACIONES 17
FIGURA 15: DIAGRAMA DE RECOPILACIÓN DE RELACIONES A JSON 18
FIGURA 16: CÓDIGO DE ENVIÓ A CODEX 19
FIGURA 17: DIAGRAMA DE COMUNICACIÓN ACCESS-CODEX 20

ÍNDICE DE TABLAS

TABLA 1: TABLA DE ABREVIACIONES V

ABREVIACIONES

Tabla 1: Tabla de abreviaciones

Abreviación	Significado
JSON	JavaScript Object Notation
ADSI	Análisis y Diseño de Sistemas de Información
VBA	Visual Basic for Applications
POO	Programación Orientada a Objetos
SQL	Structured Query Language
ROI	Return On Investment

RESUMEN

Este proyecto se basa en desarrollar una evaluación automatizada de la asignatura de Análisis y diseño de sistemas de información (ADSI) e integrarla con Codex. Codex es una herramienta online de aprendizaje implementada en Tecnun, utilizada por estudiantes y profesores para el desarrollo de actividades y pruebas evaluadas.

La evaluación consta en recopilar la información dentro de una base de datos (tablas, consultas, informes, formularios y relaciones) usando el lenguaje de programación VBA desde la herramienta Access y enviarlo al servidor de Codex mediante un string JSON donde se puntúa la respuesta y se devuelve la calificación al estudiante directamente.

El sistema consta de una interfaz gráfica en la cual el estudiante puede conectarse a Codex a través de sus credenciales y contestar los enunciados planteados por el profesor desde Codex. Las reglas de evaluación para cada ejercicio son planteadas desde Codex de tal forma que el JSON que se envía desde Access a Codex puede ser evaluado correctamente.

1. INTRODUCCIÓN

Se presenta un proyecto de evaluación automatizada para la asignatura ADSI. Este proyecto fue planteado para mejorar la experiencia tanto del alumnado como del profesor en la evaluación de la asignatura. Al integrar Codex con la asignatura los alumnos tienen la opción de autoevaluarse mediante prácticas y actividades preparadas por el profesor, de esta forma el alumnado puede prepararse mejor para la asignatura sin tener que ejercer una mayor carga de evaluación para el profesor. Para lograr esto se plantea el desarrollo de una herramienta desde Access que pueda establecer una conexión con Codex capaz de enviar y recibir información sin tener que usar el navegador.

1.1 Objetivo

Desarrollar e implementar una herramienta desde Access que sea capaz de asegurar una evaluación correcta de partes de la asignatura. La herramienta por desarrollar debe de ser intuitiva y sencilla tanto para el alumnado como el profesor para una evaluación satisfactoria mediante Codex, por lo cual una interfaz gráfica será necesaria.

Es fundamental que la herramienta sea capaz de recopilar y enviar correctamente toda la información pertinente de la base de datos, en este caso siendo las tablas, consultas, informes, formularios y relaciones entre tablas creadas por el usuario.

Por último, la herramienta debe de ser capaz de notificar al estudiante del resultado obtenido con una calificación y comentarios relacionados a la respuesta del estudiante.

1.2 Definición del proyecto

La solución que se plantea es desarrollar una interfaz gráfica mediante un formulario en Access capaz de visualizar el enunciado del ejercicio planteado y finalmente recopilar y enviar información a Codex mediante un string JSON.

1.3 Funcionalidad del sistema

El sistema constará de una interfaz gráfica usando la opción de diseño de formularios de Access en la cual tendrá cajas de texto para introducir las credenciales del alumno y para visualizar el enunciado de Codex. Se utilizarán botones para navegar los diferentes enunciados y para enviar el resultado a Codex. Los estudiantes no tendrán que interactuar con el código de recopilación de datos en ningún momento y si se quisiera ver el código este estará protegido mediante una clave que solo el profesor podrá acceder. En la figura 1 se puede observar cómo Access interactúa con Codex mediante peticiones al servidor para obtener las preguntas almacenadas dentro de Codex.

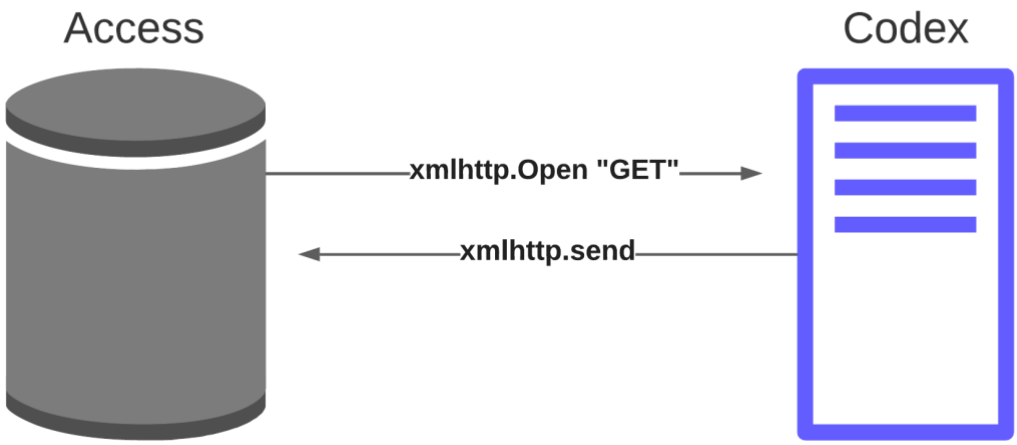


Figura 1: Conexión entre Access y Codex

2. ESTADO DEL ARTE

2.1 Codex

Codex es una herramienta online desarrollada por el Departamento de Ingeniería en Organización Industrial de la Universidad de Navarra – Tecnun. Su principal función es facilitar la enseñanza tanto para el alumnado y profesores ofreciendo recursos de aprendizaje como pruebas de evaluación, prácticas y otro contenido a medida para cada asignatura individual.

Codex permite que los profesores puedan trasladar parte de la asignatura a una plataforma digital en la que los estudiantes pueden practicar y ser evaluados a través de esta. Resumido, Codex es una plataforma de aprendizaje en línea en la cual los profesores pueden crear y editar contenido en línea para evaluar el aprendizaje de los estudiantes.

2.2 Access

Microsoft Access es un sistema de gestión de bases de datos ofrecido por Microsoft. Las funciones principales de Access son almacenar y administrar datos de tal forma que sea intuitivo analizar grandes cantidades de información. A continuación, se describen brevemente los elementos principales de la aplicación [1]:

- Tablas: Las tablas son objetos que guardan información en formato de columna y fila. Usualmente están interrelacionadas con otras tablas mediante llaves primarias y llaves extranjeras.

- Consultas: Las consultas responden a una pregunta hecha por el usuario seleccionando, ordenando y filtrando datos según los criterios de búsqueda. Las consultas extraen la información de una o más tablas. Los tipos de consulta son de selección, actualizar, insertar y eliminar datos.

- Formularios: Los formularios son objetos principalmente utilizados para generar interfaces de usuario para interactuar con la base de datos. Ayudan a mostrar datos en vivo de las tablas y su principal función está en la entrada y edición de datos.

- Informes: Los informes son objetos principalmente utilizados para calcular, imprimir y resumir datos seleccionados mediante un informe personalizado.

- Relaciones: Las relaciones son objetos que establecen una conexión entre tablas, así creando una base de datos relacional. Son capaz de interrelacionar las tablas con la ayuda de llaves primarias y extranjeras.

2.3 JSON

JavaScript Object Notation o JSON, es un formato para él envió de información basado en un subgrupo del lenguaje de programación JavaScript. Los JSON están estructurados principalmente por colecciones de pares de nombre/valor y listas ordenadas de valores [2]. JSON tiene la ventaja de tener una estructura universal soportada por mayoría de los lenguajes de programación.

Los objetos JSON pueden contener distintos tipos de datos desde pares de nombre/valor y arrays hasta otros objetos dentro del mismo, así consiguiendo objetos en forma de cascada. A continuación, en la figura 2 se puede observar un ejemplo de la información y los tipos de datos que se puede guardar usando este formato.

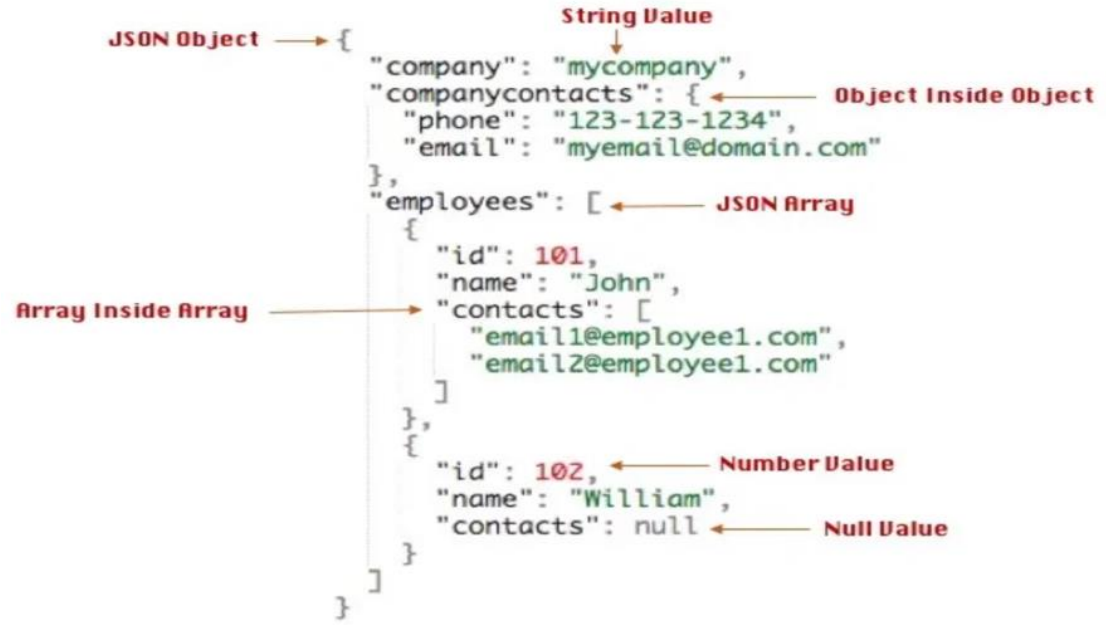


Figura 2: Ejemplo de objeto en formato JSON

Para poder enviar objetos complejos como lo pueden ser las tablas, consultas, formularios, informes y relaciones de Access, el formato JSON será el utilizado para él envió de datos hacia Codex. De esta forma se puede transformar toda la información contenida dentro de la base de datos a un simple string.

3. DISEÑO Y DESARROLLO DEL SISTEMA

3.1 Metodología seguida

Para el desarrollo del proyecto se utilizó un modelo de desarrollo ágil. Usando este modelo se minimiza el riesgo de errores en el código y requisitos cambiantes para el programa al agregar nuevas funciones. Se fue desarrollando el programa por iteraciones que contienen incrementos pequeños para agregar nuevas funcionalidades al sistema y se recolectaba feedback de parte del profesor de la asignatura. A continuación, en la figura 3 se puede observar el flujo de trabajo usando la metodología de desarrollo ágil.

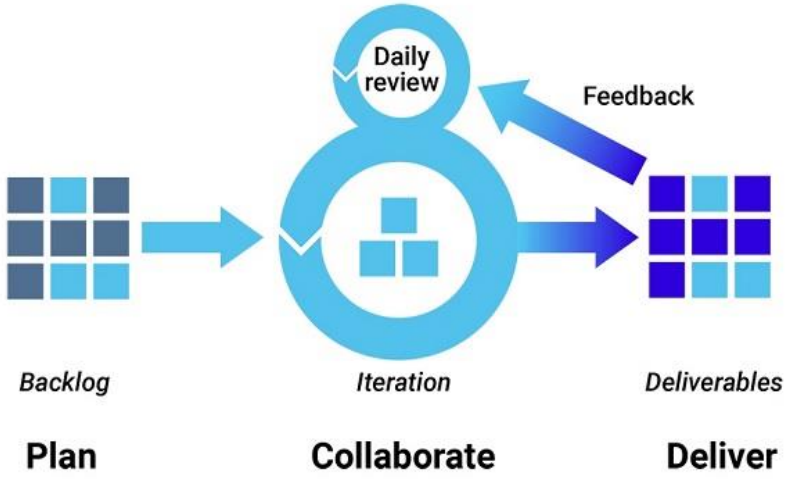


Figura 3: Metodología de desarrollo ágil

3.2 Requerimientos

Se plantearon los siguientes requerimientos con el profesor de ADSI para el sistema de evaluación automatizada:

- Debe de ser capaz de evaluar tablas, consultas, formularios, informes y relaciones dentro de la base de datos.
- Debe de tener una interfaz de usuario sencilla e intuitiva para el alumnado.
- La interfaz de usuario debe de mostrar los enunciados planteados desde Codex sin necesidad de un navegador externo.
- Debe de ser capaz de enviar la información recopilada a través de un JSON hacia Codex.
- Establecer reglas de evaluación para evaluar el objeto JSON obtenido desde Codex.
- El alumno debe de poder observar el resultado obtenido desde la interfaz junto a los comentarios escritos por el profesor en el caso de que la respuesta sea incorrecta o correcta.

3.3 Estructura de descomposición del trabajo

El proyecto se centra en cuatro áreas principales de trabajo descritas a continuación:

1. Requerimientos: Establecer los requerimientos necesarios para el sistema de evaluación automatizada. Las acciones llevadas a cabo para lograr esto serán entrevistas con el profesor de ADSI tomando feedback iterativo de cada una de estas.

2. Interfaz gráfica: Desarrollar una interfaz gráfica intuitiva capaz de presentar los enunciados y calificación obtenida desde Codex hacia Access. Las acciones llevadas a cabo para lograr esto serán la implementación de formularios y eventos programados para los controles de este.

3. Recopilación de datos: Desarrollar el código necesario para recopilar los datos de los elementos principales de la base de datos y transformarlo a un objeto JSON. La acción llevada a cabo para lograr esto será la implementación de código VBA orientado a objetos.

4. Conexión con Codex: Desarrollar el código necesario para enviar el JSON formado en la recopilación de datos a Codex y recibir la calificación seguidamente en la interfaz gráfica. Las acciones llevadas a cabo para lograr esto serán la implementación de código VBA con objetos HTTP y comprobar la compatibilidad de funciones de evaluación de Codex con el objeto JSON a enviar.

3.4 Interfaz gráfica

Se diseño una interfaz gráfica usando la opción de diseño de formularios de Access y creando eventos a ejecutar cuando se presionan los botones. La interfaz gráfica cumple con los requerimientos anteriormente descritos, siendo capaz de establecer una conexión con Codex para tanto recibir como enviar información. El código será explicado en el siguiente apartado. A continuación, en la figura 4 se puede observar la interfaz gráfica utilizada en el proyecto.

Figura 4: Interfaz gráfica del proyecto

La interfaz gráfica puede ser implementada en cualquier practica o prueba evaluada simplemente copiando y pegando el objeto en cualquier otro archivo de Access. La única preparación previa que tiene la interfaz es por parte del profesor en la que tendrá que indicar que cuaderno y actividad de Codex debe evaluar desde el código VBA, el cual será protegido por una contraseña. A continuación, en la figura 5 se puede observar el diagrama de uso de la interfaz gráfica.

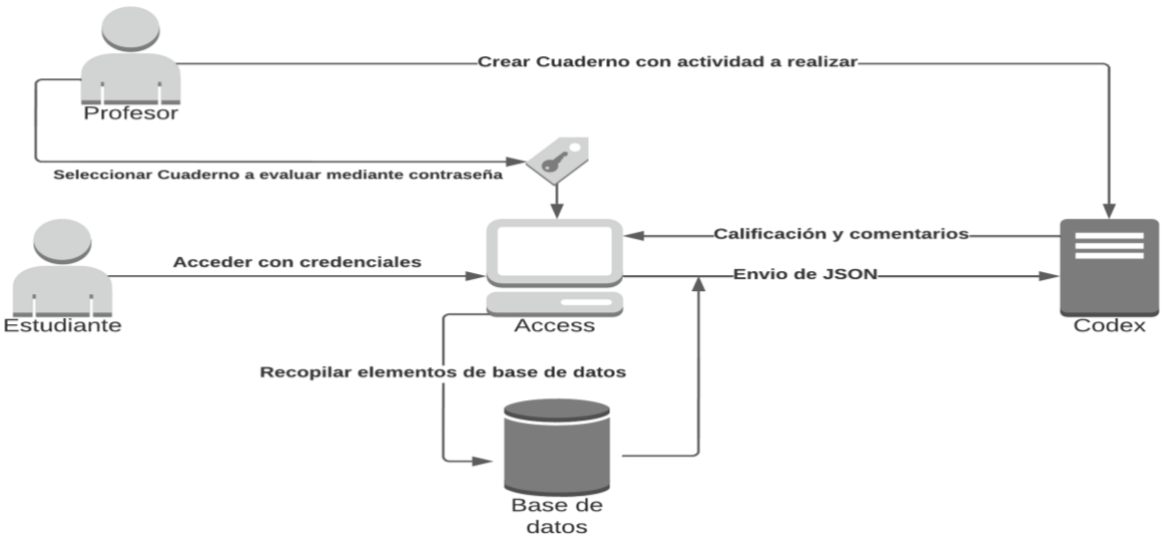


Figura 5: Diagrama de interfaz gráfica

3.5 Recopilación de datos

La recopilación de datos de los principales elementos de la base de datos se desarrolló con VBA siguiendo una programación orientada a objetos. La programación orientada a objetos (POO) es un estándar de programación basado en centrar la elaboración del software en las propiedades y métodos de objetos, en vez del uso de lógica y otras funciones [3]. Los objetos están formados por campos de datos que tienen comportamientos y atributos únicos a los que se puede acceder a través de métodos, en el caso de Access los objetos principales son las tablas, consultas, formularios, informes y relaciones.

Las ventajas principales de seguir un modelo de programación orientado a objetos son un mantenimiento, depuración del programa y diseño eficaz. Esto viene dado gracias a la modularidad que ofrece la POO, debido a esto se pueden identificar errores y agregar nuevas funcionalidades al código rápidamente ya que el código se descompone en métodos y propiedades de objetos ya establecidas.

El código desarrollado para la recopilación de los datos se basa principalmente en poder acceder a las propiedades de cada objeto dentro de la base de datos y guardar la información en un string formateado como JSON. Para acceder a la base de datos se utiliza el objeto *CurrentDB*, las tablas con *TableDefs*, consultas con *QueryDefs*, formularios con *Application*.*Forms*, informes con *Application*.*Reports* y las relaciones con *Relations*.

En los siguientes apartados se detalla cómo se recopila la información de cada objeto dentro de la base de datos con los métodos apropiados. Se ha obviado parte del código que no es de directo interés explicar para el proyecto, en caso de querer visualizar el código completo ver en el anexo del proyecto.

3.5.1 Tablas

```
1.   'Access to the current Database
2.  Set db = CurrentDb
3.
4.  'Look for the tables
5.  Dim Tables_count    As Integer
6.  For Each TableDefs In db.TableDefs
7.      If Not (TableDefs.Name Like "MSys*" Or TableDefs.Name Like "~*" Or
    TableDefs.Name Like "INFO") Then
8.          Tables_count = Tables_count + 1
9.      End If
10. Next TableDefs
11.
12. If Tables_count > 0 Then
13.     JSON = "'Tables':["
14.     For Each TableDefs In db.TableDefs
15.         'Only select the tables that we need from the database. Note that the
    informs are also sent so we also have to
16.         'specify we dont want the inform INFO that holds the log in information
17.         If Not (TableDefs.Name Like "MSys*" Or TableDefs.Name Like "~*" Or
    TableDefs.Name Like "INFO") Then
18.             JSON = JSON &        "'Tablename':'"
19.             JSON = JSON & TableDefs.Name &        "','fields':["
20.             'Get the table's properties
21.             For Each prp In TableDefs.Fields
22.                 JSON = JSON &        "'name':'" & prp.Name & "','type':'" &
    prp.Type & "'},"
23.             Next prp
24.             JSON = Left(JSON, Len(JSON) - 1)
25.             JSON = JSON & "]},"
26.         End If
27.     Next TableDefs
28.     JSON = Left(JSON, Len(JSON) - 1)
29.     JSON = JSON & "]"
30. Else
31.     JSON = ”'Tables':0"
32. End If
```

Figura 6: Código de recopilación de tablas

Se inicializa la variable *db* y su asignación es un objeto que engloba la base de datos actualmente utilizada. Esto permite acceder a los elementos dentro de la base de datos a través de sus propiedades y métodos. Para recorrer las tablas se inicializa la variable *TableDefs* la cual contiene toda la información relevante de las tablas.

Se procede a recorrer todas las tablas dentro de la base de datos con un bucle que obvia aquellas tablas que no son directamente creadas por el usuario, por esto se coloca la condición que excluye las tablas con nombres parecidos a “MSys”, “~” y “INFO”. Se realiza un contador *Tables_count* para contar el número de tablas creadas por el usuario y esta variable será utilizada en la condición *if* para construir el JSON dependiendo de su valor. En el caso de que *Tables_count* sea mayor que cero se procederá a construir el JSON tomando el nombre de la tabla y las propiedades de sus campos, en este caso siendo el nombre del campo y su tipo de dato. En el caso de que *Tables_count* no sea mayor que cero el JSON se inicializara con tablas igual a cero.

A continuación, en la figura 7 se puede observar un esquema de que información se está recopilando de las tablas y como el código estructura el JSON.

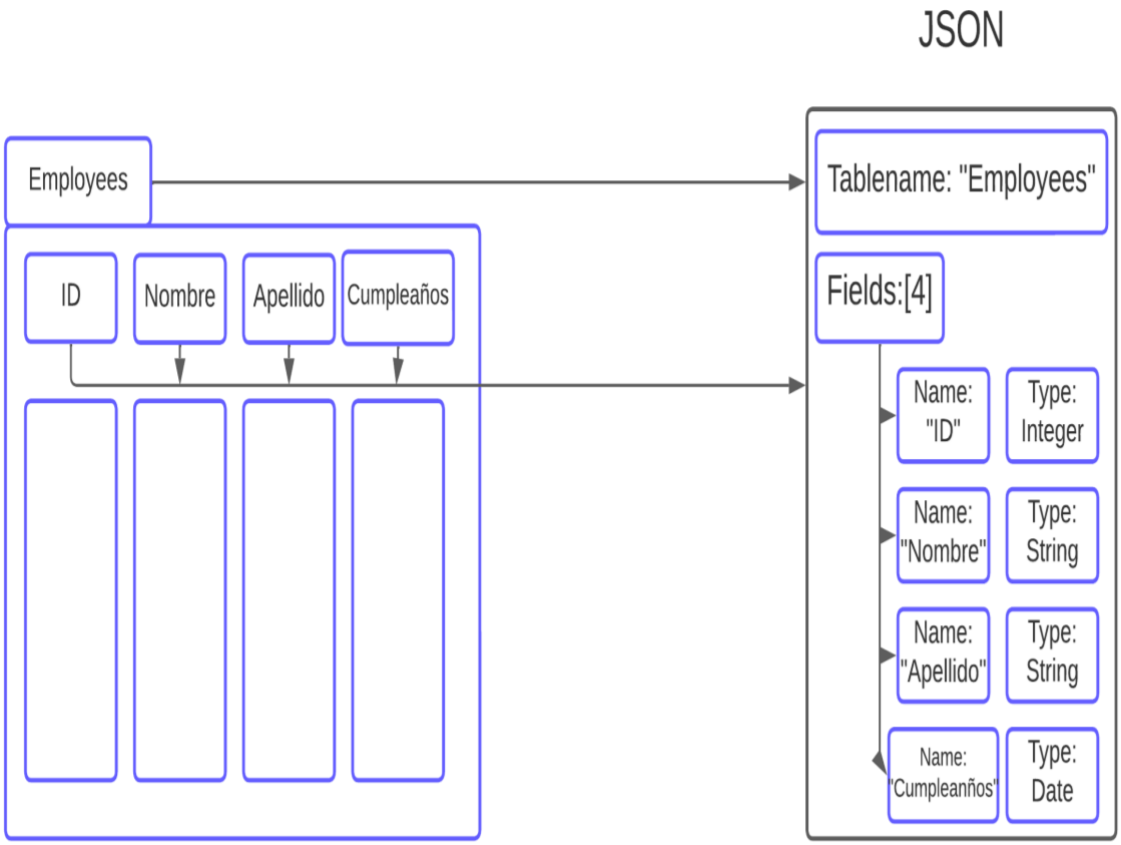


Figura 7: Diagrama de recopilación de Tablas a JSON

3.5.2 Consultas

```
1.  'Look for the Queries
2.  Dim qdfLoop         As QueryDef
3.  Dim query_count     As Integer
4.
5.  For Each qdfLoop In db.QueryDefs
6.      If Not (qdfLoop.Name Like "~sq_*") Then
7.          query_count = query_count + 1
8.      End If
9.  Next qdfLoop
10.
11. If query_count > 0 Then
12.     JSON = JSON & “'Queries':["
13.     Dim prpLoop     As Property
14.     Dim sql         As String
15.     countquery = 0
16.
17.     For Each qdfLoop In db.QueryDefs
18.         If Not (qdfLoop.Name Like "~sq_*") Then
19.             countquery = countquery + 1
20.             sql = qdfLoop.sql
21.             posicionsql = InStr(1, sql, "FROM")
22.             sql_crear = Left(sql, posicionsql - 1) & "INTO QueryTable" &
    countquery & Right(sql, Len(sql) - posicionsql + 2)
23.             db.Execute sql_crear
24.             db.TableDefs.Refresh
25.             For Each TableDefs In db.TableDefs
26.                 If TableDefs.Name Like "QueryTable" & countquery Then
27.                     JSON = JSON & “'Queryname':'"
28.                     JSON = JSON & qdfLoop.Name & “','fields':["
29.                     'Get the table's properties
30.                     For Each prp In TableDefs.Fields
31.                         JSON = JSON & “'name':'" & prp.Name & "','type':'" &
    prp.Type & "'},"
32.                     Next prp
33.                     JSON = Left(JSON, Len(JSON) - 1)
34.                     JSON = JSON & "]},"
35.                     'Delete newly created QueryTable
36.                     DoCmd.DeleteObject acTable, "QueryTable" & countquery
37.
38.                 End If
39.             Next TableDefs
40.         End If
41.     Next qdfLoop
42.     JSON = Left(JSON, Len(JSON) - 1)
43.     JSON = JSON & "]"
44. Else
45.     JSON = JSON & “'Queries':0"
46. End If
```

Figura 8: Código de recopilación de consultas

Para recorrer las consultas se inicializa la variable *qdfLoop* la cual contiene toda la información relevante de las consultas. Se procede a recorrer todas las consultas dentro de la base de datos con un bucle que obvia aquellas consultas que no son directamente creadas por el usuario, por esto se coloca la condición que excluye las consultas con nombres parecidos a "~sq_".

Se realiza un contador *query_count* para contar el número de consultas creadas por el usuario y esta variable será utilizada en la condición *if* para construir el JSON dependiendo de su valor. En el caso de que *query_count* sea mayor que cero se procederá a construir el JSON tomando el nombre de la consulta y las propiedades de la tabla que genera dicha consulta. Esto significa que las consultas se evalúan convirtiendo el SQL de la consulta en una tabla resultante y luego evaluar dicha tabla igual que en el apartado anterior. Las tablas generadas por la consulta serán eliminadas al finalizar esta sección del código para no alterar las tablas creadas por el usuario. En el caso de que *query_count* no sea mayor que cero el JSON se construirá con consultas igual a cero.

A continuación, en la figura 9 se puede observar un esquema de que información se está recopilando de las consultas y como se extrae la información de las tablas generadas del SQL de la consulta hacia el JSON, posteriormente eliminando las tablas creadas.

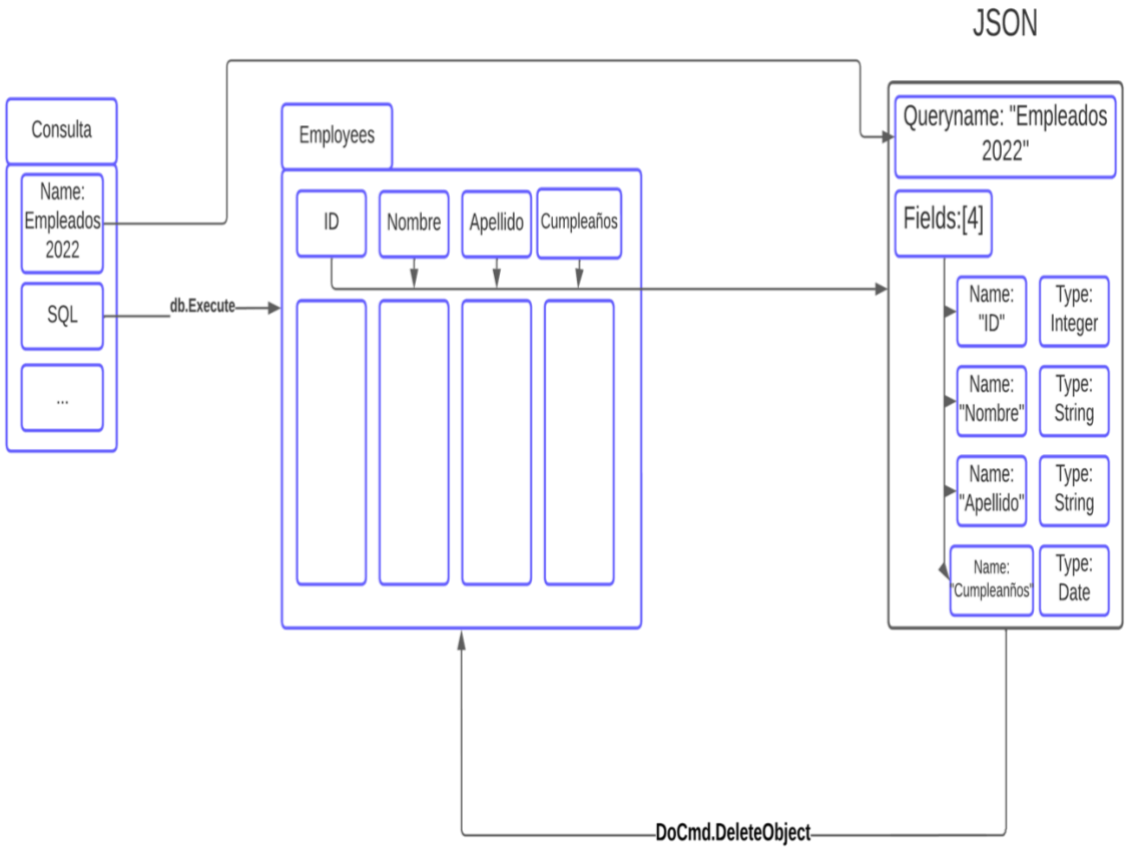


Figura 9: Diagrama de recopilación de Consultas a JSON

3.5.3 Formularios

```
1.   'Look for the Forms
2.  Dim frm             As Form
3.  Dim ctlform         As Control
4.  Dim Forms_count     As Integer
5.  For Each frm In Application.Forms
6.      If Not (frm.Name Like "INFO") Then
7.          Forms_count = Forms_count + 1
8.      End If
9.  Next frm
10.
11. If Forms_count > 0 Then
12.     JSON = JSON &        "'Forms':["
13.     For Each frm In Application.Forms
14.         If Not (frm.Name Like "INFO") Then
15.             JSON = JSON &        "'Formname':'"
16.             JSON = JSON & frm.Name &        "','fields':["
17.             On Error Resume Next
18.             For Each ctlform In frm.Controls
19.                 JSON = JSON &        "'name':'" & ctlform.Properties("Name") &
    "','type':'" & ctlform.Properties("ControlType") & "','source':'" &
    ctlform.Properties("ControlSource") & "'},"
20.             Next ctlform
21.             JSON = Left(JSON, Len(JSON) - 1)
22.             JSON = JSON & "]},"
23.         End If
24.     Next frm
25.     JSON = Left(JSON, Len(JSON) - 1)
26.     JSON = JSON & "]"
27. Else
28.     JSON = JSON &        "'Forms':0"
29. End If
```

Figura 10: Código de recopilación de formularios

Para recorrer los formularios se inicializan las variables *frm* y *ctlform* las cuales contienen toda la información relevante de los formularios y sus controles. Se procede a recorrer todos los formularios dentro de la base de datos con un bucle que obvia aquellos formularios que no son directamente creados por el usuario, por esto se coloca la condición que excluye los formularios con nombres parecidos a "INFO". En este caso se utiliza el método *Application.Forms* para interactuar con los formularios.

Se realiza un contador *Forms_count* para contar el número de formularios creados por el usuario y esta variable será utilizada en la condición *if* para construir el JSON dependiendo de su valor. En el caso de que *Forms _count* sea mayor que cero se procederá a construir el JSON tomando el nombre del formulario y todas las características de sus controles. En este caso las características principales de sus controles son su nombre, control type y control source. En el caso de que *Forms _count* no sea mayor que cero el JSON se construirá con formularios igual a cero.

A continuación, en la figura 11 se puede observar un esquema de que información se está recopilando de los formularios con sus controles y la estructura del JSON.

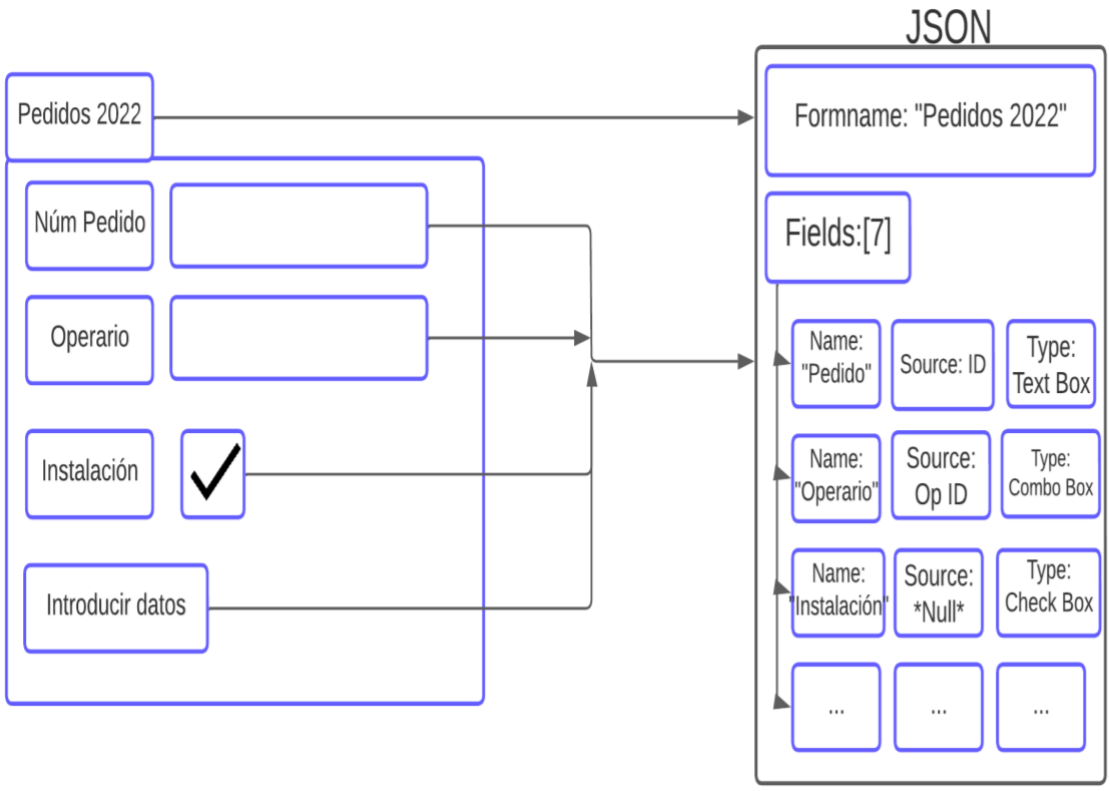


Figura 11:Diagrama de recopilación de Formularios a JSON

3.5.4 Informes

```
1.  'Look for the reports
2.  If Application.Reports.Count > 0 Then
3.      JSON = JSON &        “'Reports':["
4.      Dim rep         As Report
5.      Dim ctlreport   As Control
6.      For Each rep In Application.Reports
7.          JSON = JSON &        “'Reportname':'"
8.          JSON = JSON & rep.Name &        “','fields':["
9.          On Error Resume Next
10.         For Each ctlreport In rep.Controls
11.             JSON = JSON &        “'name':'" & ctlreport.Properties("Name") &
    "','type':'" & ctlreport.Properties("ControlType") & "'},"
12.         Next ctlreport
13.         JSON = Left(JSON, Len(JSON) - 1)
14.         JSON = JSON & "]},"
15.     Next rep
16.     JSON = Left(JSON, Len(JSON) - 1)
17.     JSON = JSON & "]"
18. Else
19.     JSON = JSON &        “'Reports':0"
20. End If
```

Figura 12: Código de recopilación de informes

Para recorrer los informes se inicializan las variables *rep* y *ctlreport* las cuales contienen toda la información relevante de los informes y sus controles. Se procede a recorrer todos los informes dentro de la base de datos con un bucle utilizando el método *Application.Reports* para poder interactuar con los informes.

No se necesita realizar un contador ya que no existen informes que no sean creados directamente por el usuario. En el caso de que *Application.Reports.Count* sea mayor que cero procederá a construir el JSON tomando el nombre del reporte y todas las características de sus controles. En este caso las características principales de sus controles son su nombre y control type. En el caso de que *Application.Reports.Count* no sea mayor que cero el JSON se construirá con informes igual a cero.

A continuación, en la figura 13 se puede observar un esquema de que información se está recopilando de los informes con sus controles y la estructura del JSON.

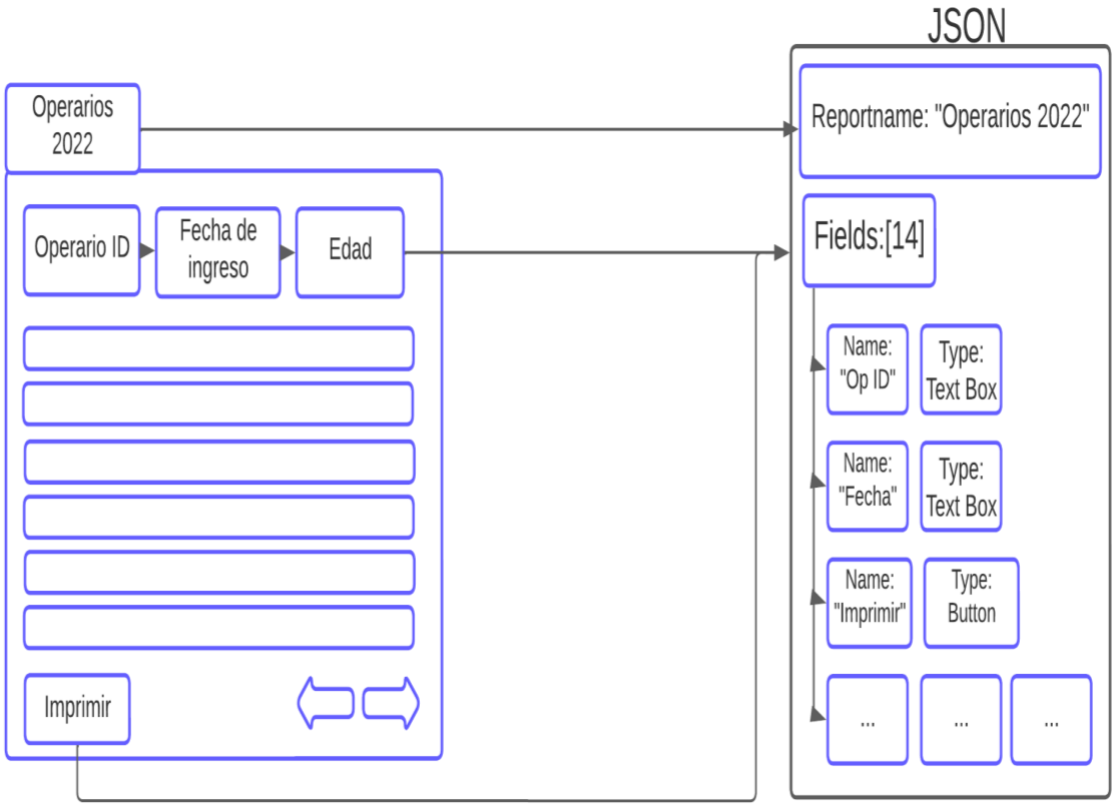


Figura 13: Diagrama de recopilación de Informes a JSON

3.5.5 Relaciones

```
1.  'Look for the Relationships
2.  For Each relLoop In db.Relations
3.      If Not relLoop.Name Like "MSys*" Then
4.          Relationships_count = Relationships_count + 1
5.      End If
6.  Next relLoop
7.
8.  If Relationships_count > 0 Then
9.      JSON = JSON &        "'Relationships':["
10.     For Each relLoop In db.Relations
11.         If Not relLoop.Name Like "MSys*" Then
12.             JSON = JSON &        "'Relationshipname':'"
13.             JSON = JSON & relLoop.Name &        "','fields':["
14.             JSON = JSON &        "'attributes':'" & relLoop.Attributes &
    "','table':'" & relLoop.table & "','primary key':'" & relLoop.Fields(0).Name &
    "','foreign table':'" & relLoop.ForeignTable & "','foreign key':'" &
    relLoop.Fields(0).ForeignName & "'}]},"
15.         End If
16.     Next relLoop
17.     JSON = Left(JSON, Len(JSON) - 1)
18.     JSON = JSON & "]}"
19. Else
20.     JSON = JSON &        "'Relationships':0}"
21. End If
```

Figura 14: Código de recopilación de relaciones

Para recorrer las relaciones entre tablas se inicializa la variable *relLoop* la cual contiene toda la información relevante de las relaciones entre tablas. Se procede a recorrer todas las relaciones dentro de la base de datos con un bucle utilizando el método *db.Relations* para poder interactuar con las relaciones entre tablas y sus propiedades.

Se realiza un contador *Relationships_count* para contar el número de relaciones creadas por el usuario y esta variable será utilizada en la condición *if* para construir el JSON dependiendo de su valor. En el caso de que *Relationships _count* sea mayor que cero se procederá a construir el JSON tomando el nombre de la relación y todas las características de esta. Las principales características de una relación son su nombre, atributos, tabla primaria, llave primaria, tabla extranjera y por último su llave extranjera. En el caso de que *Relationships _count* no sea mayor que cero el JSON se construirá con relaciones igual a cero.

A continuación, en la figura 15 se puede observar un esquema de que información se está recopilando de las relaciones entre tablas y la estructura del JSON.

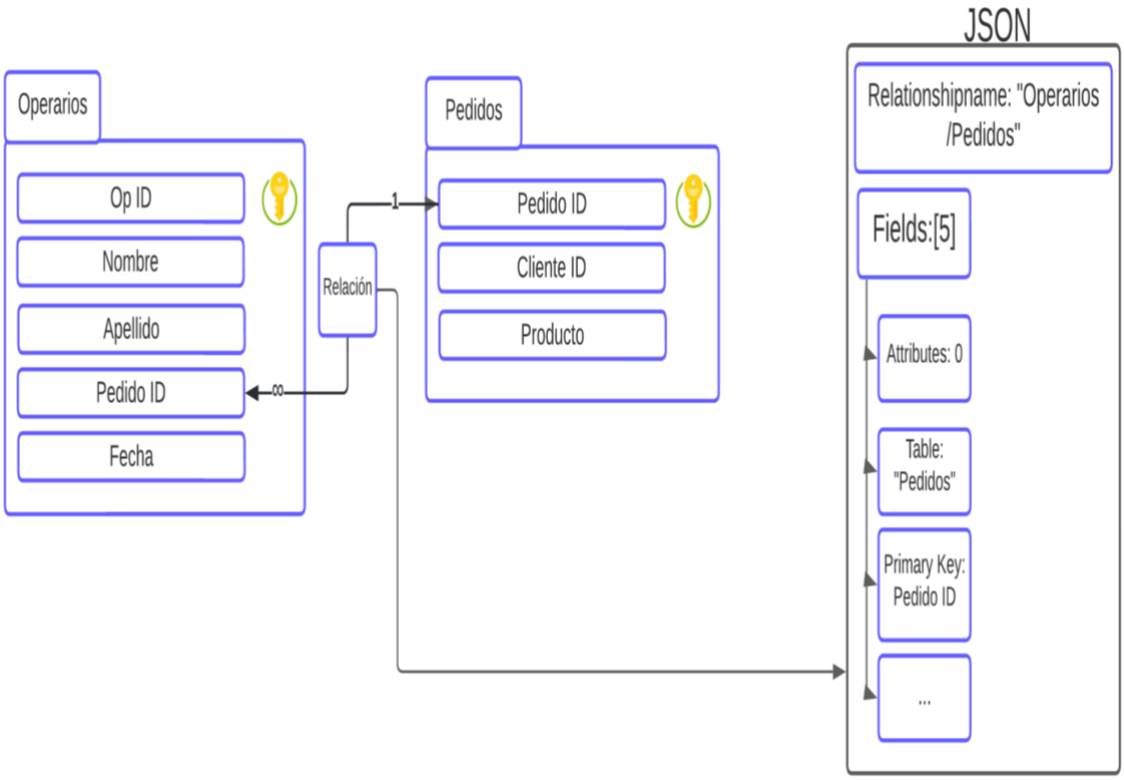


Figura 15: Diagrama de recopilación de Relaciones a JSON

3.6 Conexión con Codex

```
1.  'Send the information to Codex
2.  URLstr = "https://q-server.tecnun.es/codex/Login?username=" + UserName + "&password=" +
    Password
3.  ' Create an XMLHTTP object and add some error trapping
4.  Dim xmlhttp         As New MSXML2.XMLHTTP60, myurl As String
5.  xmlhttp.Open "GET", URLstr, FALSE
6.  xmlhttp.send
7.  'Access to all the active notebooks
8.  posicionActiveNoteBooks = InStr(1, xmlhttp.responseText, "activeNoteBooks=[{")
9.  posicionFinalActiveNotebooks = InStr(1, xmlhttp.responseText,
    "</script><script>studentid=")
10. activeNoteBooks = Mid(xmlhttp.responseText, posicionActiveNoteBooks + 16,
    posicionFinalActiveNotebooks - posicionActiveNoteBooks - 16)
11.
12. Dim JSONactiveNoteBooks As Object
13. Set JSONactiveNoteBooks = JSONConverter.ParseJson(activeNoteBooks)
14. Dim j               As Integer
15. Dim noteBookList    As ArrayList
16. Dim noteBookArray   As Variant
17. Set noteBookList = New ArrayList
18. For j = 1 To JSONactiveNoteBooks.Count
19.     If JSONactiveNoteBooks(j)("sid") = "TD2020" Then
20.         noteBookList.Add JSONactiveNoteBooks(j)("nid")
21.     End If
22. Next j
23. noteBookArray = noteBookList.toArray
24. 'Open all the notebooks from the specified subject
25. Dim z               As Integer
26. z = 0
27. For Each element In noteBookArray
28.     noteBook = "https://q-server.tecnun.es/codex/NoteBookInstance?nid=" + noteBookArray(z)
29.     xmlhttp.Open "GET", noteBook, FALSE
30.     xmlhttp.send
31.     shtml2 = xmlhttp.responseText
32.     z = z + 1
33. Next element
34. posicionInicial = InStr(1, shtml2, "nbinst", vbTextCompare)
35. posicionFinal = InStr(1, shtml2, "nname", vbBinaryCompare)
36. tamano = posicionFinal - posicionInicial
37. enunciado = Mid(shtml2, posicionInicial + 7, tamano - 8)
38. Dim enunciadoJSON   As Object
39. Set enunciadoJSON = JSONConverter.ParseJson(enunciado)
40. ‘Se obtiene el número de preguntas desde enunciadoJSON y se introduce en el textbox'
41.  count_preguntas = enunciadoJSON.Count
42.  total_preguntas.Value = count_preguntas
43. 'Formatear enunciado de Html a texto'
44.  Dim enunciado_formateado As String
45.  enunciado = enunciadoJSON(Numero_Pregunta_Text.Value)("itemcontent")
46. “Send JSON object to Codex’
47. Url = "https://q-server.tecnun.es/codex/SaveItemAnswer?uaid=" +
    enunciadoJSON(Numero_Pregunta_Text.Value)("useranswerid") + "&ans=" + HexString +
    "&grade=1&type=" + enunciadoJSON(Numero_Pregunta_Text.Value)("basetype") + "&iid=" +
    enunciadoJSON(Numero_Pregunta_Text.Value)("iid")
48. xmlhttp.Open "GET", Url, FALSE
49. xmlhttp.send
50. Dim resultado       As String
51. resultado = xmlhttp.responseText
52. calificacion.Value = RemoveHTML(resultado)
```

Figura 16: Código de envió a Codex

Primero se establece una conexión con Codex mediante un objeto XMLHTTP y un url que contiene las credenciales del estudiante recolectadas del formulario INFO. Al establecer la conexión se procede a acceder a los notebooks activos y crear una variable JSONactiveNotebooks en la que se pueda interactuar con esta al igual que un objeto JSON. Después de parsear JSONactiveNotebooks se extrae toda la información del cuaderno que se desee, en este caso se utiliza de ejemplo el cuaderno "TD2020" el cual tiene solo una actividad.

Una vez obtenida la información del notebook se crea un objeto enunciadoJSON para extraer toda la información de cada pregunta dentro de la actividad. Con este objeto enunciadoJSON es posible extraer el enunciado de cada pregunta y desplegarlo en la caja de texto de la interfaz gráfica. Por último, se utiliza enunciadoJSON para construir un url para enviar la respuesta, enviando como respuesta la variable *HexString*. *HexString* es el JSON construido en la recopilación de datos en forma hexadecimal, esto se realiza ya que no se pueden enviar ciertos caracteres a través de un url para enviar la información a Codex. La transformación del JSON a hexadecimal esta descrita en el anexo.

A continuación, en la figura 17 se puede observar un esquema de la comunicación de Access a Codex y sus principales funcionalidades.

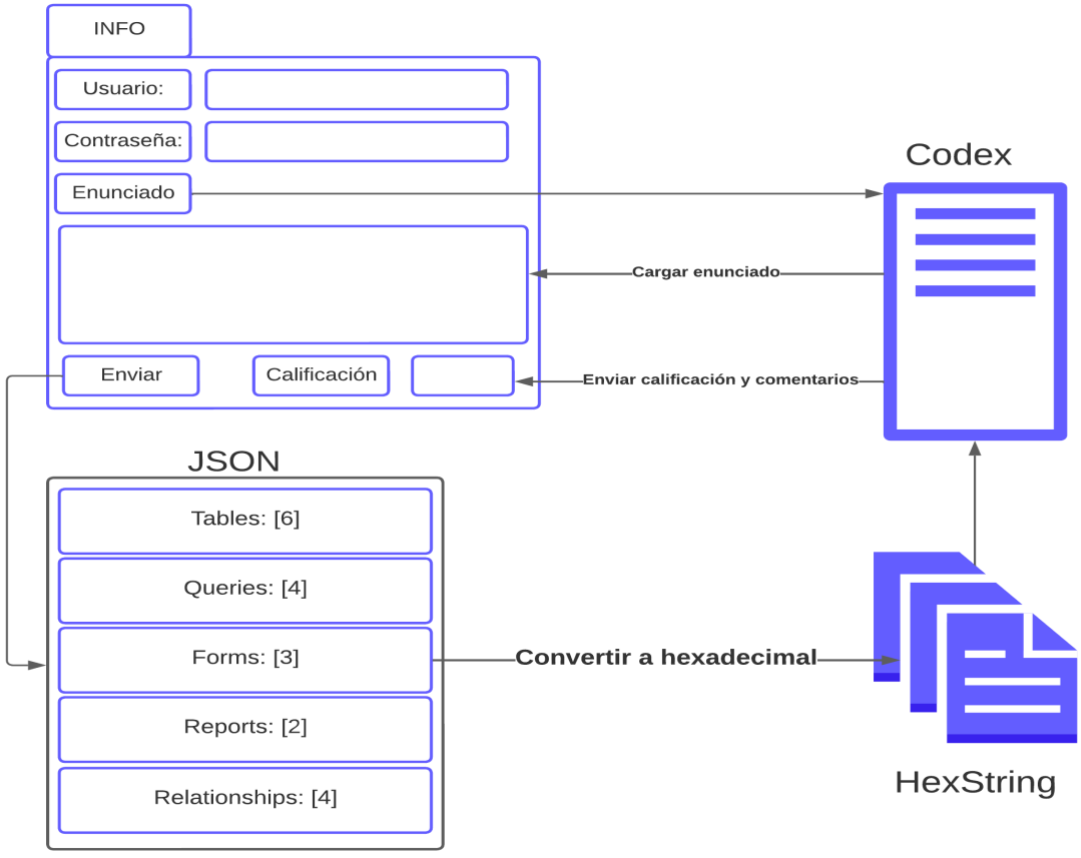


Figura 17: Diagrama de comunicación Access-Codex

4. RESULTADOS OBTENIDOS

Tras la finalización del proyecto se ha conseguido un diseño completo de la interfaz gráfica desde Access capaz de evaluar correctamente todos los objetos de interés dentro de la base de datos, siendo estos las tablas, consultas, informes, informes y relaciones.

La conexión de la interfaz gráfica con Codex es capaz de enviar el JSON con toda la información recopilada de la base de datos y también puede recibir la calificación obtenida con comentarios para notificar al estudiante. El diseño de la interfaz gráfica es simple e intuitivo tanto para el estudiante como para el profesor asegurando una evaluación correcta y satisfactoria para ambas partes.

Las principales funcionalidades desarrolladas dentro del proyecto fueron la interfaz gráfica desde Access, la recopilación de objetos y transformación a formato JSON y por último él envió y recibo de información de Access a Codex.

Por último, se realizó una práctica evaluada completa con enunciados y respuestas en formato JSON como demo del proyecto. Esta práctica evaluada descrita en el anexo es capaz de evaluar todos los elementos de interés de la base de datos y sirve como una breve introducción a la asignatura para los estudiantes.

5. CONCLUSIONES Y RECOMENDACIONES

El desarrollo del proyecto se centraba en automatizar la evaluación de la asignatura sin que esto afectara a la calidad de la corrección, se buscó que la solución fuera lo más objetiva posible al recopilar la información de la base de datos.

El desarrollo de la interfaz gráfica fue realizado a través de los formularios de Access y se logró un diseño intuitivo en el cual el estudiante es capaz de introducir sus credenciales y realizar completamente una prueba evaluada sin necesidad de usar el navegador.

El código del proyecto fue elaborado en VBA y se siguió una metodología centrada fuertemente en la programación orientada a objetos, gracias a esto el código es fácil de mantener y adaptar a cambios futuros. El proyecto consiguió la primera aplicación Office en trabajar con Codex y al ser programado con VBA demuestra que se puede trasladar a otras aplicaciones, como por ejemplo Excel, para evaluar otras asignaturas de forma automática.

El análisis económico del proyecto prueba un resultado positivo considerable con un ahorro de alrededor de 20.000 € anuales provenientes del tiempo de corrección de prácticas y pruebas evaluadas. El retorno de inversión es del 286,5% en tan solo 5 años, recuperando la inversión inicial en tan solo el primer año de uso.

Se desarrollo una práctica evaluable para los estudiantes como introducción a la asignatura en la cual pueden usar la interfaz gráfica para la evaluación. El objeto JSON desarrollado en la recopilación de datos se probó compatible con el método de evaluación de Codex.

Por último, las recomendaciones a seguir para futuros desarrollos del proyecto son:

- Mostrar todos los enunciados de la practica a la vez desde la pantalla de texto en la interfaz gráfica.
- Agregar la posibilidad de escoger cuadernos y actividad desde la interfaz gráfica en vez del profesor tener que especificarlo en el código.
- Recopilar la información de formularios y informes sin que tengan que estar abiertos.
- Ver el número de intentos restantes y la calificación de una pregunta sin tener que enviar el resultado.
- Mensaje de alerta si las credenciales del alumno no son correctas.

6. REFERENCIAS

[1] Docs.microsoft.com. 2022. *Documentación para desarrolladores de Access*. [online] Disponible en: <https://docs.microsoft.com/es-es/office/client-developer/access/access-home>
[Consultado el 20 de Julio, 2022].

[2] Json.org. 2022. *ECMA-404 The JSON Data Interchange Standard.* [online] Disponible en: <https://www.json.org/json-es.html>
[Consultado el 20 de Julio, 2022].

[3] Gillis, A., 2021. What is Object-Oriented Programming (OOP)? [online] TechTarget. Disponible en: <https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP#:~:text=Object%2Doriented%20programming%20(OOP)%20is%20a%20computer%20programming%20model,has%20unique%20attributes%20and%20behavior.>
[Consultado el 26 de Julio, 2022].

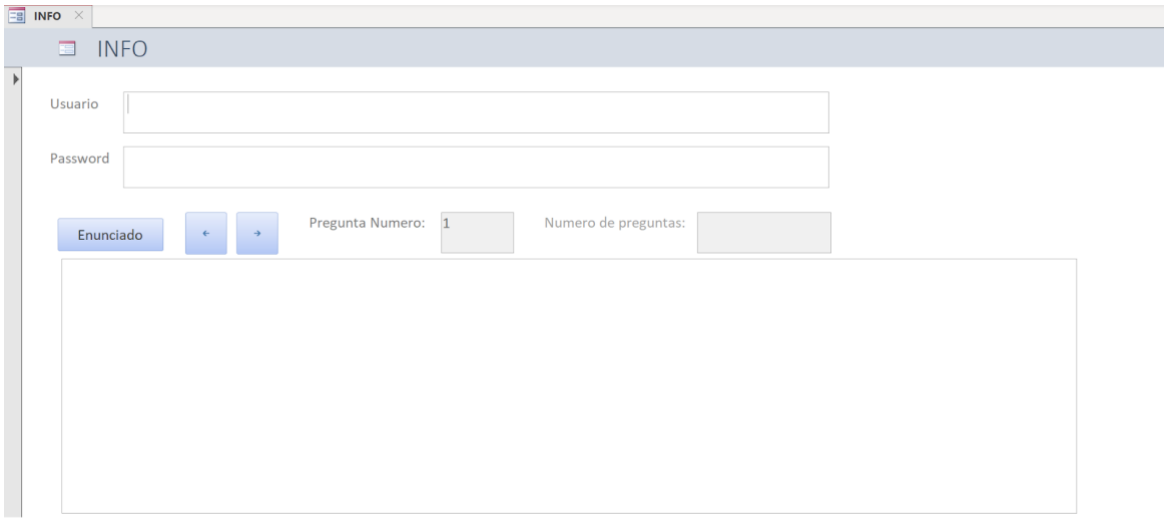
[4] Talent.com. 2022. Salario para Ingeniero Informático en España - Salario Medio. [online] Disponible en: <https://es.talent.com/salary?job=ingeniero+inform%C3%A1tico#:~:text=El%20salario%20ingeniero%20inform%C3%A1tico%20promedio,hasta%20%E2%82%AC%2035.000%20al%20a%C3%B1o.>
[Consultado el 27 de Julio, 2022].

[5] Jobted.es. 2022. ¿Cuánto Cobra un Profesor de Universidad? (Sueldo 2022). [online] Disponible en: <https://www.jobted.es/salario/profesor-universidad#:~:text=El%20salario%20medio%20de%20un,salario%20medio%20anual%20en%20Espa%C3%B1a.>
[Consultado el 27 de Julio 2022].

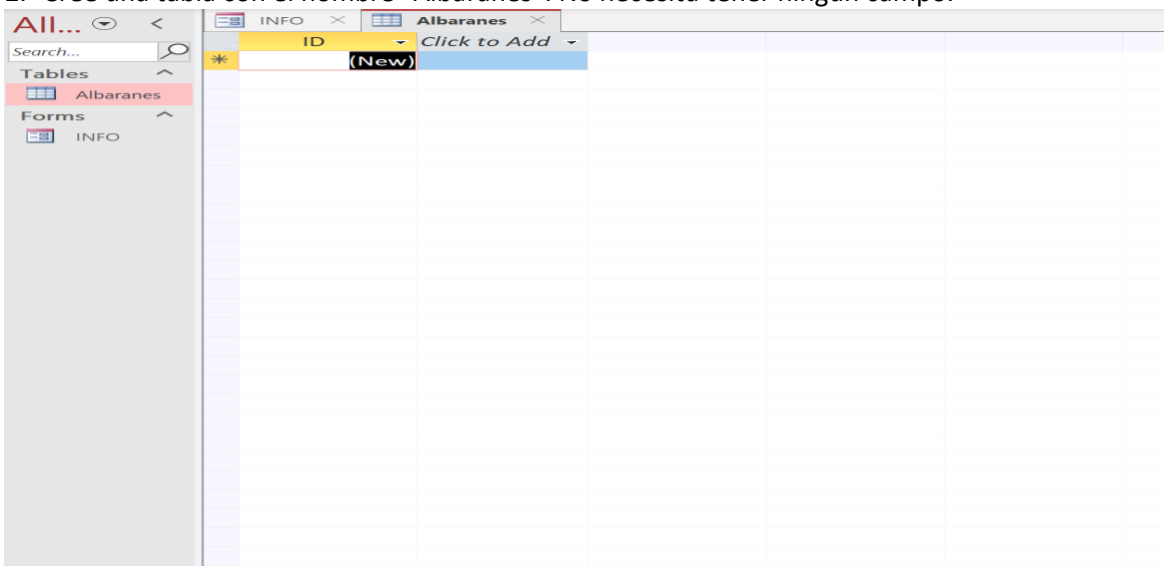
7. ANEXO

7.1 Práctica 1: Análisis y diseño de bases de datos – Introducción a Tablas, Relaciones, Consultas, Formularios y Informes

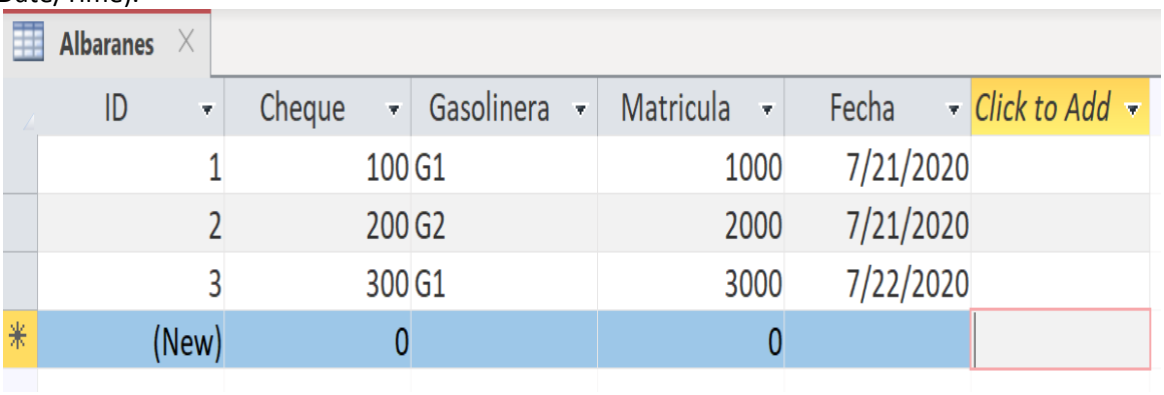
1.- Ingrese su usuario y clave de Codex en el formulario “INFO” y presione el botón “Enunciado” para empezar. En cada pregunta asegúrese de enviar su respuesta con el botón “Enviar” abajo a la izquierda. El resultado obtenido será desplegado en la ventanilla de “Calificación”.

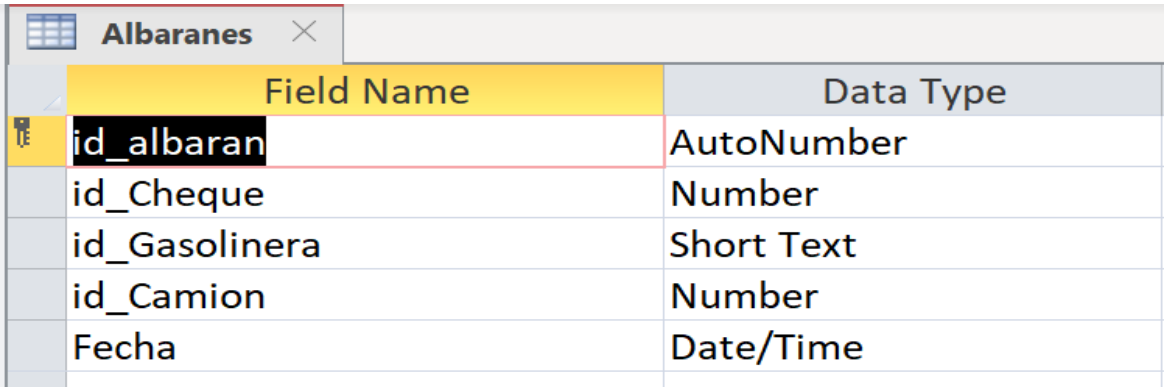


2.- Creé una tabla con el nombre “Albaranes”. No necesita tener ningún campo.

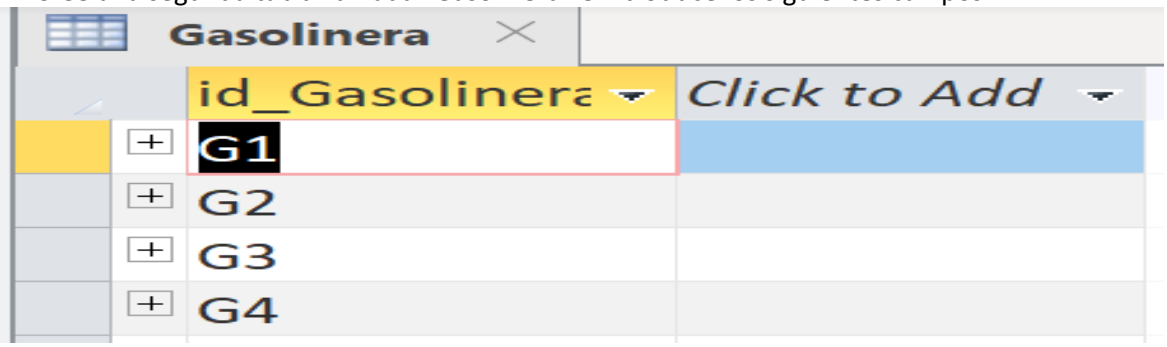


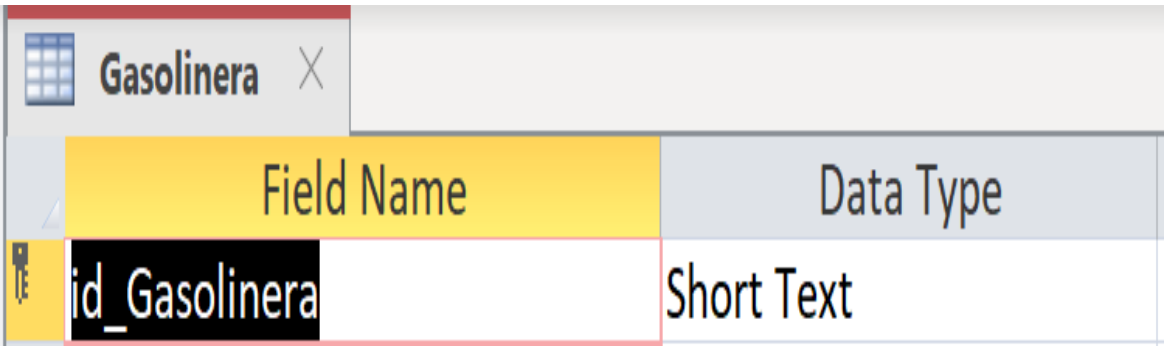
3.- Introduce los campos ID, Cheque, Gasolinera, Matricula y Fecha dentro de la tabla. Cada campo debe de tener su data type correspondiente (AutoNumber, Number, Short Text, Number, Date/Time).



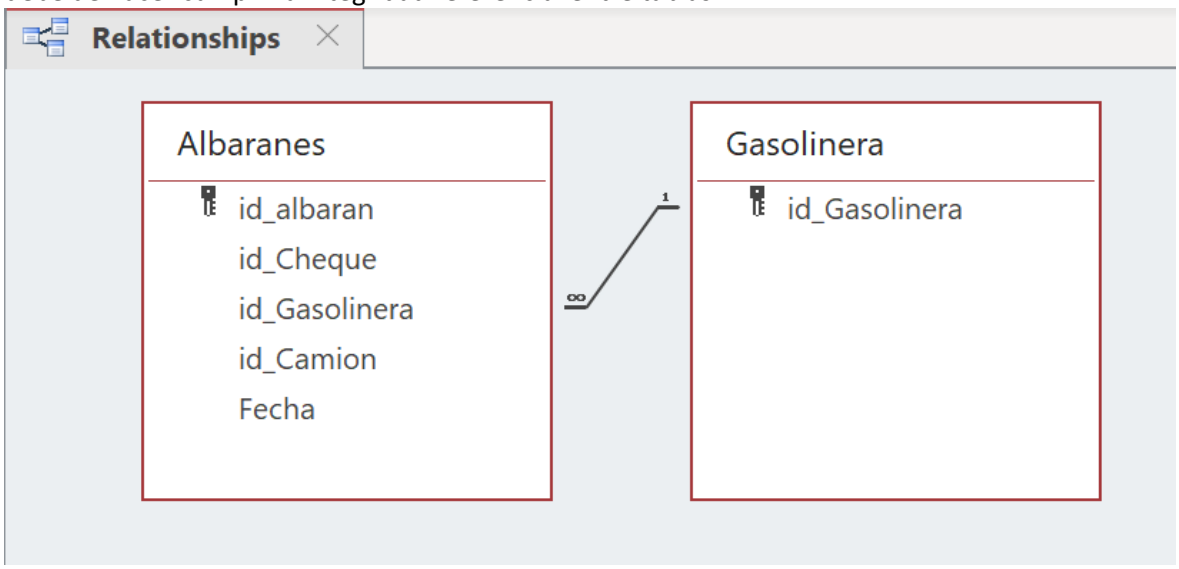


4.- Creé una segunda tabla llamada “Gasolinera” e introduce los siguientes campos:

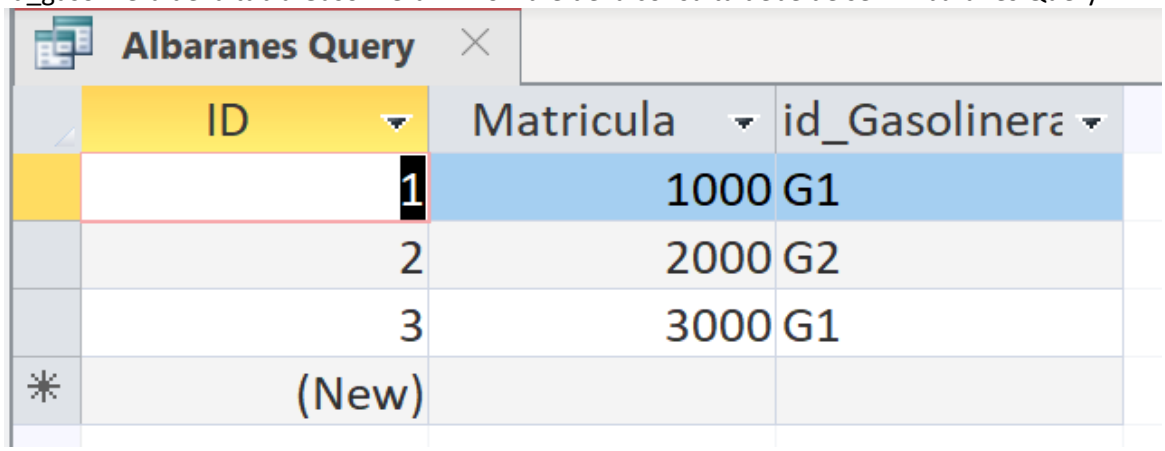




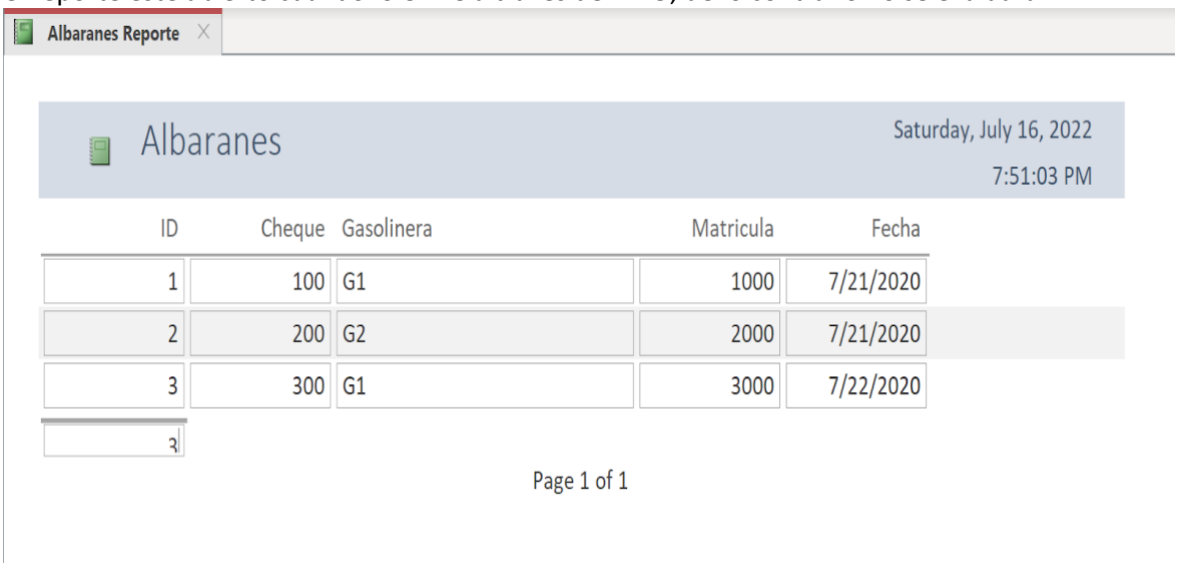
5.- Incorpore una relación entre las tablas “Albaranes” y “Gasolinera”. Recuerda que la relación debe de hacer cumplir la integridad referencial entre tablas.



6.- Realicé una consulta que contenga los campos id_albaran y fecha de la tabla Albaranes y id_gasolinera de la tabla Gasolinera. El nombre de la consulta debe de ser “Albaranes Query”.



7.- Creé un reporte de Albaranes automático con el nombre “Albaranes Reporte”. Asegúrese que el reporte este abierto cuando lo envíe a través de INFO, de lo contrario no se evaluara.



8.- Creé un formulario automático de Gasolineras, con el nombre “Gasolinera Form”. Asegúrese que el formulario este abierto cuando lo envíe a través de INFO, de lo contrario no se evaluara.