



Escuela de Ingeniería en Computación

Bases de Datos I - Grupo 1

I Tarea Programada

Intérprete de Álgebra Relacional

Documentación

Profesor:

William Mata Rodríguez

Integrantes:

Gerardo Gutiérrez Quirós 2016140286

José Gómez Casasola 2016095929

Greivin Berrocal Solano 2016220317

II Semestre 2017

Fecha de entrega:

Domingo 8 de Octubre del 2017

# Contenido

<b>Introducción</b>	<b>3</b>
<b>Enunciado - Proyecto 1</b>	<b>4</b>
INTÉRPRETE DE ÁLGEBRA RELACIONAL	4
REQUERIMIENTOS DEL PROYECTO	5
Operación: Selección	6
Operación: proyección generalizada	6
Operación: Unión	6
Operación: Diferencia de conjuntos	7
Operación: Producto cartesiano	7
Operación: Intersección	7
Operación: División	7
Operación: Renombrar una relación y sus atributos	8
Operación: concatenación (join)	8
Operación: concatenación natural(natural join)	8
Operación: Agregación	9
Operación: Agrupación	9
Tabla resultante	9
Resultados	10
Funciones adicionales del intérprete:	11
Ver tablas de la base de datos	11
Ver tablas temporales	11
Ver referencia cruzada atributos / tablas	12
Ayuda	12
Acerca de	12
Salir	12
DOCUMENTACIÓN DEL PROYECTO	14
<b>Temas investigados para desarrollar el proyecto</b>	<b>16</b>
Lenguaje de programación C#	16
Conexión con la Base de Datos	16
Herramienta de versionamiento de código Git	19
Estados	20
Secciones principales de un proyecto	21
Objetos	21
GitHub	22
Crear una clave y autenticar al usuario	22

Crear un repositorio	24
Clonar un repositorio	25
Seguimiento de nuevos archivos	25
Añadir un commit	25
Subir cambios	26
<b>Conclusiones del trabajo</b>	<b>27</b>
Problemas encontrados	27
Soluciones a los problemas	27
Aprendizajes obtenidos	28
<b>Rúbrica de evaluación y análisis de resultados</b>	<b>29</b>
<b>Manual de Usuario</b>	<b>31</b>
Pantalla Principal	31
Operaciones	34
Presentaciones	36
Acerca de	37
Salir	37
<b>Fuentes y otros objetos</b>	<b>38</b>
Fuentes	38

## Introducción

El presente documento es la sección escrita de la primer tarea programada del curso Bases de Datos I con el profesor William Mata Rodriguez. En este documento se presenta el enunciado de la tarea así como los aspectos investigados por los integrantes del grupo de trabajo, se muestra también el análisis del trabajo realizado por el equipo en las últimas semanas, un manual de usuario e información adicional sobre el desarrollo del proyecto.

El proyecto tiene como objetivo tomar en práctica los conocimientos adquiridos a lo largo del semestre sobre el álgebra relacional y el lenguaje de consultas a bases de datos SQL. Consiste en obtener nombres de tablas, predicados y realizar las operaciones básicas del álgebra relacional en el lenguaje SQL, es decir, basados en la información brindada por el usuario se muestra la simbología de la operación en álgebra relacional, la forma que tendrían las instrucciones de consulta en SQL y el resultado de las mismas.

Para el desarrollo del proyecto se utilizó la herramienta de versionamiento de código GitHub para manejar adecuadamente las actualizaciones que realizó el equipo con el código del proyecto. Se eligió utilizar el lenguaje SQL debido a que es en el cual poseemos mayor conocimiento, el tiempo que llevamos utilizando bases de datos es poco y la experiencia aún no es suficiente como para poder adaptarnos adecuadamente a otro lenguaje de bases de datos, era el más indicado a desarrollar con el objetivo de este proyecto. Por otro lado, el lenguaje escogido para realizar la programación lógica de la aplicación es C#, se escogió aprender de este lenguaje debido a las buenas referencias que nos fueron brindadas sobre el mismo, su capacidad y facilidad de conexión con bases de datos SQL, así como las facilidades que ofrece para el manejo de interfaces de usuario.

## Enunciado - Proyecto 1

### INTÉRPRETE DE ÁLGEBRA RELACIONAL

Un componente fundamental en los modelos de datos es el lenguaje de manipulación, o lenguaje de consulta, para la extracción y actualización de los datos.

Entre estos lenguajes de consulta están el álgebra relacional y el cálculo relacional definidos por Codd (1971). Ambos lenguajes son teóricos y formales. Se han utilizado como base para desarrollar otros lenguajes de manipulación de datos en el modelo relacional como el SQL.

Este proyecto consiste en desarrollar un intérprete de algunas operaciones del lenguaje álgebra relacional, ofreciendo también las instrucciones de SQL equivalentes y los datos obtenidos con la operación.

Durante el curso estudiamos conceptos de SQL desde el punto de vista del estándar de la ISO. Al usar un manejador de base de datos (DBMS o SABD) de un fabricante particular, podemos encontrar algunas diferencias pues estos han modificado parte del estándar y han agregado sus propias funcionalidades. En tales casos hay que consultar la documentación respectiva para adaptarse a un DBMS específico.

El DBMS a utilizar en el proyecto es Microsoft SQL Server o MySQL. Las demás herramientas que necesiten para desarrollar esta aplicación Ustedes las pueden seleccionar.

Para interactuar con el usuario la aplicación debe usar una GUI (Graphical User Interface) diseñada por Ustedes que cumpla con los requerimientos solicitados.

El interpretador solo hace una operación del álgebra a la vez, no hay operaciones anidadas.

El desarrollo del proyecto es en equipos de 3 estudiantes máximo, uno de ellos lo deben nombrar como el coordinador. Importante: las experiencias han demostrado

que los proyectos en equipos que no han sido administrados adecuadamente van a fallar, así que en cuanto noten que se presentan problemas al respecto de inmediato tratenlo primeramente con los miembros del equipo, y de no resolver lo comunican al profesor.

Cualquier comunicación al profesor que vaya copiada a todos los miembros del equipo.

Buenas prácticas de la ingeniería de software: deben usar un software para administrar el desarrollo de proyectos en equipo el cual incluye un control de versiones (Git, otros).

## REQUERIMIENTOS DEL PROYECTO

Las operaciones del álgebra relacional que debe manejar el interpretador son:

- Selección
- Proyección generalizada
- Unión
- Diferencia de conjuntos
- Producto cartesiano
- Intersección
- División
- Renombrar un relación y sus atributos
- Concatenación (join)
- Concatenación natural (natural join)
- Agregación
- Agrupación

El usuario debe tener un mecanismo para seleccionar la operación que necesite. El programa debe quedarse pidiendo funciones al usuario hasta que él pida finalizar el programa.

En general para cada operación Usted va a encontrar en esta especificación los datos que se deben solicitar al usuario, las validaciones que se deben hacer y los resultados esperados.

En las validaciones algunos errores deben tener un manejo de errores personalizado, es decir, se envían al usuario los mensajes tales como se especifican aquí. Cuando no se indiquen mensajes personalizados el manejo de los errores puede hacerlo como ustedes decidan.

### Operación: Selección

Datos solicitados: Tabla  
Predicado (string)  
Tabla resultante (se explica más adelante)

Validaciones:  
Tabla debe existir. Mensaje personalizado cuando no exista: ERROR:  
NO EXISTE LA TABLA nombre\_de\_tabla  
Otros errores Usted los maneja.

### Operación: proyección generalizada

Datos solicitados: Tabla  
Expresión de la proyección generalizada (string)  
Tabla resultante

Validaciones:  
Tabla debe existir. Mensaje personalizado cuando no exista: ERROR:  
NO EXISTE LA TABLA nombre\_de\_tabla  
Otros errores Usted los maneja.

### Operación: Unión

Datos solicitados: Tabla 1  
Tabla 2  
Tabla resultante

Validaciones:  
Tablas deben existir. Mensaje personalizado cuando no existan:  
ERROR: NO EXISTE LA TABLA nombre\_de\_tabla  
Las tablas deben tener igual aridad. Mensaje personalizado: ERROR:  
TABLAS CON DIFERENTE ARIDAD. LA TABLA 1 TIENE ARIDAD "X" Y LA  
TABLA 2 TIENE ARIDAD "Y"

Cada atributo respectivo debe tener el mismo dominio. Mensaje personalizado: ERROR: DOMINIOS DIFERENTES. EL ATRIBUTO “X” TIENE “DOMINIO X” Y EL ATRIBUTO “Y” TIENE “DOMINIO “Y”.

Otros errores Usted los maneja.

### Operación: Diferencia de conjuntos

Datos solicitados: Tabla 1  
Tabla 2  
Tabla resultante

Validaciones:  
Igual a la operación de unión.

### Operación: Producto cartesiano

Datos solicitados: Tabla 1  
Tabla 2  
Tabla resultante

Validaciones:  
Tablas deben existir. Mensaje personalizado cuando no existan:  
ERROR: NO EXISTE LA TABLA nombre\_de\_tabla  
Otros errores Usted los maneja.

### Operación: Intersección

Datos solicitados: Tabla 1  
Tabla 2  
Tabla resultante

Validaciones:  
Igual a la operación de unión.

### Operación: División

Datos solicitados: Tabla 1  
Tabla 2  
Tabla resultante

Validaciones:  
Tablas deben existir. Mensaje personalizado cuando no existan:  
ERROR: NO EXISTE LA TABLA nombre\_de\_tabla  
Todos los atributos de Tabla 2 deben estar en Tabla 1. Mensaje personalizado: ERROR: EL ATRIBUTO “X” DE LA TABLA nombre\_de\_tabla1 NO ESTÁ EN LA TABLA nombre\_de\_tabla2.



Otros errores Usted los maneja.

### Operación: Renombrar una relación y sus atributos

Físicamente en la base de datos tanto la tabla como los atributos siguen manteniendo el nombre de origen. El renombramiento es solo para efectos de conocer esos objetos con otro nombre en la corrida actual de esta aplicación.

Datos solicitados: Tabla

Nombres de atributos (string): Cada uno de los nombres nuevos de los atributos: el i-ésimo atributo dado aquí corresponde al i-ésimo atributo de la tabla.

Tabla resultante

Validaciones:

Tabla debe existir. Mensaje personalizado cuando no exista: ERROR: NO EXISTE LA TABLA nombre\_de\_tabla

Por cada atributo en la tabla debe existir un nombre de atributo. Mensaje de error personalizado: ERROR: NO HAY CORRESPONDENCIA EN LA CANTIDAD DE ATRIBUTOS. LA TABLA nombre\_de\_tabla TIENE "X" ATRIBUTOS Y SE ESTÁN DANDO "Y" ATRIBUTOS.

Otros errores Usted los maneja.

### Operación: concatenación (join)

Datos solicitados: Tabla 1

Tabla 2

Predicado (string): para unir las tablas

Tabla resultante.

Validaciones:

Tablas deben existir. Mensaje personalizado cuando no existan: ERROR: NO EXISTE LA TABLA nombre\_de\_tabla

Otros errores Usted los maneja.

### Operación: concatenación natural(natural join)

Datos solicitados: Tabla 1

Tabla 2

Tabla resultante

Validaciones:

Tablas deben existir. Mensaje personalizado cuando no existan: ERROR: NO EXISTE LA TABLA nombre\_de\_tabla

Para hacer este tipo de join deben existir al menos dos nombres de atributos comunes (y dominios comunes) entre las dos tablas. Mensaje personalizado de error: ERROR: NO HAY ATRIBUTOS COMUNES

Otros errores Usted los maneja.

### Operación: Agregación

Datos solicitados: Tabla

Lista de operaciones de agregación (string)

Tabla resultante

Se permiten al menos las funciones básicas: sum, count, min, max, avg.

Validaciones:

Tablas deben existir. Mensaje personalizado cuando no existan:  
ERROR: NO EXISTE LA TABLA nombre\_de\_tabla

Otros errores Usted los maneja.

### Operación: Agrupación

Datos solicitados: Tabla

Lista de atributos que indican las agrupaciones a realizar (string)

Lista de operaciones de agregación que se van a aplicar a los grupos (string)

Tabla resultante

Se permiten al menos las funciones básicas: sum, count, min, max, avg.

Validaciones:

Tablas deben existir. Mensaje personalizado cuando no existan:  
ERROR: NO EXISTE LA TABLA nombre\_de\_tabla

Otros errores Usted los maneja.

### Tabla resultante

El usuario puede dejar en blanco este dato solicitado o dar un nombre de tabla.

En caso de dar un nombre significa que necesita guardar los resultados en una tabla temporal para usos posteriores en otras operaciones. Note que este guardado es la operación de asignación.

Estas tablas donde se guardan los resultados son temporales, van a existir solamente cuando esté corriendo la aplicación. Cuando la aplicación termina las tablas temporales deben eliminarse de la base de datos.

No se permite dar el nombre de una tabla que sea permanente de la base de datos.

Validar esta condición, mensaje personalizado de error:

ERROR: NO SE PUEDE DEJAR EL RESULTADO EN UNA TABLA PERMANENTE DE LA BASE DE DATOS.

## Resultados

Después de validar los datos solicitados en cada operación se deben mostrar tres resultados:

- La instrucción del álgebra relacional: siempre debe estar visible al usuario
- La instrucción equivalente de SQL: siempre debe estar visible al usuario
- La tabla resultante: el programa debe ofrecer una forma para que el usuario pueda desplazarse por los resultados: se despliega una línea por cada tupla

Ejemplo usando la base de datos Banco: determinar los nombres de los clientes del banco excepto los que vivan en la ciudad León.

Esta consulta se debe hacer en dos operaciones: una selección y una proyección.

Primera operación: selección:

Tabla: cliente

Predicado: ciudad\_cliente <> 'León'

Tabla resultante: r1

Instrucción de álgebra relacional:  $r1 \sigma_{ciudad\_cliente \neq 'León'}(cliente)$

Instrucción SQL:

`select * from cliente where ciudad_cliente <> 'León'`

Tabla resultante: r1

<b>nombre cliente</b>	<b>calle cliente</b>	<b>ciudad cliente</b>
Abril	Preciados	Valsaín
Amo	Embajadores	Arganzuela

Se manejan  
barras de scroll  
horizontal y  
vertical para  
navegar por el  
resultado



Segunda operación: proyección generalizada

Tabla: r1

Expresión: nombre\_cliente

Tabla resultante:

Instrucción de álgebra relacional:

$\Pi$  nombre\_cliente (r1)

Instrucción SQL:

```
select distinct nombre_cliente from r1
```

Tabla resultante:

... datos ...

Funciones adicionales del intérprete:

[Ver tablas de la base de datos](#)

Nombre de la tabla1

Atributo1 Dominio Propiedades (PK, FK, etc.) (un atributo por línea)

...

AtributoN Dominio Propiedades (PK, FK, etc.)

...

Nombre de la tablaN

Atributo1 Dominio Propiedades (PK, FK, etc.)

...

AtributoN Dominio Propiedades (PK, FK, etc.)

[Ver tablas temporales](#)

Nombre de la tabla1

Atributo1 Propiedades (dominio, PK, FK, etc.) (un atributo por línea)

...

AtributoN Propiedades (dominio, PK, FK, etc.)

...

Nombre de la tablaN

Atributo1 Propiedades (dominio, PK, FK, etc.)

...

AtributoN Propiedades (dominio, PK, FK, etc.)

#### Ver referencia cruzada atributos / tablas

Esta consulta permite ver las tablas de la base de datos en donde aparece cada atributo según el diccionario de datos.

Atributo1 Propiedades (dominio, PK, FK, etc.)

Tabla1

...

TablaN

...

AtributoN Propiedades (dominio, PK, FK, etc.)

Tabla1

...

TablaN

#### Ayuda

Esta opción la usaremos para que el usuario pueda ver el Manual de Usuario (explicado más adelante) preparado en la documentación.

#### Acerca de

Esta opción la usaremos para desplegar una ventana con información “Acerca de la aplicación” donde colocaremos al menos los datos del nombre de la aplicación, la versión, la fecha de creación y los autores.

#### Salir

Esta opción se usa para salir del programa.

En este momento las tablas temporales se deben borrar.

Puede agregar cualquier funcionalidad que usted considere va a mejorar el producto.

Al nombre de la base de datos asígnele: bdproy1

Al nombre del esquema asígnele: proy1

Crear dos usuarios:

**dbaproxy1** : va a tener la responsabilidad de DBA. Como DBA tiene todos los privilegios. Con este usuario se van a crear los objetos de la base de datos. Para pruebas puede usar la base de datos Banco que se ha trabajado en clases.

**usproy1** : va a ser un usuario regular de la base de datos. Debe tener estos privilegios de acceso:

- Lectura para los datos de las tablas permanentes.
- CRUD para las tablas temporales.

## DOCUMENTACIÓN DEL PROYECTO

### REQUISITO PARA REVISAR EL PROYECTO

**El requisito consiste en presentar la documentación del proyecto indicada en esta sección.**

**La nota de la documentación del proyecto sirve para aceptar o rechazar el proyecto: se revisan los proyectos que cumplan con este requisito en un 90% o más.**

Enviar vía tecDigital, sección EVALUACIONES / PROYECTOS, una carpeta comprimida (.rar, .zip, etc.) de nombre **proyecto1** que contenga las siguientes partes:

- Parte 1: Documentación del proyecto (**nombre: documentación\_interpretador álgebra\_relacional.PDF**).
  - Portada.
  - Contenido. (3 p)
  - Enunciado completo del proyecto. (2 p)
  - Temas investigados para desarrollar el proyecto: material no estudiado en el curso. Por cada uno de estos temas debe poner el marco teórico incluyendo de qué trata y el detalle de cómo lo usaron en el proyecto.
    - Temas investigados. (15 p): de estos temas excluya el lenguaje de programación pero incluya las metodologías de programación de bases de datos.
    - Software para manejo de versiones. (15 p)
  - Conclusiones del trabajo: (15 p)
    - Problemas encontrados y soluciones a los mismos.
    - Aprendizajes obtenidos.
  - Rúbrica de evaluación y análisis de resultados (PONGA LA HOJA DE LA RÚBRICA EN PÁGINA NUEVA DE TAL FORMA QUE LOS CONCEPTOS QUEDEN EN UNA MISMA PÁGINA). (15 p)

- Tome la rúbrica de evaluación y por cada concepto calificado Usted debe indicar el % de avance y el análisis de resultados
  - 100: Totalmente desarrollado. No hace falta análisis excepto que requiera hacer alguna observación.
  - 80: Desarrollado parcialmente, un 80% (el % que corresponda). En el análisis indicar: ¿qué hace?, ¿qué falta?, ¿por qué no se completó ?
  - 0: No desarrollado. En el análisis indicar el motivo.
- Partes que desarrolló adicionales a los requerimientos.
  - Manual de usuario (**nombre: manual\_de\_usuario\_interpretador álgebra\_relacional.PDF**). (35 p)

Es un documento de comunicación técnica utilizado para guiar a las personas que usan el software. Explica paso a paso cómo usar cada una de las funcionalidades del programa. Apóyese en imágenes, capturas de pantallas, menús, diagramas y los aspectos que considere van a servir como una guía útil para que el usuario pueda usar el programa. Este manual será usado en la revisión del proyecto para hacer las pruebas desde el punto de vista funcional.
  - Parte 2: Fuentes y otros objetos necesarios para ejecutar la aplicación.

En la semana 10 se revisará un prototipo de la GUI: modelos de pantallas.

**IMPORTANTE: CONOCIMIENTO DE LA SOLUCIÓN PRESENTADA.** En la revisión del trabajo, los estudiantes deben demostrar un completo dominio de la solución que implementaron, tanto desde el punto de vista técnico (uso de herramientas) como de la funcionalidad del proyecto. La revisión se puede hacer individualmente o en grupos, examinando la solución o temas específicos aplicados en el proyecto.

Última línea



## Temas investigados para desarrollar el proyecto

A continuación se muestra información acerca de los programas utilizados para el desarrollo del proyecto, así como algunas técnicas de programación utilizadas. Se indicó omitir los lenguajes de programación, sin embargo decidimos mostrar un mínimo de información sobre C# y posteriormente incluir un link por si el lector desea conocer más sobre el mismo.

### Lenguaje de programación C#

Este lenguaje de programación con seguridad de tipos y orientado a objetos, que permite a los desarrolladores crear una gran variedad de aplicaciones seguras y sólidas que se ejecutan en .NET Framework .NET, fue creado por Microsoft, su primer aparición fue en el año 2000. Este lenguaje de programación tiene la influencia de lenguajes como: C++, Java, Eiffel, Modula-3 y Pascal. Maneja diferentes tipos de paradigmas, entre los cuales caben destacar: Estructurado, Imperativo, POO, Dirigido por eventos, Funcional, Genérico y Reflexivo. Además es un lenguaje multiplataformas. Si desea saber más acerca del programa puede ir a la sección “Fuentes y otros objetos” de este documento donde encontrará la dirección web que lo llevará a la página de Microsoft que brinda una introducción al lenguaje.

### Conexión con la Base de Datos

Para realizar una conexión a una base de datos, se deben tomar en cuenta la siguientes implicaciones: el nombre del servidor de bases de datos que se utilizará, el nombre de la base de datos, el nombre del usuario y la contraseña del mismo, estos últimos dos se manejan dependiendo del tipo de conexión que se quiera realizar, dado que a la hora de realizar una conexión con una base de datos se puede autenticar con los permisos de SQL o bien con los permisos del sistema operativo, dentro de poco se explicará más a detalle cada una de estas conexiones.

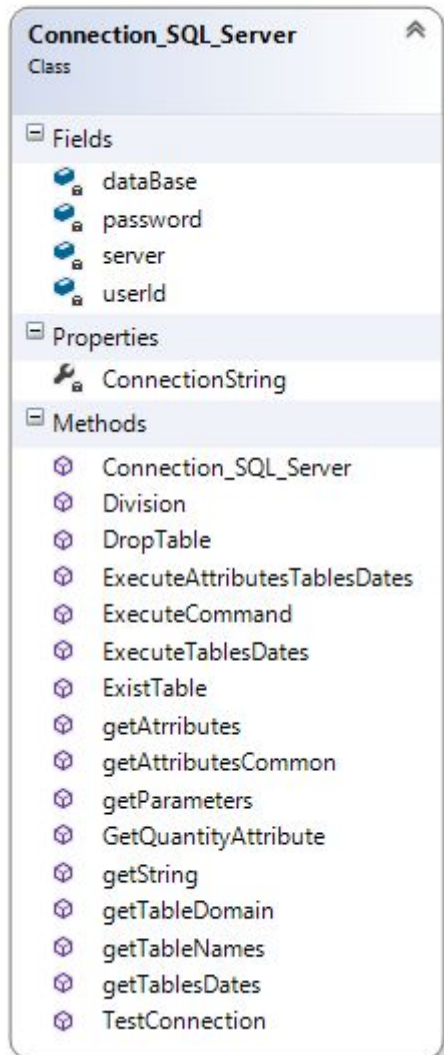
Cuando una conexión es exitosa tenemos el control de la bases de datos, esto quiere decir que podemos desarrollar consultas con el fin de obtener la información que necesitamos de la base de datos. Ahora bien, surge la siguiente pregunta: ¿Cómo conectarse a la base de datos?

Para poder responder esta pregunta, es necesario preguntarse qué lenguaje se utilizará como herramienta para conectar nuestra aplicación con el DBMS. En nuestro caso utilizamos el lenguaje de programación conocido como C#. Para realizar la conexión en este lenguaje, usamos este método:

### C#

```
private static void ReadOrderData(string connectionString)
{
    string queryString = "SELECT OrderID, CustomerID FROM dbo.Orders;";
    using (SqlConnection connection = new SqlConnection(connectionString))
    {
        SqlCommand command = new SqlCommand(queryString, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();
        try
        {
            while (reader.Read())
            {
                Console.WriteLine(String.Format("{0}, {1}", reader[0], reader[1]));
            }
        }
        finally
        {
            reader.Close();
        }
    }
}
```

Para la conexión e intercomunicación de nuestra aplicación con el servidor de base de datos fue necesario crear diversos métodos, los cuales son los siguientes:



- **Connection\_SQL\_Server:** Es el constructor de la clase. Este recibe como parámetros 4 strings que corresponden al nombre del servidor, nombre de la base de datos, nombre de usuario y su contraseña.
- **Division:** Este método se utiliza para realizar la operación de división en el SQL Server. Recibe como parámetro 3 elementos: una lista referenciada y las dos tablas a las cuales se les va a realizar la operación.
- **DropTable:** Elimina las tablas de la base de datos. Recibe como parámetro un string que es el nombre de la tabla a eliminar.
- **ExecuteAttributesTablesDates:** Devuelve una tabla que tiene los nombres de los atributos de todas las tablas y sus respectivos datos.
- **ExecuteCommand:** Este método realiza la operación enviada, tiene como parámetros un string y una lista referenciada.
- **ExecuteTablesDates:** Este método devuelve las tablas que se le indican para poder extraer la información de los metadatos de la base de datos, tales como: nombres de las tablas, llaves primarias, atributos, llaves secundarias y sus referencias, entre otros.
- **ExistTable:** Este método se recibe como parámetro el nombre de una tabla, el cual se utiliza para averiguar si está dentro de la base de datos o no.

- **getAttributes:** Este método recibe como parámetros el nombre de la tabla de la cual se quiere averiguar la información y una lista referenciada que almacenará los atributos.
- **getAttributesCommon:** Toma dos listas y con una tercera empieza a comparar uno por uno los elementos de cada lista, cada que vez que encuentra una similitud este elemento es agregado a la tercer lista.
- **getParameter:** Método que crea el predicado de algunas funciones. Lo que hace es determinar cuáles son los atributos que se quieren obtener de una operación realizada.
- **GetQuantityAttribute:** Realiza una operación del tipo escalar que devolverá un entero que representa la cantidad de atributos que posee una tabla.
- **getString:** Este método se utiliza en la operación de división, sirve para indicar cuáles parámetros se utilizarán para realizar esta operación.
- **getTableDomain:** Este método indica cuáles son los dominios de cada uno de los atributos de la tabla que se especifica.
- **getTableNames:** Obtiene de los metadatos los nombres de las tablas que tiene la base de datos.
- **getTablesDates:** Este método sirve para poder obtener los datos generales de las tablas.
- **TestConnection:** Este método prueba la conexión, indica si la conexión fue exitosa o si hubo algún error en el intento.

## Herramienta de versionamiento de código Git

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Git aporta principalmente las siguientes ventajas:

- Velocidad
- Fuerte apoyo al desarrollo no lineal (miles de ramas paralelas)
- Completamente distribuido

- Capaz de manejar grandes proyectos de manera eficiente
- Auditoría del código
- Control sobre el cambio del proyecto a través del tiempo
- Acceso a versiones anteriores de una forma rápida
- Control de versiones a través de etiquetas
- Seguridad
- Mejora la capacidad de trabajar en equipo
- Merging and branching sumamente eficientes

Git modela sus datos como un conjunto de “fotos” de un mini sistema de archivos. Cada vez que se confirma un cambio, o se guarda el estado del proyecto en Git, él básicamente hace una foto del aspecto de todos los archivos en ese momento, y guarda una referencia a esa foto. La mayoría de las operaciones en Git sólo necesitan archivos y recursos locales para operar.

Todo en Git es verificado mediante una suma de comprobación (checksum en inglés) antes de ser almacenado, y es identificado a partir de ese momento mediante dicha suma. Esto significa que es imposible cambiar los contenidos de cualquier archivo o directorio sin que Git lo sepa.

Casi todas las acciones en Git sólo añaden información a la base de datos de Git. Es muy difícil conseguir que el sistema haga algo que no se pueda deshacer, o que de algún modo borre información. Esto hace que se pueda experimentar sin peligro de fastidiar gravemente las cosas.

## Estados

Git tiene tres estados principales en los que se pueden encontrar los archivos:

- Confirmado (committed): los datos están almacenados de manera segura en la base de datos local

- Modificado (modified): se ha modificado el archivo pero todavía no se ha confirmado a la base de datos
- Preparado (staged). se ha marcado un archivo modificado en su versión actual para que vaya en la próxima confirmación.

## Secciones principales de un proyecto

- Directorio de Git (Git directory): es donde Git almacena los metadatos y la base de datos de objetos para el proyecto creado. Es la parte más importante de Git, y es lo que se copia cuando se hace una clonación a un repositorio desde otro ordenador.
- Directorio de trabajo (working directory): es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que puedan ser usados y modificados.
- Área de preparación (staging area): es un sencillo archivo, generalmente contenido en el directorio de Git, que almacena información acerca de lo que va a ir en la próxima confirmación. A veces también se le denomina índice.

## Objetos

En git hay 4 tipos de objetos:

- Blob: se utiliza para almacenar datos de archivos, en general es un archivo sobre archivos.
- Tree: es como un directorio que hace referencia a un conjunto de otros *trees* y/o *blobs* (por ejemplo archivos y subdirectorios).
- Commit: apunta a un determinado *tree*, marcando como era en un momento determinado. Contiene información sobre ese momento determinado, los cambios del autor desde el último commit, el commit anterior (conocido como *parent*), y otras funcionalidades. También se puede entender un commit como la modificación o el conjunto de modificaciones a uno o varios archivos

del repositorio. Otra forma de entenderlo es como el estado de uno o varios archivos del repositorio en un momento determinado.

- Tag: es una forma específica de marcar un commit. Se usa principalmente para marcar algunos commits como releases específicos o algo destacable en esas líneas.

## GitHub

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por *GitHub, Inc.* Desde enero de 2010, GitHub opera bajo el nombre de *GitHub, Inc.* En esta plataforma el código se almacena de forma pública, aunque también se puede hacer de forma privada, pagando por ello. Github facilita toda la infraestructura para trabajar en equipos distribuidos a través de una interfaz web sumamente cómoda.

Para usar GitHub primero se debe instalar el programa Git (el Fuentes y otros objetos se encuentra un link para la descarga), el cual posee un programa con interfaz de usuario (GUI) pero lo recomendable es utilizar la línea de comandos.

### Crear una clave y autenticar al usuario

Luego de instalado se debe proceder a crear una clave para autenticar al usuario que se conecta entre la computadora y Git, los pasos son los siguientes y están en terminal de comandos en Windows:

- Escribir el comando (sin comillas dobles): “ssh-keygen -t rsa -C ‘su\_correo@ejemplo.com’ (aquí ‘su\_correo@ejemplo.com’” se debe sustituir por el correo que se usó para el registro en la cuenta de GitHub)

- Esto devolverá un par de mensajes, a los cuales hay que responderles presionando la tecla Enter. El primer mensaje es para que guarde la clave en el directorio por defecto (la carpeta de usuario). El segundo mensaje simplemente es para aceptar el passphrase
- Hecho esto mostrará un mensaje indicando la dirección del directorio donde fue guardada la clave
- Se debe copiar la clave ssh. En Windows se debe ir a la GUI de Git (llamada Git Extensions), en la pestaña Help (Ayuda) - Show Key (Mostrar Clave), aquí se presiona Copy to clipboard (Copiar al portapapeles) para copiar la clave ssh al portapapeles.
- Ir a la página de GitHub y hacer click en “Add another public key” y pegar la clave pegada en el portapapeles en la parte donde dice “Key” y se le indica un título para identificación
- Para comprobar lo hecho hasta ahora se debe introducir el comando (sin comillas): “ssh -T git@github.com” luego de esto se mostrará un mensaje indicando el establecimiento de la conexión, al cual hay que responderle ingresando la palabra “yes”
- Finalmente mostrará el mensaje de autenticación exitosa

Luego de haber hecho esto se debe ingresar la cuenta del usuario con los siguientes comandos:

```
git config --global user.name "Nombre Apellido"
```

```
git config --global user.email "su_correo@ejemplo.com"
```

El primer comando es para establecer el nombre de la cuenta que se está usando (nombre de usuario) y el segundo es el correo correspondiente a la cuenta de GitHub.



Ahora se mostrarán algunos comandos básicos para la utilización de la plataforma, tales como crear repositorios o subir archivos a la web GitHub

## Crear un repositorio

Primero se debe crear un repositorio o proyecto en GitHub, para esto hay que ir a la página web y seleccionar “Start a project” . Aquí se indicaría el nombre que va a tener el proyecto, una descripción del mismo (opcional) y seleccionar la visibilidad del proyecto entre pública (gratuita y visible para todo el mundo) y privada (paga y accesible solo para el usuario que lo crea y los contribuidores). Finalmente se selecciona “Create repository”.

Lo enviará a una página donde se muestran los primeros para crear y subir proyectos locales al proyecto remoto, además de mostrar la dirección del proyecto (por https o ssh) a la cual hay que pushear (subir) los archivos.

Una vez creado el repositorio en Github se debe crear el proyecto local si no se ha hecho y luego situar el directorio del mismo en la consola. Para navegar por consola hasta el directorio del proyecto se debe usar el comando:

```
cd direcciondelproyecto
```

Habiendo llegado al directorio del proyecto se debe iniciar el repositorio git y el repositorio remoto con los comandos:

```
git init  
git remote add origin direccionhttps/ssh
```

El primer comando crea un directorio oculto que contiene los datos sobre la conexión remota con GitHub, y el segundo comando agrega la dirección del

proyecto https o ssh creado en GitHub (indicada anteriormente) en donde se va a subir el proyecto local.

## Clonar un repositorio

La clonación de un proyecto es básicamente hacer una copia de un repositorio Git existente, por ejemplo un proyecto en el que se está interesado(a) a contribuir. Primero se debe llegar al directorio local donde se quiere clonar el repositorio (mediante comandos cd), luego de haberse posicionado, el comando para clonar es el siguiente:

```
git clone direccionhttps
```

## Seguimiento de nuevos archivos

Para realizar el seguimiento de nuevos archivos es necesario estar posicionado en el directorio del proyecto local y utilizar el comando:

```
git add
```

En este comando se puede indicar el nombre del archivo específico, por ejemplo: git add README, o bien todos los archivos del directorio, lo cual se puede hacer de dos formas: git add . o la otra forma: git add --all

Esto identifica los archivos que no se han subido al repositorio, para hacerlo en una posterior actualización.

## Añadir un commit

Cuando se tiene todo listo para realizar los cambios en el repositorio de GitHub, se procede a realizar un commit. El commit prepara la actualización e indica en un mensaje el cambio que se le hizo a esta versión. El comando es:

`git commit -m "mensaje de actualización"`

## Subir cambios

Finalmente, cuando se realizaron todos los cambios al proyecto y se estableció correctamente la conexión con el repositorio remoto, solo queda subir los archivos del proyecto al mismo. Esto se hace con el comando:

`git push origin master`

Esto sube los cambios del proyecto local al repositorio GitHub según se haya indicado en el master. Si se desea crear una rama (branch) y posicionarse en ella, se puede realizar con los comandos:

`git branch miNuevoBranch`  
`git checkout miNuevoBranch`

Aunque estos también se pueden resumir en uno solo:

`git checkout -b miNuevoBranch`

Si se desea subir los cambios del proyecto al branch, simplemente se toma el comando dado anteriormente y se cambia el nombre master por el nombre del branch, por ejemplo:

`git push origin miNuevoBranch`

Al hacer estos pasos, el repositorio remoto tendrá una copia de los cambios que se hicieron localmente al proyecto.

## Conclusiones del trabajo

### Problemas encontrados

Los problemas encontrados a la hora de realizar este proyecto, fueron los siguientes:

- Cómo enlazar SQL Server con una aplicación
- Qué lenguaje usar para crear la aplicación.
- Cómo indicarle a SQL qué ejecutar para obtener la información.
- Qué tipo de conexión se va realizar
- En caso de usar un lenguaje de programación del que no tengamos conocimiento previo, cómo aprender lo necesario para hacer la interfaz y crear la aplicación en sí.

### Soluciones a los problemas

Las soluciones implementadas fueron las siguientes:

1. Se realizó una investigación de cómo enlazar SQL Server con cualquier lenguaje de programación, nuestra fuente de ayuda fue la página que maneja Microsoft, la cual explicaba cómo enlazar SQL Server con algunos lenguajes, tales como C++, C#, Visual Basic, entre otros.
2. Con ayuda del punto anterior, más la ayuda de un ingeniero en sistemas, nos instó a utilizar el lenguaje de programación conocido como C#. Esto nos benefició, debido a que en la página de Microsoft, tiene la particularidad de que explicaba paso a paso cómo realizar la conexión y, además de eso, cómo realizar un comando, mandar una instrucción en lenguaje SQL, obtener datos de la bases, nombres de tablas, etc.
3. En el tema de crear la aplicación, gracias a los conocimientos de uno de los integrantes, la elaboración de la interfaz fue sencilla y rápida, dado el caso, sí fue necesario investigar un poco sobre la funcionalidad del lenguaje, sin

embargo, no llevo mucha dificultad ya que los integrantes en general teníamos conocimiento sobre el paradigma de POO, el cual es utilizado por C#.

4. Para resolver el problema de la tarea programada en general, realizamos un plan que consistía en dividir el problema en 3 partes: conexión, validaciones e interfaz. Esto ayudó en la elaboración del programa, ya que disminuyó considerablemente la dificultad.
5. Con respecto a la conexión, se configuró para que nuestro programa pueda realizar la conexión, ya sea con los permisos del sistema operativo o bien los permisos del DBMS que utilizamos, en nuestro caso SQL Server.
6. También para realizar algunos métodos en la conexión, fue necesario realizar preguntas en internet de como realizarlas, la principal fuente de información, es el sitio web conocido como Stack OverFlow.

## Aprendizajes obtenidos

Dentro de los temas aprendidos, podemos mencionar los siguientes:

- ❖ Aprendimos a conectar una aplicación con un DBMS.
- ❖ Se aprendió sobre un lenguaje de programación, el cual fue C#.
- ❖ Se descubrió que si un problema es muy grande y este se divide en subpartes, es mucho más fácil solucionarlo.
- ❖ Se entendió porqué una herramienta como Git, es muy útil a la hora de llevar un control de la creación del programa.

## Rúbrica de evaluación y análisis de resultados

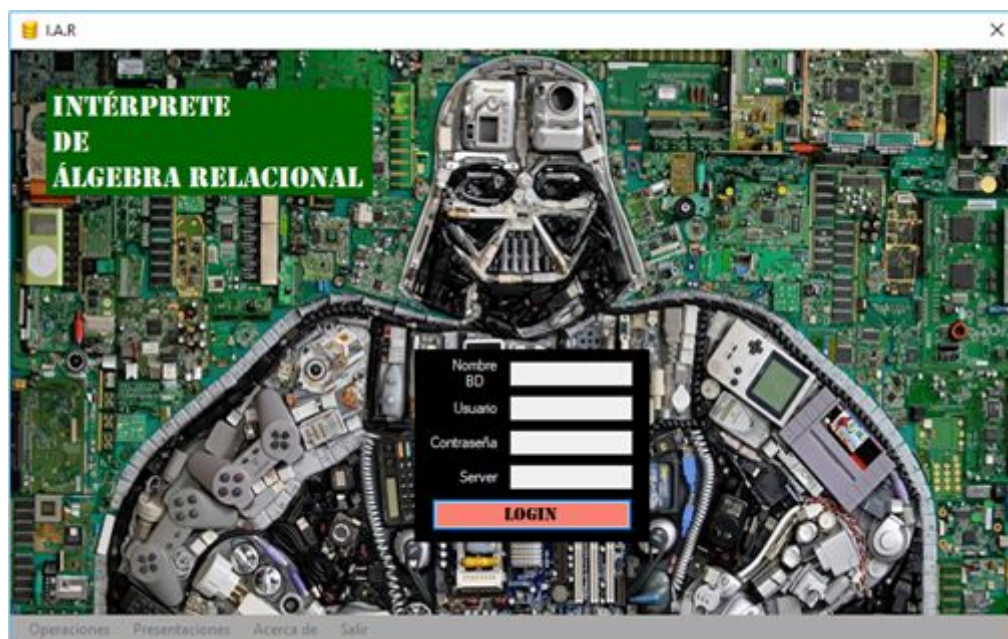
Concepto	Puntos	Puntos obtenidos	% de Avance 100/x/0	Análisis de Resultados
Resultado de selección	2		100%	
Validación	1		100%	
Resultado de proyección generalizada	2		100%	
Validación	1		100%	
Resultado de unión	2		100%	
Validación	4		100%	
Resultado diferencia de conjuntos	2		100%	
Validación	4		100%	
Resultado de producto cartesiano	2		100%	
Validación	1		100%	
Resultado de Intersección	2		100%	
Validación	4		100%	
Resultado de división	2		100%	
Validación	4		100%	
Resultado de renombrar una relación y atributos	7		100%	
Validación	7		100%	
Resultado de concatenación (join)	2		100%	

Validación	1		100%	
Resultado de concatenación natural (natural join)	5		100%	
Validación	5		100%	
Resultado de agregación	2		100%	
Validación	1		100%	
Resultado de agrupación	2		100%	
Validación	1		100%	
Resultado mostrar expresión de álgebra relacional	2		100%	
Resultado mostrar instrucción SQL	3		100%	
Manejo de la tabla de datos resultante (incluye scroll)	5		100%	
Ver tablas de la base de datos	5		100%	
Ver tablas temporales	5		100%	
Ver referencia cruzada atributos / tablas	5		100%	
Borrar tablas temporales	2		100%	
Creación de usuarios (DBA, usuario)	2		70%	Se creó el usuario dbaproy1 como propietario de la base de datos bdproy1. El usuario usproy1 no logramos concederle los permisos de lectura y CRUD por aparte, se le asignaron permisos de propietario por este inconveniente. No pudimos completarlo porque no lográbamos darle permisos de manejo de tablas para sí mismo, y lectura para las ya existentes.

Ayuda (manual de usuario)	5		100%	
Acerca de / Salir	0		100%	
<b>TOTAL</b>	<b>100</b>			
Partes desarrolladas adicionalmente				

## Manual de Usuario

### Pantalla Principal



Al iniciar el intérprete de álgebra relacional, se presentará la pantalla principal, la cual solo posee habilitadas las casillas para iniciar la conexión para realizar las operaciones de SQL, y la X para cerrar el programa.





El usuario puede escoger llenar las casillas de nombre de usuario y contraseña. Si el usuario deja en blanco estas casillas, se hará el login con la autorización de Windows. Si el usuario no escribe el nombre de la base de datos a utilizar el programa presentará una ventanilla de error, misma situación si no se escribe el nombre del servidor, o si estos datos otorgados son erróneos.

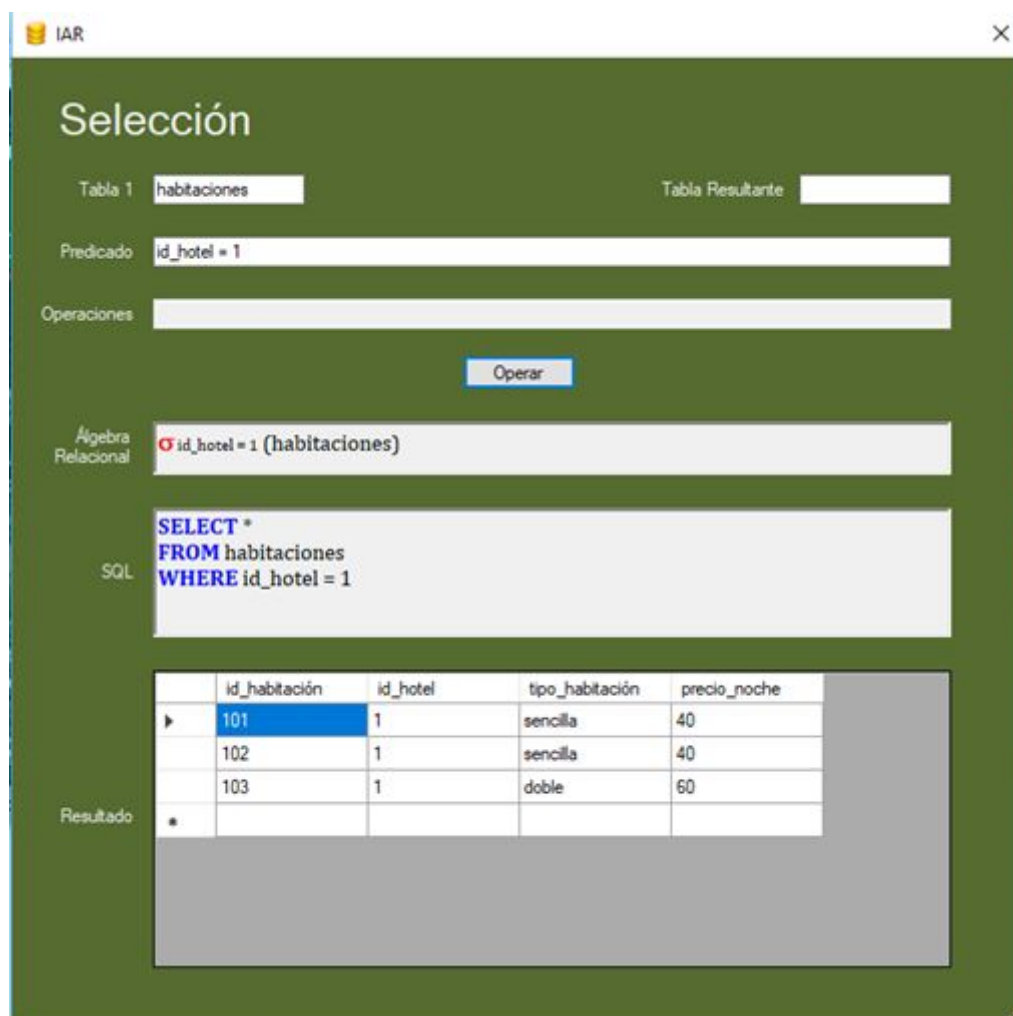


Si el inicio de sesión es correcto se habilitarán las opciones del programa, donde se presentan casillas con los nombres:

1. Operaciones
  - a. Selección
  - b. Proyección Generalizada
  - c. Unión
  - d. Diferencia de Conjuntos
  - e. Producto Cartesiano
  - f. Intersección
  - g. División
  - h. Renombramiento
  - i. Join
  - j. Natural Join
  - k. Agregación
  - l. Agrupación
2. Presentaciones
  - a. Tablas Originales
  - b. Tablas Resultantes
  - c. Atributos
3. Acerca de
4. Salir

## Operaciones

Es el conjunto de operaciones de álgebra relacional elegibles a realizar. Al elegir cualquiera de las opciones se desplegará una ventana de la siguiente manera:



**Selección**

Tabla 1:  Tabla Resultante:

Predicado:

Operaciones:

Algebra Relacional:

SQL:

Resultado:

	id_habitación	id_hotel	tipo_habitación	precio_noche
▶	101	1	sencilla	40
	102	1	sencilla	40
	103	1	doble	60
*				

Cada una de las operaciones a realizar posee una ventana con la forma que se muestra en la imagen anterior, la cual posee habilitados los espacios que corresponden a cada operación. Tomando el ejemplo de la imagen anterior, la selección ocupa que se escriba el nombre de la tabla a utilizar y el predicado de la operación, en caso de que no se rellenen estos espacios obligatorios para la operación la misma no se realizará, también ocurrirá esto si la información proporcionada es inválida, como por ejemplo colocar el nombre de una tabla que no

existe. El espacio de tabla resultante no es obligatorio, en caso de colocar un nombre en esta casilla, este pertenecerá al nombre de la tabla temporal que se creará y contendrá la información producida por la operación. En este caso la información se mostraría de la siguiente manera:

The screenshot shows the IAR software interface with a green background. The title bar says 'IAR'. The main window is titled 'Selección'. It contains several input fields and a button:

- Tabla 1:**
- Tabla Resultante:**
- Predicado:**
- Operaciones:**
- Operar:** A blue button.
- Algebra Relacional:**
- SQL:**
- Resultado:** A table with 5 columns: id\_habitación, id\_hotel, tipo\_habitación, precio\_noche, and a selection icon. The first three rows are highlighted in blue.

	id_habitación	id_hotel	tipo_habitación	precio_noche
▶	101	1	sencilla	40
	102	1	sencilla	40
	103	1	doble	60
*				

La información producida por la operación ahora se encuentra dentro de la tabla llamada ejemplo, la cual podrá ser utilizada para futuras operaciones mientras no se cierre totalmente el programa. Al cerrarse el programa las tablas temporales serán eliminadas de la base de datos.

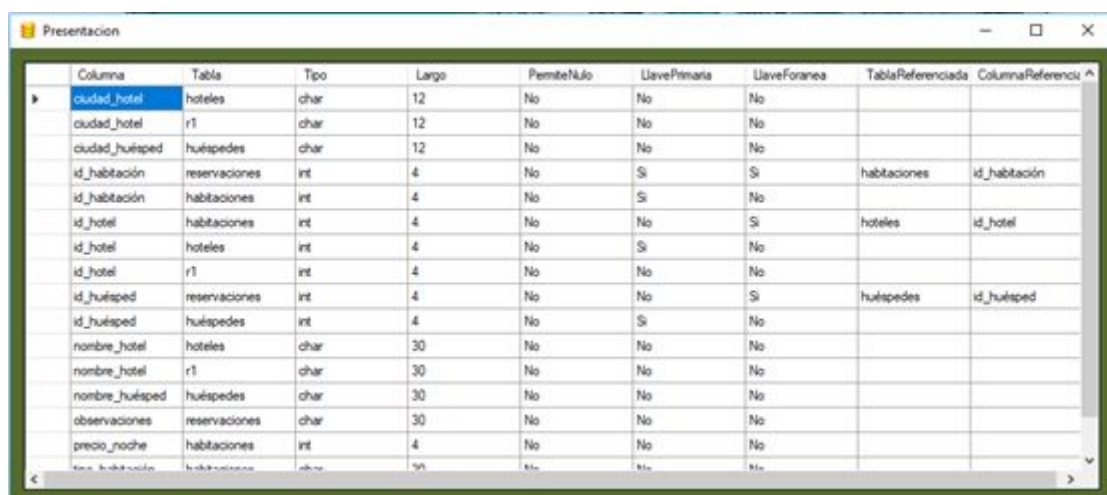
## Presentaciones

Las opciones Tablas Originales y Tablas Resultantes, presentan la información de sus respectivas tablas y atributos, también se presenta la información que corresponde a cada atributo, si es llave primaria, foránea, su dominio, entre otros. Y se presentan de esta manera:



Tabla	Columna	Tipo	Largo	PermiteNulo	LlavePrimaria	LlaveForanea	TablaReferenciada	ColumnaReferenciada
habitaciones	id_habitación	int	4	No	Si	No		
habitaciones	id_hotel	int	4	No	No	Si	hoteles	id_hotel
habitaciones	tipo_habitación	char	20	No	No	No		
habitaciones	precio_noche	int	4	No	No	No		
hoteles	id_hotel	int	4	No	Si	No		
hoteles	nombre_hotel	char	30	No	No	No		
hoteles	ciudad_hotel	char	12	No	No	No		
huéspedes	id_huésped	int	4	No	Si	No		
huéspedes	nombre_huésped	char	30	No	No	No		
huéspedes	ciudad_huésped	char	12	No	No	No		
r1	id_hotel	int	4	No	No	No		
r1	nombre_hotel	char	30	No	No	No		
r1	ciudad_hotel	char	12	No	No	No		
reservaciones	id_habitación	int	4	No	Si	Si	habitaciones	id_habitación
reservaciones	id_huésped	int	4	No	No	Si	huéspedes	id_huésped

En el caso de la opción Atributos, esta nos presentará los atributos, organizados por nombre, con sus dominios, y demás características, además de las tablas en las cuales se encuentran. Y se presenta en una tabla de la siguiente forma:

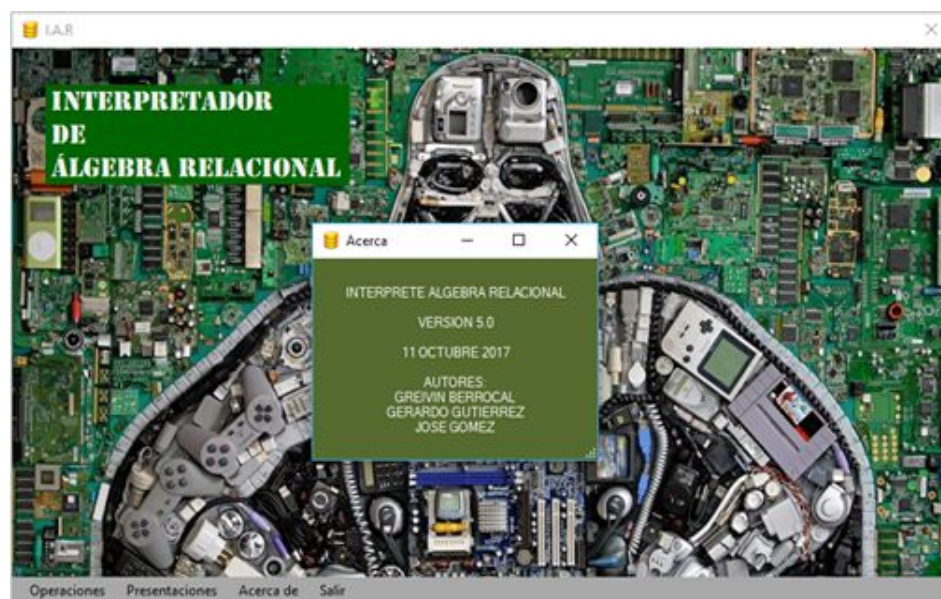


Columna	Tabla	Tipo	Largo	PermiteNulo	LlavePrimaria	LlaveForanea	TablaReferenciada	ColumnaReferenciada
ciudad_hotel	hoteles	char	12	No	No	No		
ciudad_hotel	r1	char	12	No	No	No		
ciudad_huésped	huéspedes	char	12	No	No	No		
id_habitación	reservaciones	int	4	No	Si	Si	habitaciones	id_habitación
id_habitación	habitaciones	int	4	No	Si	No		
id_hotel	habitaciones	int	4	No	No	Si	hoteles	id_hotel
id_hotel	hoteles	int	4	No	Si	No		
id_hotel	r1	int	4	No	No	No		
id_huésped	reservaciones	int	4	No	No	Si	huéspedes	id_huésped
id_huésped	huéspedes	int	4	No	Si	No		
nombre_hotel	hoteles	char	30	No	No	No		
nombre_hotel	r1	char	30	No	No	No		
nombre_huésped	huéspedes	char	30	No	No	No		
observaciones	reservaciones	char	30	No	No	No		
precio_noche	habitaciones	int	4	No	No	No		



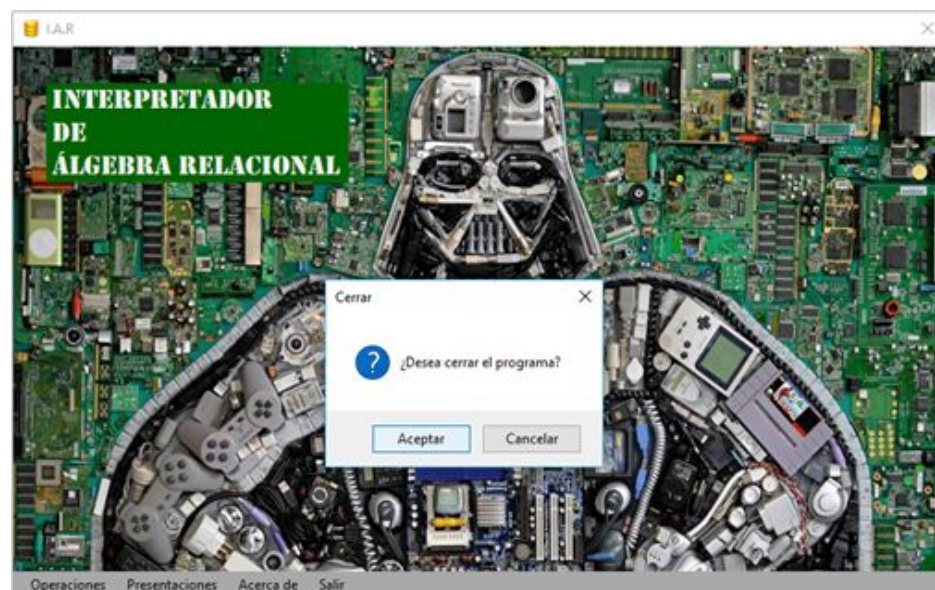
## Acerca de

En esta ventanilla se presenta la información general acerca de la aplicación. Se muestra el nombre, la versión, fecha de creación, y autores.



## Salir

Al igual que al presionar la X de la esquina superior derecha, aparecerá una ventana consultando si desea cerrar el programa, si presiona Aceptar el programa internamente eliminará las tablas resultantes producidas y se cerrará.



## Fuentes y otros objetos

Para la ejecución del programa, como ya se ha mencionado anteriormente, utilizamos el lenguaje de programación C# junto con el lenguaje SQL.

Es necesario tener instalado en la computadora el programa Visual Studio 2015 o una versión posterior para la ejecución de la interfaz de usuario y lógica del programa, y con respecto al lenguaje de consultas a la base de datos se debe tener el motor de base de datos, nosotros utilizamos SQL Server en su versión Express 2014.

No es necesaria la utilización de software adicional a los programas especificados en esta sección.

## Fuentes

En el caso de C#, se utilizó las siguientes páginas web:

C# guide. Retrieved from

<https://docs.microsoft.com/en-us/dotnet/csharp/>

<http://csharp.net-informations.com/data-providers/csharp-sql-server-connection.htm>

<https://social.msdn.microsoft.com/Forums/es-ES/df4fb680-cfff-419f-b86a-eb498509a36e/conectar-a-base-sqlserver-con-c-con-una-clase-general?forum=vcses>

En el caso de SQL, se utilizó la siguiente página web:

SQL guide. Retrieved from

<https://stackoverflow.com/questions/>

En el caso de Git y GitHub, se utilizaron diferentes capítulos de la siguiente página web:

Git guide. Retrieved from

<https://git-scm.com/book/es/v1/Empezando>