

Actividad | 3 | Ejecución

Lenguaje de Programación II

Ingeniería en Desarrollo de
Software



TUTOR: Félix Acosta Hernández

ALUMNO: Sarahi Jaqueline Gómez Juárez, sara_2mil@outlook.com

FECHA: Domingo, 11 de agosto de 2024.

Índice

Índice.....	2
Introducción:	5
Descripción:	8
Justificación:.....	13
Desarrollo:	15
Tablas:.....	15
<i>Visualización de los datos de las Tablas: "EMPLEADO", "Directivo" y</i>	
<i>"Centro de Trabajo".</i>	<i>15</i>
<i>Tabla la tabla Respuesta de si es o no Directivo (Combinación de las tablas):</i>	<i>17</i>
Código:	21
<i>"Configuración de la Consola para Soporte de Codificación UTF-8 y Manejo</i>	
<i>de ODBC en un Programa C++"</i>	<i>22</i>
<i>"Conexión a Base de Datos SQL Server y Visualización de Encabezados de la</i>	
<i>Tabla "EMPLEADO" en un Programa C++":.....</i>	<i>23</i>
<i>"Recuperación y Organización de Datos de la Tabla "EMPLEADO" en un</i>	
<i>Programa C++ usando ODBC":.....</i>	<i>24</i>
<i>Código de manejo y consulta SQL para tabla 'EMPLEADO' con verificación de</i>	
<i>errores en C++:</i>	<i>24</i>
<i>"Código de consulta SQL y visualización de datos para la tabla 'Directivo' en</i>	
<i>C++"(parte 1):.....</i>	<i>25</i>
<i>"Código de consulta SQL y visualización de datos para la tabla 'Directivo' en</i>	
<i>C++"(parte 2):.....</i>	<i>25</i>

<i>Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla “Centro_de_Trabajo”(parte 1):.....</i>	<i>26</i>
<i>Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla “Centro_de_Trabajo” (parte 2):.....</i>	<i>27</i>
<i>Código de manejo y consulta SQL para tabla “Centro_de_Trabajo” con verificación de errores en C++ (parte 1):.....</i>	<i>27</i>
<i>Código de manejo y consulta SQL para tabla “Centro_de_Trabajo” con verificación de errores en C++ (parte 2):.....</i>	<i>28</i>
<i>Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla “Respuesta de si es o no Directivo” (parte 1):.....</i>	<i>28</i>
<i>Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 2):</i>	<i>29</i>
<i>Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 3):</i>	<i>29</i>
<i>Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 4):</i>	<i>29</i>
<i>Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 5):</i>	<i>30</i>
<i>Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo y el manejo de errores” (parte 6):.....</i>	<i>30</i>
<i>Código en C++ para la liberación de la Última Consulta y Desconexión de la Base de Datos:.....</i>	<i>31</i>
Enlaces de las diferentes etapas del Programa en C++:.....	31

Conclusión: 32

Referencias: 34

Introducción:

Este proyecto documenta la edición del código en C++ llevada a cabo durante la segunda etapa, centrada en la consulta y visualización de datos provenientes de varias tablas de una Base de Datos SQL Server, inicialmente, se detalla la sintaxis utilizada para extraer y mostrar la información contenida en las tablas **“EMPLEADO”**, **“Directivo”** y **“Centro_de_Trabajo”**, finalmente, se combinan los datos de estas tablas para determinar si un empleado tiene un rol directivo, mostrando los resultados en la tabla **“Respuesta de si es o no Directivo”**, adicionalmente, el programa configura la consola para utilizar UTF-8, lo que garantiza una visualización correcta de caracteres especiales, y asegura la liberación adecuada de los recursos ODBC al concluir la ejecución.

Durante la tercera etapa, descrita en este documento, se ha refinado el formato para la visualización de datos, aplicando conceptos fundamentales de Programación Orientada a Objetos (POO) y herencia en C++ para establecer relaciones entre las distintas tablas y mostrar la información de manera más estructurada.

Cambios Implementados:

Modificación en la columna “NOMBRE COMPLETO”:

Se ha ajustado la consulta SQL para incluir una columna que concatena el nombre, el apellido paterno y el apellido materno del empleado.

Ajuste de Encabezados y Eliminación de Columnas:

Se modificaron los encabezados de la tabla y se eliminaron las columnas **#Puesto y Prest.**

Combustible, optimizando la presentación de la información.

Adición de la columna ES_DIRECTIVO:

Se incorporó una nueva columna para mostrar si un empleado es directivo, implementada mediante una expresión **CASE** en la consulta SQL que verifica la presencia del empleado en la tabla **“Directivo”**.

Manejo de Valores Nulos con COALESCE:

Se utilizó la función **COALESCE** para manejar valores nulos en **“Prestacion_de_Combustible”**, garantizando que se muestre un valor de 0 cuando no hay un dato definido.

Generación de una Tabla Combinada:

Se implementó una consulta combinada que muestra información sobre si un empleado es directivo, junto con otros datos relacionados.

Mejora en la Presentación de Resultados:

Se utilizó `wcout` para imprimir los resultados de manera más clara y estructurada en la consola, empleando `<< L"|" << endl;` para mejorar la separación y presentación de las filas de datos.

Comentarios Adicionales y Ajuste de Variables:

Se añadieron comentarios adicionales para mejorar la comprensión y el mantenimiento del código. Las variables se ajustaron para manejar la nueva estructura de datos, realizando cambios en tamaño utilizando `setw` y en tipo para reflejar las nuevas columnas y los datos concatenados.

Para implementar estas mejoras, se han utilizado diversas librerías estándar de C++ como `<iostream>`, `<windows.h>`, `<sql.h>`, `<sqlext.h>`, `<iomanip>`, `<locale>`, `<codecvt>`, y `<string>`, asegurando una interacción eficaz con la Base de Datos SQL Server y una presentación clara y estructurada de los datos extraídos.

El código en C++ ha sido diseñado meticulosamente para interactuar con una Base de Datos SQL Server, garantizando una extracción y presentación eficiente de la información contenida en múltiples tablas, la estructura del programa no solo facilita la consulta y el

despliegue de datos, sino que también asegura la correcta gestión de los recursos, con especial atención a la configuración de la consola para UTF-8, permitiendo la visualización precisa de caracteres especiales.

La implementación de la Programación Orientada a Objetos (POO) en C++ y la utilización de herencia en este proyecto han sido fundamentales para establecer relaciones entre las diferentes tablas de la base de datos, esto no solo optimiza el código, sino que también mejora su mantenimiento y escalabilidad, permitiendo futuras extensiones y modificaciones con mayor facilidad.

A lo largo de las etapas del desarrollo, se ha mantenido un enfoque en la claridad del código, tanto en su lógica como en su presentación, los comentarios detallados y las variables cuidadosamente nombradas contribuyen a un código comprensible y manejable, incluso para otros desarrolladores que puedan involucrarse en el futuro.

Además, se analizará la relevancia de adquirir este conocimiento en el ámbito cotidiano y profesional.

Descripción:

El presente documento describe un programa escrito en C++ que utiliza una estructura de clases para la gestión de los trabajadores de la empresa UNI, este programa realiza diversas consultas SQL y está configurado para conectarse con una base de datos SQL Server, diseñada en la primera etapa del proyecto.

El principal objetivo de la tercera etapa es la obtención y presentación de los datos necesarios, para lo cual se han realizado varios cambios en la sintaxis del código, iniciados en la segunda etapa, estos cambios incluyen:

Modificaciones en la Consulta Combinada de la Tabla “Respuesta de si es o no Directivo”:

Columna Nombre Completo: Se ha utilizado la función **CONCAT** para combinar las columnas **NOMBRE**, **APELLIDO_PATERNO** y **APELLIDO_MATERNO** en una nueva columna denominada **NOMBRE_COMPLETO**.

Presentación de Datos: Los datos se muestran en formato tabular, sin alterar el resto del código.

Columna Fecha de nacimiento: Se ha añadido esta columna en la tercera posición de la tabla, el encabezado de la tabla y el manejo de datos en el bucle que procesa y muestra los resultados han sido actualizados en consecuencia.

Modificaciones en la Consulta SQL:

Columna Nombre Completo: La consulta SQL ha sido ajustada para incluir la columna **NOMBRE_COMPLETO**, que concatena el nombre, apellido paterno y apellido materno del empleado.

Eliminación y Ajuste de Encabezados: Se han eliminado las **columnas #Puesto y Prest. Combustible**, optimizando los encabezados de la tabla para una presentación más concisa.

Columna ES_DIRECTIVO: Se ha añadido la columna **ES_DIRECTIVO**, que indica si un empleado ocupa un cargo directivo, esto se ha logrado mediante una expresión **CASE** en la consulta SQL, que verifica la presencia del empleado en la tabla Directivo.

Manejo de Valores Nulos: Para la columna **Prestacion_de_Combustible**, se ha utilizado la función **COALESCE**, garantizando que se muestre un valor de 0 en caso de ausencia de datos.

Consulta Combinada: Se ha implementado una consulta combinada que genera una tabla que integra información sobre la condición de directivo de los empleados junto con sus demás datos.

Mejoras en la Presentación de Datos:

Uso de wcout: Para mejorar la claridad en la salida de los datos, se ha utilizado **wcout**, lo que permite una impresión más estructurada y legible en formato tabular.

Separación de Filas: Se ha empleado la expresión `<< L"|" << endl;` para mejorar la separación entre filas, haciendo que el formato de la consola sea más accesible y ordenado.

Actualización de Encabezados: Los encabezados de la tabla “**Respuesta de si es o no Directivo**” se han actualizado para reflejar las nuevas columnas.

Comentarios y Mantenimiento:

Documentación del Código: Se han añadido comentarios detallados a lo largo del código, facilitando su comprensión y mantenimiento futuro.

Ampliación de Variables: Las variables han sido ajustadas en tamaño y tipo para

acomodar la nueva estructura de datos, incluyendo las columnas adicionales y los datos concatenados.

El código implementa principios de Programación Orientada a Objetos (POO), utilizando herencia para estructurar el programa de manera eficiente.

Este programa en C++ está diseñado para interactuar de manera efectiva con una base de datos SQL Server, extrayendo y presentando información de múltiples tablas de manera estructurada y legible en la consola, lo que facilita el análisis y la revisión de datos.

Se incluyen varias librerías estándar de C++ (iostream, iomanip, locale, codecvt, string) y librerías específicas de Windows (windows.h) y ODBC (sql.h, sqllex.h).

La función SetConsoleToUtf8() establece la consola para usar codificación UTF-8, lo que permite manejar caracteres especiales correctamente.

Para la conexión a la Base de Datos:

Se asignan manejadores para el entorno **ODBC (hEnv)** y la conexión (hDbc) y se establece la conexión a la base de datos SQL Server mediante SQLConnect, antes de mostrar los datos, se imprimen los encabezados de las tablas en la consola, se utiliza setw() para alinear correctamente las columnas, después se utilizan funciones como SQLFetch y SQLGetData para recuperar los datos de cada fila de las tablas consultadas, los datos se almacenan en variables específicas, que luego se imprimen en la consola.

En la última parte del código, se realiza una consulta combinada que une datos de las tablas “**EMPLEADO**”, “**Directivo**”, y “**Centro_de_Trabajo**”, se utiliza **CONCAT** para crear una columna **Nombre_completo** que combina el nombre y los apellidos, y un **CASE** para verificar si el empleado es directivo, después de cada consulta, se liberan los manejadores de consulta (hStmt) utilizando SQLFreeHandle.

Detalles Técnicos:

Manejo de Codificación: Se utiliza `wstring_convert` para manejar la conversión entre UTF-8 y UTF-16.

Orden de Salida: Se emplean manipuladores como `setw()` para controlar el formato de salida en la consola, garantizando que las columnas estén bien alineadas.

Consulta SQL Compleja: La consulta combinada es un ejemplo de cómo se pueden unir múltiples tablas y transformar los datos dentro de una consulta SQL, incluyendo la concatenación de columnas y el uso de **COALESCE** y **CASE**.

Utilización de diferentes bibliotecas:

<iostream>: Proporciona funcionalidades de entrada y salida estándar en C++, como **std::cout** para la salida de datos a la consola y **std::cin** para la entrada desde la consola.

<windows.h>: Es un encabezado que incluye muchas de las API de Windows, se utiliza para desarrollar aplicaciones que interactúan directamente con el sistema operativo Windows.

<sql.h> y <sqlext.h>: Son encabezados de **ODBC (Open Database Connectivity)** que contienen definiciones y funciones necesarias para conectar y ejecutar consultas en bases de datos. **sql.h** es el encabezado principal, y **sqlext.h** proporciona extensiones adicionales.

<iomanip>: Contiene manipuladores de entrada/salida que permiten formatear la salida, como la configuración de la precisión decimal, la anchura del campo, la alineación, etc.

<locale>: Proporciona funciones y clases para gestionar las configuraciones regionales (locales) que afectan a la forma en que los datos son procesados y presentados, como el formato de números y fechas.

<codecvt>: Contiene la clase **std::codecvt** que se utiliza para la conversión entre diferentes conjuntos de caracteres, por ejemplo, de UTF-8 a UTF-16.

<string>: Proporciona la clase **std::string** para manejar cadenas de texto de una manera más cómoda y segura que con los arreglos de caracteres (**char**).

Este código es un ejemplo robusto de cómo integrar consultas SQL en un programa C++ para interactuar con una base de datos SQL Server, manejando tanto la conexión como la presentación de los datos de manera eficiente y clara.

Justificación:

El objetivo de este proyecto radica en la necesidad de desarrollar y optimizar herramientas que faciliten la conexión y gestión eficiente de bases de datos SQL mediante el uso de lenguajes de programación avanzados como C++, la implementación de consultas SQL dentro de un entorno de desarrollo como Visual Studio, utilizando ODBC para conectarse a un servidor SQL, es fundamental debido a la importancia de manejar grandes volúmenes de datos de manera segura y eficiente.

El proyecto se enfoca en la integración de datos provenientes de múltiples tablas para extraer información clave, como la identificación de empleados en cargos directivos, este tipo de análisis de datos es crucial para la toma de decisiones en cualquier organización, ya que permite una comprensión más profunda de la estructura jerárquica y la asignación de recursos.

Una de las funcionalidades implementadas es la concatenación de datos relevantes, como nombre, apellido paterno y apellido materno, en una sola columna, lo que mejora la presentación y facilita la interpretación de la información.

La mejora del código en C++ para ejecutar estas tareas no solo incrementa la eficiencia y rapidez de las consultas, sino que también asegura la integridad y consistencia de los datos, en este sentido, el proyecto tiene un valor académico significativo al aplicar conocimientos avanzados de programación y bases de datos, y también posee una relevancia práctica considerable al ofrecer soluciones reales a problemas complejos de gestión de datos en un contexto empresarial.

El desarrollo del programa se lleva a cabo en Visual Studio Community, un entorno de desarrollo integrado que facilita la gestión y depuración del código, la conexión a la Base de

Datos SQL Server mediante ODBC garantiza la interoperabilidad y permite manejar grandes volúmenes de datos de forma eficiente y segura.

El proyecto incluye la integración de datos de diferentes tablas para obtener resultados precisos y relevantes, como la creación de una tabla que identifica los empleados en cargos directivos marcándolos con un Si y los que no lo son con un No, este enfoque no solo optimiza el rendimiento de las consultas, sino que también asegura la claridad y precisión en la presentación de los datos, aspectos esenciales en el análisis de la estructura organizativa.

Otro componente esencial de este proyecto es el manejo de versiones, una práctica fundamental en el desarrollo de software que asegura la integridad y evolución del código a lo largo del tiempo, a través de sistemas de control de versiones, se facilita la colaboración entre diferentes desarrolladores, se permite el seguimiento detallado de cambios, y se asegura la capacidad de revertir a versiones anteriores en caso de errores o conflictos, esto no solo mejora la eficiencia del desarrollo, sino que también garantiza que el código sea mantenible y escalable a medida que el proyecto crece y evoluciona.

En conclusión, este proyecto no solo representa una aplicación práctica de conceptos avanzados en programación y bases de datos, sino que también responde a necesidades reales en el ámbito empresarial, al integrar el manejo de versiones, optimizar el rendimiento y garantizar la precisión en el análisis de datos, el proyecto contribuye significativamente al desarrollo de soluciones tecnológicas que fortalecen la competitividad y éxito de las organizaciones.

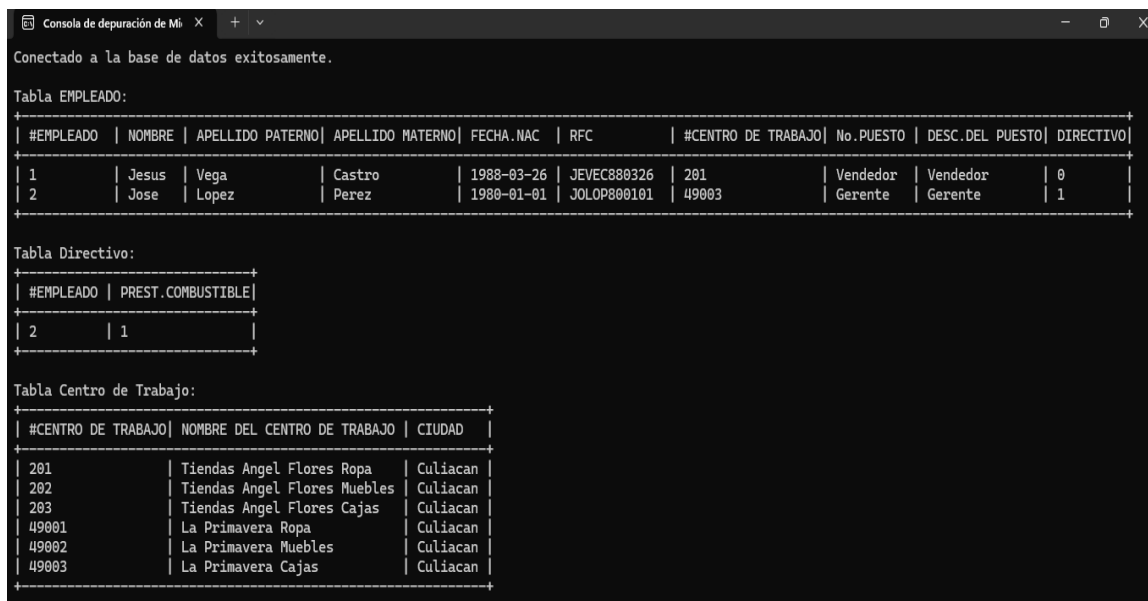
Desarrollo:

Tablas:

Se muestran las Tablas: “Empleados”, “Directivos” y “Centros de Trabajo” de la base de datos SQL Server, y la tabla “**Respuesta si es o no Directivo**”, en esta última se indica si un empleado es directivo o no, estos resultados se muestran en las figuras 1 y 2:

Figura 1

Visualización de los datos de las Tablas: "EMPLEADO", "Directivo" y "Centro de Trabajo".



Consola de depuración de Mi X + -

Conectado a la base de datos exitosamente.

Tabla EMPLEADO:

#EMPLEADO	NOMBRE	APELLIDO PATERNO	APELLIDO MATERNO	FECHA.NAC	RFC	#CENTRO DE TRABAJO	No.PUESTO	DESC.DEL PUESTO	DIRECTIVO
1	Jesus	Vega	Castro	1988-03-26	JEVEC880326	201	Vendedor	Vendedor	0
2	Jose	Lopez	Perez	1980-01-01	JOLOP800101	49003	Gerente	Gerente	1

Tabla Directivo:

#EMPLEADO	PREST.COMBUSTIBLE
2	1

Tabla Centro de Trabajo:

#CENTRO DE TRABAJO	NOMBRE DEL CENTRO DE TRABAJO	CIUDAD
201	Tiendas Angel Flores Ropa	Culiacan
202	Tiendas Angel Flores Muebles	Culiacan
203	Tiendas Angel Flores Cajas	Culiacan
49001	La Primavera Ropa	Culiacan
49002	La Primavera Muebles	Culiacan
49003	La Primavera Cajas	Culiacan

Nota: La siguiente imagen muestra las tablas (esto ya se había explicado en la segunda etapa): **Tabla EMPLEADO:** Esta tabla contiene información básica sobre los empleados, incluyendo su nombre, apellidos, fecha de nacimiento, RFC, centro de trabajo, número y descripción del puesto, y si el empleado es directivo: se escribirá con 1 y si no es directivo se registrará con 0, esta columna para indicar si el empleado es directivo, almacenando información detallada sobre los empleados de la empresa.

Tabla Directivo: Almacena datos específicos relacionados con empleados que tienen un rol directivo, incluye el número de empleado y beneficios adicionales (como la prestación de combustible), registrar información específica sobre los empleados que tienen algún beneficio de combustible, esta tabla parece estar vinculada a la tabla de empleados mediante el número de empleado y se enfoca en aquellos que tienen un beneficio adicional relacionado con combustible.

Tabla Centro de Trabajo: Registra los detalles de los centros de trabajo donde están asignados los empleados, esto incluye el identificador del centro de trabajo, su nombre y la ciudad en la que se encuentra, contener los datos sobre los diferentes centros de trabajo dentro de la empresa, como el nombre del centro y la ciudad en la que se encuentra, esta tabla sirve para identificar y relacionar a los empleados con sus respectivos lugares de trabajo.

Estas tablas están conectadas entre sí para proporcionar una visión integral de la organización, combinando datos de empleados, sus roles, y los lugares donde trabajan, estas tablas están diseñadas para manejar la información esencial de los empleados, los beneficios que reciben y la localización de sus lugares de trabajo, lo que facilita la gestión y consulta de datos en la empresa, confirmando que se a conectado correctamente la base de datos diseñada en la etapa inicial del programa.

Figura 2

Tabla la tabla Respuesta de si es o no Directivo (Combinación de las tablas):

Tabla Respuesta de si es o no Directivo:

#EMPLEADO	NOMBRE COMPLETO	FECHA.NAC	RFC	CENTRO DE TRABAJO	DESC.PUESTO	DIRECTIVO
1	Jesus Vega Castro	1988-03-26	JEVEC880326	Tiendas Angel Flores Ropa	Vendedor	No
2	Jose Lopez Perez	1980-01-01	JOLOP800101	La Primavera Cajas	Gerente	Si

C:\Users\sara_\source\repos\ConexionBDSQLSERVER-Lenguajes de Programacion 2\x64\Debug\ConexionBDSQLSERVER-Lenguajes de Programacion 2.exe (proceso 35948) se cerró con el código 0.
 Para cerrar automáticamente la consola cuando se detiene la depuración, habilite Herramientas ->Opciones ->Depuración ->Cerrar la consola automáticamente al detenerse la depuración.
 Presione cualquier tecla para cerrar esta ventana. . |

Nota: Esta consulta constituye una síntesis de datos obtenidos mediante la combinación y unificación de varias tablas fundamentales dentro del sistema de gestión de la base de datos, cada una de estas tablas desempeña un papel crucial en la organización, y su integración permite una visión holística y detallada de la información relativa a los empleados, sus puestos, beneficios y los centros de trabajo en los que operan:

Tabla EMPLEADO:

Esta tabla constituye el núcleo central de la base de datos, donde se almacena información fundamental sobre los empleados de la organización, cada registro en la tabla representa a un individuo específico, detallado mediante los siguientes campos:

ID_EMPLEADO: Un identificador único asignado a cada empleado, que permite su reconocimiento inequívoco dentro del sistema.

NOMBRE: El nombre de pila del empleado, que junto con los apellidos conforma su

identidad dentro de la base de datos.

APELLIDO PATERNO y APELLIDO MATERNO: Los apellidos del empleado, que, en conjunto con el nombre, forman la identificación completa del mismo.

FECHA_NAC: La fecha de nacimiento del empleado, un dato esencial tanto para la administración de recursos humanos como para el cumplimiento de regulaciones laborales.

RFC: El Registro Federal de Contribuyentes, un identificador fiscal esencial en México, que se utiliza para fines tributarios y administrativos.

#CENTRO DE TRABAJO: Un código numérico que relaciona al empleado con el centro de trabajo al cual está asignado, permitiendo un seguimiento de su ubicación laboral.

No. PUESTO: El número que identifica el puesto específico que ocupa el empleado dentro de la estructura organizativa.

DESC. DEL PUESTO: Una descripción textual que detalla las responsabilidades y el rol del empleado en su posición actual.

DIRECTIVO: Un indicador binario que señala si el empleado ocupa una posición directiva dentro de la empresa, donde 0 indica que no lo es y 1 que sí lo es.

Tabla Directivo:

Esta tabla está especializada en gestionar información adicional para aquellos empleados que ocupan posiciones de dirección. Incluye:

ID_EMPLEADO: Referencia al identificador único del empleado, conectando esta tabla directamente con la tabla principal de empleados.

PREST. COMBUSTIBLE: Un campo que indica si el directivo tiene acceso a beneficios específicos, como el suministro de combustible, reflejando aspectos de compensación y beneficios extrasalariales.

Tabla Centro de Trabajo:

Aquí se documentan los diferentes centros de trabajo de la organización, proporcionando una visión clara de la estructura geográfica y funcional de la empresa. Cada centro de trabajo está descrito mediante los siguientes campos:

#CENTRO DE TRABAJO: Un identificador numérico único para cada centro de trabajo, permitiendo su distinción dentro de la red de ubicaciones de la empresa.

NOMBRE DEL CENTRO DE TRABAJO: Una descripción nominal que denota el nombre del centro de trabajo, facilitando su identificación.

CIUDAD: La ciudad donde se ubica el centro de trabajo, lo que permite un análisis geográfico de la distribución de la fuerza laboral.

Tabla Respuesta de si es o no Directivo:

Esta tabla es una vista combinada que sintetiza la información más relevante de los empleados, especialmente enfocada en su estatus directivo, a través de un diseño que integra múltiples dimensiones de la información, presenta:

ID_EMPLEADO: El identificador del empleado, asegurando la unicidad del registro.

NOMBRE COMPLETO: El nombre completo del empleado, compuesto por el nombre y los apellidos, presentado de manera concatenada para una lectura más natural.

FECHA_NAC: La fecha de nacimiento del empleado, esencial para el seguimiento del historial personal y laboral.

RFC: El Registro Federal de Contribuyentes, manteniendo la trazabilidad fiscal del empleado.

CENTRO DE TRABAJO: El nombre del centro de trabajo en el que está asignado el empleado, brindando contexto a su entorno laboral.

DESC. PUESTO: Una descripción detallada del rol del empleado dentro de la organización, aportando una visión clara de sus responsabilidades.

DIRECTIVO: Un campo que indica con precisión si el empleado ocupa o no una posición directiva, presentado de manera afirmativa (Sí) o negativa (No), según corresponda.

Esta descripción busca no solo exponer los elementos técnicos de cada tabla, sino también contextualizar su importancia dentro del sistema, destacando su papel en la administración y operación de la organización.

Código:

En esta **tercera etapa del proyecto**, se presenta el código revisado y perfeccionado, tal como se anticipó en la segunda etapa, el programa: Desarrollado en **C++**, se ilustra desde las figuras 3 hasta la 20, este código establece una conexión con una base de datos **SQL Server** mediante **ODBC** y ejecuta una serie de consultas SQL, los resultados se organizan para desplegar datos provenientes de las tablas “**EMPLEADO**”, “**Directivo**” y “**Centro_de_Trabajo**”, posteriormente, el programa integra esta información para determinar si un empleado ostenta una posición directiva, reflejando los resultados en la tabla “Respuesta si es o no Directivo”.

El código en C++ ha sido meticulosamente diseñado para extraer y presentar la información de manera estructurada y elegante en la consola, lo que facilita un análisis minucioso y una revisión exhaustiva de los datos:

Figura 3

"Configuración de la Consola para Soporte de Codificación UTF-8 y Manejo de ODBC en un Programa C++"

```

1  // ConexiónBDSQLSERVER-Lenguajes de Programacion 2.cpp : Este archivo contiene la función "main". La ejecución del programa comienza y termina ahí.
2  //
3
4  #include <iostream> // Librería para funciones de Windows, como la configuración de la consola.
5  #include <windows.h> // Librería para funcionalidades de Windows
6  #include <sql.h> // Librería principal de ODBC para funciones y estructuras
7  #include <sqlext.h> // Librería de extensiones de ODBC para funciones adicionales
8  #include <iomanip> // Librería para manipulación de formatos de entrada y salida
9  #include <locale> // Librería para manejo de locales
10 #include <codecvt> // Librería para conversiones entre diferentes codificaciones de caracteres
11 #include <string> // Librería para manejar cadenas de texto
12
13 using namespace std; // Usar el espacio de nombres estándar para simplificar el uso de clases y funciones comunes.
14
15 // Configura la consola para que use la codificación UTF-8
16 void SetConsoleToUtf8() {
17     SetConsoleOutputCP(CP_UTF8); // Establece la página de códigos de salida de la consola a UTF-8.
18     SetConsoleCP(CP_UTF8); // Establece la página de códigos de entrada de la consola a UTF-8.
19 }
20
21 wstring_convert<codecvt_utf8_utf16<wchar_t>> converter; // Convertidor para cadenas UTF-8 a UTF-16
22
23 int main() {
24     // Configurar la consola para UTF-8
25     SetConsoleToUtf8();
26
27     SQLHENV hEnv; // Manejador de entorno ODBC
28     SQLHDBC hDbc; // Manejador de conexión ODBC
29     SQLRETURN ret; // Variable para almacenar el resultado de las funciones ODBC
30
31     // Asignar un gestor de entorno
32     ret = SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &hEnv);
33     ret = SQLSetEnvAttr(hEnv, SQL_ATTR_ODBC_VERSION, (SQLPOINTER)SQL_OV_ODBC3, 0); // Establece el entorno de ODBC a la versión 3.0

```

Nota: La imagen resalta la sintaxis en C++, que refleja el contenido y propósito del código que se muestra en la captura de pantalla, destacando tanto la configuración de la consola para UTF-8 como la inicialización de los componentes de ODBC.

Figura 4

"Conexión a Base de Datos SQL Server y Visualización de Encabezados de la Tabla "EMPLEADO" en un Programa C++":

```

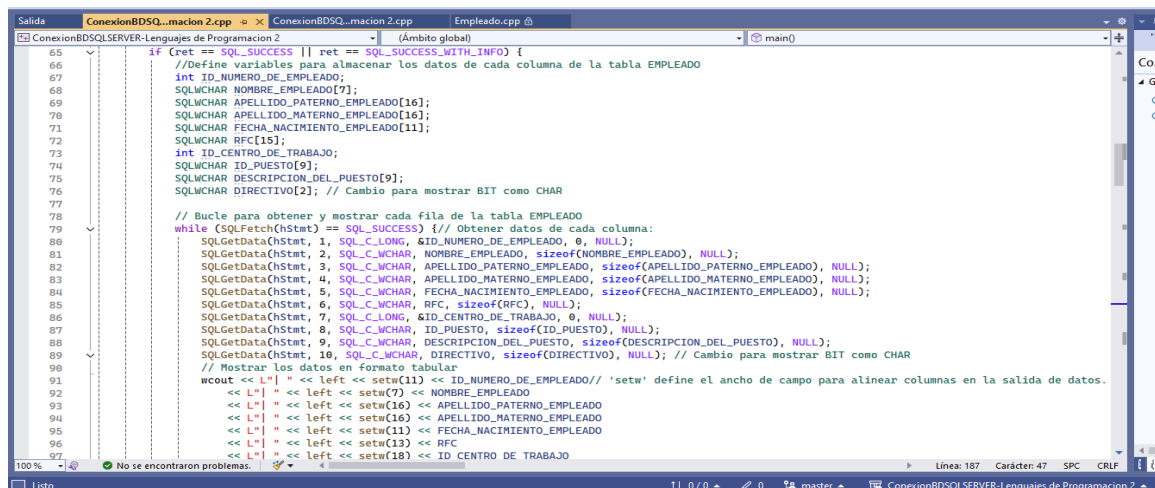
33  ret = SQLSetEnvAttr(hEnv, SQL_ATTR_ODBC_VERSION, (SQLPOINTER)SQL_OV_ODBC3, 0); // Establece el entorno de ODBC a la versión 3.0
34
35  // Asignar un gestor de conexión
36  ret = SQLAllocHandle(SQL_HANDLE_DBC, hEnv, &hdbc);
37
38  // Conectarse a la base de datos SQL Server con los parámetros proporcionados(de ser necesario se colocara nombre de servidor, usuario y contraseña)
39  ret = SQLConnect(hdbc, (SQLWCHAR*)"sqlserver", SQL_NTS, (SQLWCHAR*)"username", SQL_NTS, (SQLWCHAR*)"password", SQL_NTS);
40  // Verificar si la conexión fue exitosa
41  if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {
42      wcout << L"Conectado a la base de datos exitosamente." << endl; // Conexión exitosa, se muestra un mensaje
43
44      // Consulta y mostrar datos de la tabla EMPLEADO
45      SQLHSTMT hstmt; // Crear un manejador de consulta
46      ret = SQLAllocHandle(SQL_HANDLE_STMT, hdbc, &hstmt);
47
48      // Mostrar encabezado de la tabla EMPLEADO
49      wcout << endl << L"Tabla EMPLEADO:" << endl;
50      wcout << L"-----";
51      wcout << L" | " << left << setw(11) << L"#EMPLEADO"
52      << L" | " << left << setw(7) << L"NOMBRE"
53      << L" | " << left << setw(16) << L"APELLIDO PATERNO"
54      << L" | " << left << setw(16) << L"APELLIDO MATERNO"
55      << L" | " << left << setw(11) << L"FECHA.NAC"
56      << L" | " << left << setw(13) << L"RFC"
57      << L" | " << left << setw(16) << L"#CENTRO DE TRABAJO"
58      << L" | " << left << setw(10) << L"No.PUESTO"
59      << L" | " << left << setw(8) << L"DESC. DEL PUESTO"
60      << L" | " << left << setw(5) << L"DIRECTIVO" << L" | ";
61      wcout << L"-----";
62
63      // Consulta SELECT para EMPLEADO(seleccionar todos los datos de la tabla EMPLEADO)
64      ret = SQLExecDirect(hstmt, (SQLWCHAR*)"SELECT * FROM EMPLEADO", SQL_NTS);
65      if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {

```

Nota: La imagen resalta la sintaxis en C++, que destaca la conexión a la base de datos y la preparación para mostrar los encabezados de la tabla “EMPLEADO”, que se observa en el código.

Figura 5

"Recuperación y Organización de Datos de la Tabla "EMPLEADO" en un Programa C++ usando ODBC":



```

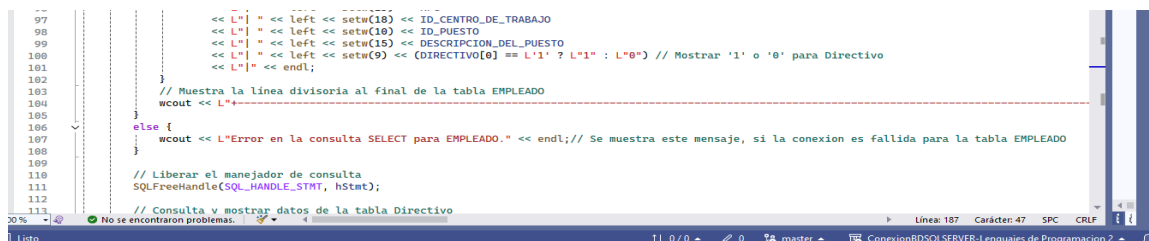
65 if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {
66     //Define variables para almacenar los datos de cada columna de la tabla EMPLEADO
67     int ID_NUMERO_DE_EMPLEADO;
68     SQLWCHAR NOMBRE_EMPLEADO[7];
69     SQLWCHAR APELLIDO_PATERNO_EMPLEADO[16];
70     SQLWCHAR APELLIDO_MATERNO_EMPLEADO[16];
71     SQLWCHAR FECHA_NACIMIENTO_EMPLEADO[11];
72     SQLWCHAR RFC[15];
73     int ID_CENTRO_DE_TRABAJO;
74     SQLWCHAR ID_PUESTO[9];
75     SQLWCHAR DESCRIPCION_DEL_PUESTO[9];
76     SQLWCHAR DIRECTIVO[2]; // Cambio para mostrar BIT como CHAR
77
78     // Bucle para obtener y mostrar cada fila de la tabla EMPLEADO
79     while (SQLFetch(hstmt) == SQL_SUCCESS) { // Obtener datos de cada columna:
80         SQLGetData(hstmt, 1, SQL_C_LONG, &ID_NUMERO_DE_EMPLEADO, 0, NULL);
81         SQLGetData(hstmt, 2, SQL_C_WCHAR, NOMBRE_EMPLEADO, sizeof(NOMBRE_EMPLEADO), NULL);
82         SQLGetData(hstmt, 3, SQL_C_WCHAR, APELLIDO_PATERNO_EMPLEADO, sizeof(APELLIDO_PATERNO_EMPLEADO), NULL);
83         SQLGetData(hstmt, 4, SQL_C_WCHAR, APELLIDO_MATERNO_EMPLEADO, sizeof(APELLIDO_MATERNO_EMPLEADO), NULL);
84         SQLGetData(hstmt, 5, SQL_C_WCHAR, FECHA_NACIMIENTO_EMPLEADO, sizeof(FECHA_NACIMIENTO_EMPLEADO), NULL);
85         SQLGetData(hstmt, 6, SQL_C_WCHAR, RFC, sizeof(RFC), NULL);
86         SQLGetData(hstmt, 7, SQL_C_LONG, &ID_CENTRO_DE_TRABAJO, 0, NULL);
87         SQLGetData(hstmt, 8, SQL_C_WCHAR, ID_PUESTO, sizeof(ID_PUESTO), NULL);
88         SQLGetData(hstmt, 9, SQL_C_WCHAR, DESCRIPCION_DEL_PUESTO, sizeof(DESCRIPCION_DEL_PUESTO), NULL);
89         SQLGetData(hstmt, 10, SQL_C_WCHAR, DIRECTIVO, sizeof(DIRECTIVO), NULL); // Cambio para mostrar BIT como CHAR
90         // Mostrar los datos en formato tabular
91         wcout << L" | " << left << setw(11) << ID_NUMERO_DE_EMPLEADO // 'setw' define el ancho de campo para alinear columnas en la salida de datos.
92         << L" | " << left << setw(7) << NOMBRE_EMPLEADO
93         << L" | " << left << setw(16) << APELLIDO_PATERNO_EMPLEADO
94         << L" | " << left << setw(16) << APELLIDO_MATERNO_EMPLEADO
95         << L" | " << left << setw(11) << FECHA_NACIMIENTO_EMPLEADO
96         << L" | " << left << setw(13) << RFC
97         << L" | " << left << setw(18) << ID_CENTRO_DE_TRABAJO

```

Nota: La imagen resalta la sintaxis en C++, para la acción de recuperar los datos de la tabla "EMPLEADO" y organizarlos para su salida en la consola, lo que se refleja en el código.

Figura 6

Código de manejo y consulta SQL para tabla 'EMPLEADO' con verificación de errores en C++:



```

97 << L" | " << left << setw(18) << ID_CENTRO_DE_TRABAJO
98 << L" | " << left << setw(10) << ID_PUESTO
99 << L" | " << left << setw(15) << DESCRIPCION_DEL_PUESTO
100 << L" | " << left << setw(9) << [DIRECTIVO[0] == '1' ? L"1" : L"0"] // Mostrar '1' o '0' para Directivo
101 << L" | " << endl;
102
103 // Muestra la línea divisoria al final de la tabla EMPLEADO
104 wcout << L"-----" << endl;
105
106 }
107 else {
108     wcout << L"Error en la consulta SELECT para EMPLEADO." << endl; // Se muestra este mensaje, si la conexión es fallida para la tabla EMPLEADO
109 }
110
111 // Liberar el manejador de consulta
112 SQLFreeHandle(SQL_HANDLE_STMT, hstmt);
113
114 // Consulta y mostrar datos de la tabla Directivo

```

Nota: La imagen refleja la muestra un fragmento de código en C++ que se encarga de realizar una consulta a la tabla 'EMPLEADO', manejar la consulta y verificar si ocurre algún error durante la ejecución del comando SQL.

Figura 7

"Código de consulta SQL y visualización de datos para la tabla 'Directivo' en C++"(parte 1):

```

113 // Consulta y mostrar datos de la tabla Directivo
114 ret = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Directivo y el resto de tablas
115
116 // Mostrar encabezado de la tabla Directivo
117 wcout << endl << L"Tabla Directivo:" << endl;
118 wcout << L"-----" << endl;
119 wcout << L"|" << left << setw(10) << L"#EMPLEADO" // Es crucial contabilizar los espacios contenidos dentro de los paréntesis para garantizar un al
120 << L"|" << left << setw(15) << L"PREST.COMBUSTIBLE"
121 << L"|" << endl;
122 wcout << L"-----" << endl;
123
124 // Consulta SELECT para Directivo
125 ret = SQLExecDirect(hStmt, (SQLWCHAR*)"SELECT * FROM Directivo", SQL_NTS);
126 if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {
127 // Declaración de las variables para almacenar los datos de la tabla Directivo:
128 int ID_Numero_de_Empleado;
129 SQLWCHAR Prestacion_de_Combustible[2]; // Cambio para mostrar BIT como CHAR
130 // Bucle para obtener y mostrar cada fila de la tabla Directivo:
131 while (SQLFetch(hStmt) == SQL_SUCCESS) { // Obtener y mostrar cada fila de la tabla Directivo
132     SQLGetData(hStmt, 1, SQL_C_LONG, &ID_Numero_de_Empleado, 0, NULL);
133     SQLGetData(hStmt, 2, SQL_C_WCHAR, Prestacion_de_Combustible, sizeof(Prestacion_de_Combustible), NULL); // Cambio para mostrar BIT como CHAR
134     // Mostrar los datos en formato tabular
135     wcout << L"|" << left << setw(10) << ID_Numero_de_Empleado
136     << L"|" << left << setw(17) << (Prestacion_de_Combustible[0] == L'1' ? L'1' : L'0') // Mostrar '1' o '0' para Prestacion_de_Combustible
137     << L"|" << endl;
138 }
139 // Muestra la línea divisoria al final de la tabla Directivo
140 wcout << L"-----" << endl; // Es fundamental contar las líneas divisoras para asegurar un alineado preciso en todos l
141
142 else {
143     wcout << L"Error en la consulta SELECT para Directivo." << endl; // Se muestra este mensaje, si la conexión es fallida para la tabla Directivo
144 }
145
146

```

Nota: La imagen muestra un fragmento de código en C++, que se encarga de realizar una consulta a la tabla “**Directivo**”, obtener y mostrar los datos en un formato tabular, y manejar posibles errores en la ejecución del comando SQL.

Figura 8

"Código de consulta SQL y visualización de datos para la tabla 'Directivo' en C++"(parte 2):

```

113 // Se repite el proceso para la tabla Directivo y el resto de tablas
114
115
116
117
118
119
120
121
122
123
124
125
126 // la Directivo:
127
128 // como CHAR
129
130 // fila de la tabla Directivo
131
132 // sizeof(Prestacion_de_Combustible), NULL); // Cambio para mostrar BIT como CHAR
133
134 // [0] == L'1' ? L'1' : L'0') // Mostrar '1' o '0' para Prestacion_de_Combustible
135
136
137
138
139
140 // tal contar las líneas divisoras para asegurar un alineado preciso en todos los campos donde se aplican.
141
142 // Se muestra este mensaje, si la conexión es fallida para la tabla Directivo
143
144
145

```

Figura 9

Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla

“Centro_de_Trabajo”(parte 1):

```

144     }
145
146     // Liberar el manejador de consulta
147     SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
148
149     // Consulta y mostrar datos de la tabla Centro_de_Trabajo
150     ret = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Directivo
151
152     // Mostrar encabezado de la tabla Centro_de_Trabajo
153     wcout << endl << L"Tabla Centro de Trabajo:" << endl;
154     wcout << L"-----" << endl; // 'wcout' imprime caracteres anchos, y 'L' indica literales de
155     wcout << L" | " << left << setw(13) << L"#CENTRO DE TRABAJO"
156     << L" | " << left << setw(29) << L"NOMBRE DEL CENTRO DE TRABAJO"
157     << L" | " << left << setw(9) << L"CIUDAD"
158     << L" | " << endl;
159     wcout << L"-----" << endl;
160
161     // Consulta SELECT para Centro_de_Trabajo
162     ret = SQLExecDirect(hStmt, (SQLWCHAR*)L"SELECT * FROM Centro_de_Trabajo", SQL_NTS); // Se repite el proceso para la tabla Centro_de_Trabajo
163     if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {
164         // Declaracion de las variables para almacenar los datos de la tabla Centro_de_Trabajo :
165         int ID_Centro_de_trabajo;
166         SQLWCHAR Nombre_del_Centro_de_trabajo[29];
167         SQLWCHAR CIUDAD[16];
168         // Bucle para obtener y mostrar cada fila de la tabla Centro_de_Trabajo:
169         while (SQLFetch(hStmt) == SQL_SUCCESS) {
170             SQLGetData(hStmt, 1, SQL_C_LONG, &ID_Centro_de_trabajo, 0, NULL);
171             SQLGetData(hStmt, 2, SQL_C_WCHAR, Nombre_del_Centro_de_trabajo, sizeof(Nombre_del_Centro_de_trabajo), NULL);
172             SQLGetData(hStmt, 3, SQL_C_WCHAR, CIUDAD, sizeof(CIUDAD), NULL);
173             // Mostrar los datos en formato tabular
174             wcout << L" | " << left << setw(18) << ID_Centro_de_trabajo
175             << L" | " << left << setw(29) << Nombre_del_Centro_de_trabajo
176             << L" | " << left << setw(9) << CIUDAD

```

Nota: La imagen muestra un fragmento de código en C++, que se enfoca en la funcionalidad principal del código: realizar una consulta SQL y mostrar tanto los encabezados como los datos de la tabla “**Centro_de_Trabajo**” en un programa C++.

Figura 10

Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla

“Centro_de_Trabajo” (parte 2):

```

1444
1445
1446     de consulta
1447     E_STMT, hStmt);
1448
1449     atos de la tabla Centro_de_Trabajo
1450     L_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Directivo
1451
1452     e la tabla Centro_de_Trabajo
1453     a Centro de Trabajo:" << endl;
1454
1455     setw(13) << L"#CENTRO DE TRABAJO" << endl; // 'wcout' imprime caracteres anchos, y 'L' indica literales de caracteres anchos.
1456     setw(20) << L"NOMBRE DEL CENTRO DE TRABAJO"
1457     setw(9) << L"CIUDAD"
1458
1459     -----+<< endl;
1460
1461     Centro_de_Trabajo
1462     ret, (SQLWCHAR*)L"SELECT * FROM Centro_de_Trabajo", SQL_NTS); // Se repite el proceso para la tabla Centro_de_Trabajo
1463     if (ret == SQL_SUCCESS_WITH_INFO) {
1464         as variables para almacenar los datos de la tabla Centro_de_Trabajo :
1465         abajo;
1466         _Centro_de_trabajo[29];
1467
1468         er y mostrar cada fila de la tabla Centro_de_Trabajo:
1469         while (ret == SQL_SUCCESS) {
1470             t, 1, SQL_C_LONG, &ID_Centro_de_trabajo, 0, NULL);
1471             t, 2, SQL_C_WCHAR, Nombre_del_Centro_de_trabajo, sizeof(Nombre_del_Centro_de_trabajo), NULL);
1472             t, 3, SQL_C_WCHAR, CIUDAD, sizeof(CIUDAD), NULL);
1473             datos en formato tabular
1474             << left << setw(18) << ID_Centro_de_trabajo
1475             left << setw(29) << Nombre_del_Centro_de_trabajo
1476             left << setw(9) << CIUDAD
1477
1478             -----+<< endl;
1479
1480             // Muestra la línea divisoria al final de la tabla Centro_de_Trabajo
1481             wcout << L"+-----+<< endl;
1482
1483             else {
1484                 wcout << L"Error en la consulta SELECT para Centro_de_Trabajo." << endl; // Se muestra este mensaje, si la conexión es fallida para la tabla Cer
1485             }
1486
1487             // Liberar el manejador de consulta
1488             SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
1489
1490             // Consulta y mostrar datos de la tabla Respuesta de si es o no Directivo:
1491             ret = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Respuesta de si es o no Directivo(se repite el proceso p

```

Figura 11

Figura 12 Código de manejo y consulta SQL para tabla “Centro_de_Trabajo” con verificación de errores en C++ (parte 7):

Código de manejo y consulta SQL para tabla “Centro_de_Trabajo” con verificación de errores en C++ (parte 1):

```

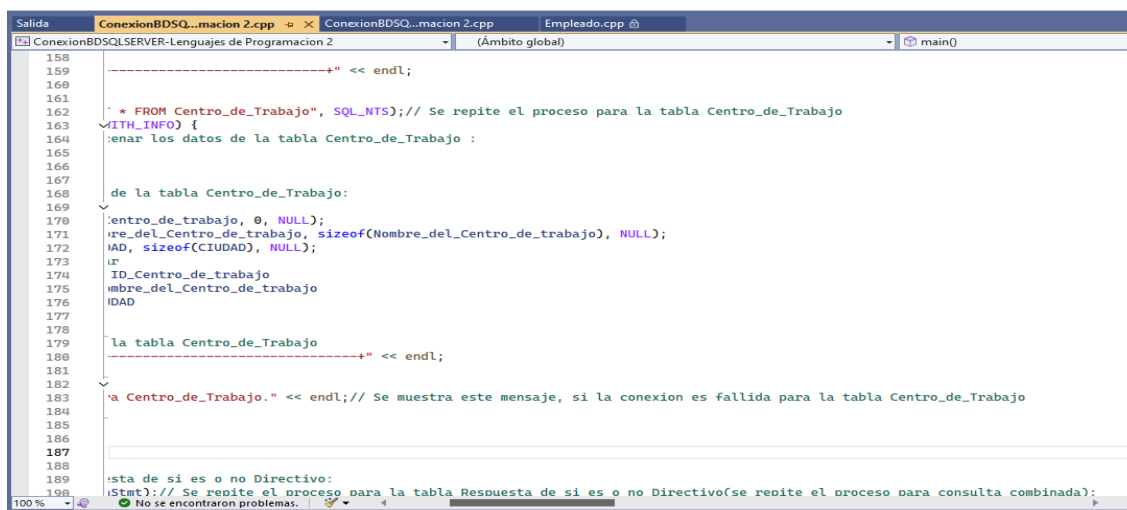
173
174
175     // Mostrar los datos en formato tabular
176     wcout << L"| " << left << setw(18) << ID_Centro_de_trabajo
177     << L"| " << left << setw(29) << Nombre_del_Centro_de_trabajo
178     << L"| " << left << setw(9) << CIUDAD
179     << L"| " << endl;
180
181     // Muestra la línea divisoria al final de la tabla Centro_de_Trabajo
182     wcout << L"+-----+<< endl;
183
184     else {
185         wcout << L"Error en la consulta SELECT para Centro_de_Trabajo." << endl; // Se muestra este mensaje, si la conexión es fallida para la tabla Cer
186     }
187
188     // Liberar el manejador de consulta
189     SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
190
191     // Consulta y mostrar datos de la tabla Respuesta de si es o no Directivo:
192     ret = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Respuesta de si es o no Directivo(se repite el proceso p

```

Nota: La imagen refleja la muestra un fragmento de código en C++ que se encarga de realizar una consulta a la tabla “Centro_de_Trabajo”, manejar la consulta y verificar si ocurre algún error durante la ejecución del comando SQL.

Figura 13

Código de manejo y consulta SQL para tabla “Centro_de_Trabajo” con verificación de errores en C++ (parte 2):



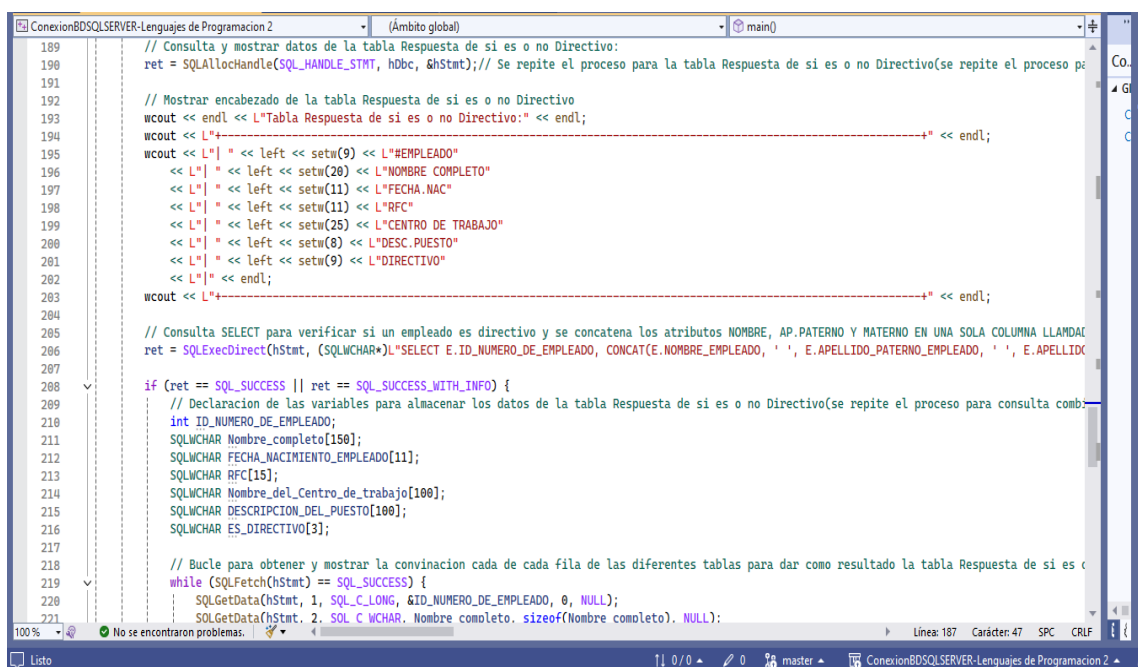
```

158
159
160
161
162 * * FROM Centro_de_Trabajo", SQL_NTS); // Se repite el proceso para la tabla Centro_de_Trabajo
163 WITH_INFO {
164 Se repite el proceso para la tabla Centro_de_Trabajo :
165
166 de la tabla Centro_de_Trabajo:
167
168
169 Centro_de_Trabajo, 0, NULL);
170 Nombre_del_Centro_de_Trabajo, sizeof(Nombre_del_Centro_de_Trabajo), NULL);
171 ID, sizeof(CIUDAD), NULL);
172 ID_Centro_de_Trabajo
173 Nombre_del_Centro_de_Trabajo
174 IDAD
175
176
177
178
179 La tabla Centro_de_Trabajo
180
181
182
183 La tabla Centro_de_Trabajo." << endl; // Se muestra este mensaje, si la conexión es fallida para la tabla Centro_de_Trabajo
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Figura 14

Código de consulta SQL en C++ para visualización de encabezados y datos de la tabla “Respuesta de si es o no Directivo” (parte 1):



```

189
190 // Consulta y mostrar datos de la tabla Respuesta de si es o no Directivo:
191 ret = SQLAllocHandle(SQL_HANDLE_STMT, hDbc, &hStmt); // Se repite el proceso para la tabla Respuesta de si es o no Directivo(se repite el proceso pa
192
193 // Mostrar encabezado de la tabla Respuesta de si es o no Directivo
194 wcout << endl << L"Tabla Respuesta de si es o no Directivo:" << endl;
195 wcout << L" " << endl;
196 wcout << L" " << left << setw(9) << L"#EMPLEADO"
197 << L" " << left << setw(20) << L"#NOMBRE COMPLETO"
198 << L" " << left << setw(11) << L"#FECHA.NAC"
199 << L" " << left << setw(11) << L"#RFC"
200 << L" " << left << setw(25) << L"#CENTRO DE TRABAJO"
201 << L" " << left << setw(8) << L"#DESC.PUESTO"
202 << L" " << left << setw(9) << L"#DIRECTIVO"
203 << L" " << endl;
204 wcout << L" " << endl;
205
206 // Consulta SELECT para verificar si un empleado es directivo y se concatena los atributos NOMBRE, AP.PATERNO Y MATERNO EN UNA SOLA COLUMNA LLAMADA
207 ret = SQLExecDirect(hStmt, (SQLWCHAR*)L"SELECT E.ID_NUMERO_DE_EMPLEADO, CONCAT(E.NOMBRE_EMPLEADO, ' ', E.APELLIDO_PATERNO_EMPLEADO, ' ', E.APELLIDO
208
209 if (ret == SQL_SUCCESS || ret == SQL_SUCCESS_WITH_INFO) {
210 // Declaracion de las variables para almacenar los datos de la tabla Respuesta de si es o no Directivo(se repite el proceso para consulta comb
211 int ID_NUMERO_DE_EMPLEADO;
212 SQLWCHAR Nombre_completo[100];
213 SQLWCHAR FECHA_NACIMIENTO_EMPLEADO[11];
214 SQLWCHAR RFC[15];
215 SQLWCHAR Nombre_del_Centro_de_Trabajo[100];
216 SQLWCHAR DESCRIPCION_DEL_PUESTO[100];
217 SQLWCHAR ES_DIRECTIVO[3];
218
219 // Bucle para obtener y mostrar la combinación de cada fila de las diferentes tablas para dar como resultado la tabla Respuesta de si es o
220 while (SQLFetch(hStmt) == SQL_SUCCESS) {
221 SQLGetData(hStmt, 1, SQL_C_LONG, &ID_NUMERO_DE_EMPLEADO, 0, NULL);
222 SQLGetData(hStmt, 2, SQL_C_WCHAR, Nombre_completo, sizeof(Nombre_completo), NULL);
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Figura 15

Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 2):

```

189
190 : repite el proceso para consulta combinada):
191
192
193
194 : endl;
195
196
197
198
199
200
201
202
203 : endl;
204
205 \ SOLA COLUMNA LLAMADA NOMBRE COMPLETO Y SE ELIMINA LA COLUMNA PREST.COMBUSTIBLE Y ID PUESTO (Se obtiene información de las tablas: Empleados, Directivo,
206 EADO, ' ', E.APELLIDO_MATERNO_EMPLEADO) AS Nombre_completo, E.FECHA_NACIMIENTO_EMPLEADO, E.RFC, C.Nombre_del_Centro_de_trabajo, E.DESCRIPCION_DEL_PUESTO,
207
208
209 io para consulta combinada):
210
211
212
213
214
215
216
217
218 \ Respuesta de si es o no Directivo
219
220
221

```

Figura 16

Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 3):

```

204
205 le las tablas: Empleados, Directivo, Centro de trabajo)
206 :trabajo, E.DESCRIPCION_DEL_PUESTO, CASE WHEN D.ID_Numero_de_Empleado IS NOT NULL THEN 'Si' ELSE 'No' END AS ES_DIRECTIVO, COALESCE(D.Prestacion_de_Combustible, 0) AS Prestacion_de_Combustible FROM EMPLEADO E LEFT JOIN DIRECTIVO D ON E.ID_NUMERO_DE_EMPLEADO = D.ID_NUMERO_DE_EMPLEADO
207

```

Figura 17

Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 4):

```

205
206 COALESCE(D.Prestacion_de_Combustible, 0) AS Prestacion_de_Combustible FROM EMPLEADO E LEFT JOIN DIRECTIVO D ON E.ID_NUMERO_DE_EMPLEADO = D.ID_NUMERO_DE_EMPLEADO
207

```

Figura 18

Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo” (parte 5):

```
205
206 NUMERO_DE_EMPLEADO = D.ID_Numero_de_Empleado JOIN Centro_de_Trabajo C ON E.ID_CENTRO_DE_TRABAJO = C.ID_Centro_de_trabajo"; SQL_NTS);
207
```

Figura 19

Código de consulta SQL en C++ para visualización de datos de la tabla “Respuesta de si es o no Directivo y el manejo de errores” (parte 6):

```

218 // Bucle para obtener y mostrar la combinacion cada de cada fila de las diferentes tablas para dar como resultado la tabla Respuesta de si es o no Directivo
219 while (SQLFetch(hStmt) == SQL_SUCCESS) {
220     SQLGetData(hStmt, 1, SQL_C_LONG, &ID_NUMERO_DE_EMPLEADO, 0, NULL);
221     SQLGetData(hStmt, 2, SQL_C_WCHAR, Nombre_completo, sizeof(Nombre_completo), NULL);
222     SQLGetData(hStmt, 3, SQL_C_WCHAR, FECHA_NACIMIENTO_EMPLEADO, sizeof(FECHA_NACIMIENTO_EMPLEADO), NULL);
223     SQLGetData(hStmt, 4, SQL_C_WCHAR, RFC, sizeof(RFC), NULL);
224     SQLGetData(hStmt, 5, SQL_C_WCHAR, Nombre_del_Centro_de_trabajo, sizeof(Nombre_del_Centro_de_trabajo), NULL);
225     SQLGetData(hStmt, 6, SQL_C_WCHAR, DESCRIPCION_DEL_PUESTO, sizeof(DESCRIPCION_DEL_PUESTO), NULL);
226     SQLGetData(hStmt, 7, SQL_C_WCHAR, ES_DIRECTIVO, sizeof(ES_DIRECTIVO), NULL);
227
228     // Mostrar los datos en formato tabular
229     wcout << L"| " << left << setw(9) << ID_NUMERO_DE_EMPLEADO
230           << L"| " << left << setw(20) << Nombre_completo
231           << L"| " << left << setw(11) << FECHA_NACIMIENTO_EMPLEADO
232           << L"| " << left << setw(10) << RFC
233           << L"| " << left << setw(25) << Nombre_del_Centro_de_trabajo
234           << L"| " << left << setw(11) << DESCRIPCION_DEL_PUESTO
235           << L"| " << left << setw(9) << ES_DIRECTIVO
236           << L"| " << endl;
237
238
239     // Muestra la línea divisoria al final de la tabla Respuesta de si es o no Directivo
240     wcout << L"+-----+" << endl;
241
242     else {
243         wcout << L"Error en la consulta SELECT para verificar si es Directivo." << endl; //Se muestra este mensaje, si la conexion es fallida para la t
244     }
245
246     // Liberar el manejador de consulta
247     SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
248
249     // Desconectarse de la base de datos
250     ret = SQLDisconnect(hDbc);
  
```

Figura 20

Código en C++ para la liberación de la Última Consulta y Desconexión de la Base de Datos:

```

245 // Liberar el manejador de consulta
246 SQLFreeHandle(SQL_HANDLE_STMT, hStmt);
247
248 // Desconectarse de la base de datos
249 ret = SQLDisconnect(hDbc);
250
251 }
252 else {
253     wcout << L"No se pudo conectar a la base de datos." << endl; //Se muestra este mensaje, si la conexion es fallida con a la base de datos SQL Server
254 }
255
256 // Liberar el manejador de conexión y el manejador de entorno
257 SQLFreeHandle(SQL_HANDLE_DBC, hDbc);
258 SQLFreeHandle(SQL_HANDLE_ENV, hEnv);
259
260 return 0; // Finaliza el programa
261 }
262

```

Enlaces de las diferentes etapas del Programa en C++:

Enlace del documento de la segunda etapa: [Sarahi Jaqueline Gomez Juarez A2.pdf - Google Drive](#)

Enlace de la etapa 3 Código C++ en Drive: [Codigo ConexionBDSQLSERVER-Lenguajes de Programacion 2-A3.cpp - Google Drive](#)

Enlace de la carpeta del Programa en C++ en GitHub: [ConexionBDSQLSERVER-Lenguajes-de-Programacion-2/ConexionBDSQLSERVER-Lenguajes de Programacion 2 at master · SarahiJaquelineGomezJuarez/ConexionBDSQLSERVER-Lenguajes-de-Programacion-2 \(github.com\)](#)

Enlace del commit realizado en la Etapa 3 Código C++ en Drive: [Los cambios aplicados en la etapa tres fueron: · SarahiJaquelineGomezJuarez/ConexionBDSQLSERVER-Lenguajes-de-Programacion-2@be63f20 \(github.com\)](#)

Enlace de la carpeta del Programa en C++ del commit realizado en la Etapa 3 en Drive: <https://drive.google.com/file/d/1Ra1nr-c5oAOTX218o2E5XbA5VGTD--Tq/view?usp=sharing>

Conclusión:

Adquirir la competencia para diseñar un programa en C++ que se conecte a una base de datos SQL Server y realice operaciones específicas, como la visualización de datos con encabezados ordenados, la concatenación de columnas y la determinación de características de los empleados, resulta de gran relevancia tanto en el ámbito profesional como en el cotidiano:

En el entorno laboral, la habilidad para desarrollar aplicaciones en C++ que interactúan eficazmente con bases de datos es crucial para muchos roles en desarrollo de software, ingeniería de sistemas y administración de bases de datos, el conocimiento en la gestión eficiente de conexiones a bases de datos y en la correcta presentación de datos contribuye a la creación de aplicaciones más rápidas y fiables, cualidades esenciales en entornos empresariales, además, la capacidad para integrar tecnologías diversas, como C++ y SQL Server, facilita la combinación de sistemas y aplicaciones que emplean distintas plataformas y tecnologías, diseñar programas capaces de extraer, procesar y mostrar datos automáticamente puede aumentar la eficiencia y reducir el trabajo manual en procesos empresariales, la destreza para identificar y corregir errores en el código que interactúa con bases de datos es esencial para el mantenimiento de sistemas y la resolución de problemas técnicos, permitiendo adaptar y modificar aplicaciones existentes para satisfacer nuevos requisitos o mejorar su funcionalidad.

En la vida cotidiana, la programación en C++ agudiza habilidades de resolución de problemas y lógica, aplicables a una amplia gama de tareas y decisiones diarias, la capacidad para mostrar y ordenar datos de manera efectiva resulta útil en diversas actividades que requieren la gestión y presentación de información, además, desarrollar programas que interactúan con bases de datos permite emprender proyectos personales o freelance, tales como aplicaciones

domésticas, análisis de datos personales o herramientas de productividad.

En resumen, el conocimiento para diseñar y desarrollar un programa en C++ que se conecte a una base de datos SQL Server no solo constituye una habilidad técnica valiosa en el ámbito profesional, sino que también ofrece beneficios significativos en la vida cotidiana al mejorar la capacidad para manejar y presentar datos de manera efectiva.

Referencias:

Chugugrace. (2023, 1 septiembre). *Conectarse a un origen de datos ODBC (Asistente para importación y exportación de SQL Server) - SQL Server Integration Services (SSIS)*.

Microsoft Learn. <https://learn.microsoft.com/es-es/sql/integration-services/import-export-data/connect-to-an-odbc-data-source-sql-server-import-and-export-wizard?view=sql-server-ver16>

CONCAT (función CONCAT) - Soporte técnico de Microsoft. (s. f.).

<https://support.microsoft.com/es-es/office/concat-funci%C3%B3n-concat-9b1a9a3f-94ff-41af-9736-694cbd6b4ca2>

El Profe Tech. (2023, 29 noviembre). *Conexion De Base De Datos Sql Server Con Visual Studio / Conectar SQL server con VISUAL BASIC* [Vídeo]. YouTube.

<https://www.youtube.com/watch?v=c3omleCdJPo>

González, J. D. M. (2020, 29 noviembre). *Librerías o bibliotecas*.

<https://www.programarya.com/Cursos/C++/Bibliotecas-o-Librerias>

Guillermo Gerard: Videítos para mi futuro yo. (2020a, julio 17). *Abstracción, encapsulamiento, herencia y polimorfismo. Ejemplo en c++ con arduino en Platformio* [Vídeo].

YouTube. <https://www.youtube.com/watch?v=h10aYATQ304>

Guillermo Gerard: Videítos para mi futuro yo. (2020b, septiembre 12). *POO y C++: Programación orientada a objetos con Platformio y Arduino: Qué es y como y por qué se usa*

[Vídeo]. YouTube. <https://www.youtube.com/watch?v=KTkXFRkZI7A>

Kathuria, H. (2022, 5 diciembre). *Cómo utilizar CASE en SQL*. LearnSQL.es.

<https://learnsql.es/blog/como-utilizar-case-en-sql/>

Mvochoa. (2014, 14 junio). *Curso de Programación C++ 3.- librerías (#Include)*

[Vídeo]. YouTube. <https://www.youtube.com/watch?v=RKEd6e0m75w>

Nivardo. (2024a, abril 11). *POO en C++*. Oregoom.com. <https://oregoom.com/cpp/poo/>

Nivardo. (2024b, abril 11). *POO en C++*. Oregoom.com. <https://oregoom.com/cpp/poo/>

Rivera, N. (2021, 11 marzo). Los beneficios psicológicos de la programación.

Hipertextual. <https://hipertextual.com/2015/09/programacion-beneficios>

Slack. (s. f.-a). *Slack*. [\[academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\]\(https://ids-academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\)
\[l_studio_2022.mkv\]\(https://ids-academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\)](https://ids-</p>
</div>
<div data-bbox=)

Slack. (s. f.-b). *Slack*. [\[academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\]\(https://ids-academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\)
\[l_studio_2022.mkv\]\(https://ids-academiaglobal.slack.com/files/U067NHBV25U/F07FNV720H3/crear_repositorio_github_visua\)](https://ids-</p>
</div>
<div data-bbox=)

Vaitkun, D. (2022, 5 diciembre). *¿Cuáles son los diferentes tipos de JOIN de SQL?*

LearnSQL.es. <https://learnsql.es/blog/cuales-son-los-diferentes-tipos-de-join-de-sql/>

Video conferencing, web conferencing, webinars, screen sharing. (s. f.-a). Zoom.

<https://academiaglobal->

[mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-)

[4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-)

[HrK_yhPeTrme_?canPlayFromShare=true&from=share_recording_detail&continueMode=true](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-HrK_yhPeTrme_?canPlayFromShare=true&from=share_recording_detail&continueMode=true)

[&componentName=rec-play&originRequestUrl=https%3A%2F%2Facademiaglobal-](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-HrK_yhPeTrme_?canPlayFromShare=true&from=share_recording_detail&continueMode=true)

[mx.zoom.us%2Frec%2Fshare%2FKiatp0kAL82TrHXR3_knB5i5NI5twXObQRBnrOgLvIA86r](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-HrK_yhPeTrme_?canPlayFromShare=true&from=share_recording_detail&continueMode=true)

[0PPHr7_EmX2jBM2c7o.aB2M6DaiQwb3Q1__](https://academiaglobal-mx.zoom.us/rec/play/QyXDsqoxCEXQ4YKudnRNdhyi5viIU451Bd3PLUaoo-4Mg179gOOSNgq7DDO8u0MpeJ1XC_7sR5pvAXRU.gs-HrK_yhPeTrme_?canPlayFromShare=true&from=share_recording_detail&continueMode=true)

Video conferencing, web conferencing, webinars, screen sharing. (s. f.-b). Zoom.

https://academiaglobal-mx.zoom.us/rec/play/vxZGq2-
 E25FkCc4LJbIT4BYxSGAfyD8pOEr87bld1oqEUOYQOwHFBCkyIp61r858Jtb3any2_Po7Nqje
 .IvmtarQph0duYkiW?canPlayFromShare=true&from=share_recording_detail&continueMode=tr
 ue&componentName=rec-play&originRequestUrl=https%3A%2F%2Facademiaglobal-
 mx.zoom.us%2Frec%2Fshare%2FFCU6n0G90oCKLuF_d-
 xusMXyIk9Ve9Zjc8vaaa0OKEfKiOsl0vM4ozDvQzXffYye.SIwJqtHUPbCjB-Q3

Video conferencing, web conferencing, webinars, screen sharing. (s. f.-c). Zoom.

https://academiaglobal-mx.zoom.us/rec/play/8olAKNX6h-6qP2VF3T-
 Qa9A6Cx4Z3kGwjxi6Rn8JLdKmWMhniw3jvnnvCSYTK58icLeohYpHgnbFMMs75.tYx7h-
 fBy_dHacUU?canPlayFromShare=true&from=share_recording_detail&continueMode=true&co
 mponentName=rec-play&originRequestUrl=https%3A%2F%2Facademiaglobal-
 mx.zoom.us%2Frec%2Fshare%2FoPZSLBFfNnRC5eUbNWJb7oIatoWhw_Br1VkODj7fL0DAj
 5WUkQ0aqql36mr0mBZ0.n9M8sIgJwgsKFWSt