UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA

FACULTAD DE INGENIERÍA EN SISTEMAS DE INFORMACIÓN

LICENCIATURA EN INGENIERIA EN SISTEMAS DE INFORMACIÓN

CURSO: 04 – Introducción a los Sistemas de Cómputo

CATEDRATICO: Ing. Wilmer Tezén

PROYECTO FINAL

NOMBRE DEL ALUMNO:	Carne No.:
Ian Gabriel Barrera Medina	7590-23-15170
Jonathan Derik Oron Aguilar	7590-23-21328
Kevin Denilson Cax Coc	7590-23-24668
Jonathan Josué Sabán Cua	7590-23-18023
Luis Estuardo Rodas Méndez	7590-23-16644

SAN JUAN SACATEPEQUEZ, 01-01-2023.

Índice

Introducción

El objetivo principal de este proyecto es proporcionar a Multiservicios CGT una herramienta eficiente y efectiva para administrar los datos relacionados con sus operaciones de catering. Esto incluye la gestión de clientes, eventos, meseros, contratos, pagos, y cualquier otra información relevante para el negocio.

Una base de datos entidad-relación (ER) se utilizará como el modelo conceptual para diseñar y estructurar la base de datos. Este modelo permite representar las entidades clave en el negocio, sus atributos y las relaciones entre ellas. Al utilizar este enfoque, se busca capturar de manera precisa y comprensible la estructura y las interacciones de los diferentes elementos involucrados en el negocio de catering de Multiservicios CGT.

Además, se ha destacado la importancia de agregar datos históricos a la base de datos. Esto implica incluir información relevante sobre eventos pasados, clientes atendidos, servicios ofrecidos y cualquier otro dato que sea valioso para el análisis, la toma de decisiones y el seguimiento de las operaciones en el tiempo.

Al implementar esta base de datos, se espera lograr una serie de beneficios para Multiservicios CGT, como una mayor eficiencia en la gestión de eventos y contratos, una mejor comprensión de los patrones y tendencias del negocio, una mayor capacidad para personalizar los servicios ofrecidos a los clientes y una mejora general en la calidad y satisfacción del servicio.

Desarrollo del proyecto

Análisis de requisitos:

La empresa Multiservicios CGT desea implementar una base de datos para el control de su negocio de catering.

Se requiere almacenar información sobre contratos, eventos, meseros y otros aspectos relevantes.

Los contratos deben incluir datos como el nombre del contratante, la fecha de contratación, el monto y el vendedor asociado.

Los eventos deben registrar información como la fecha, el tipo de evento, el cliente y el lugar.

Se necesita llevar un seguimiento de los meseros asignados a cada evento, incluyendo su nombre, fecha de contratación y salario.

Diseño del modelo entidad-relación:

El diagrama entidad-relación (DER) se construye en base a los requisitos identificados:

Entidades:

- Vendedor
- Servicio
- Evento
- Contrato
- Meseros
- Ofrece
- Asigna

Creación de las tablas:

Utilizando el diseño del ER, se crean las tablas correspondientes en la base de datos "catering" con sus respectivos campos:

```
CREATE TABLE Vendedor ( --
  id INTEGER PRIMARY KEY,
  nombre_vendedor VARCHAR(100)
);

CREATE TABLE Servicio ( ---
  id INTEGER PRIMARY KEY,
  nombre varchar(100),
  descripcion VARCHAR(200),
  costo VARCHAR(15),
  meseros_asignados INTEGER
);

CREATE TABLE Evento ( --
  id INTEGER PRIMARY KEY,
  fecha_evento VARCHAR(20),
  hora_evento VARCHAR(20),
  lugar_evento VARCHAR(100),
  tipo_evento VARCHAR(100),
  tiempo_evento VARCHAR(100)
);

CREATE TABLE Contrato ( --
  id INTEGER PRIMARY KEY,
  nombre_contratante VARCHAR(100),
  telefono_contratante VARCHAR(20),
  fecha_requerimiento VARCHAR(20),
  descripcion_opcional VARCHAR(200),
  motivo_cancelacion VARCHAR(200),
  metodo_pago VARCHAR(100),
  monto_pagado VARCHAR(15),
```

```
   estado_cumplimiento VARCHAR(50),
   id_vendedor INTEGER,
   id_evento INTEGER,
    FOREIGN KEY (id_vendedor) REFERENCES Vendedor(id),
    FOREIGN KEY (id_evento) REFERENCES Evento(id)
);

CREATE TABLE meseros (
  id INTEGER PRIMARY KEY,
  nombre VARCHAR(100),
  dpi VARCHAR(20),
  telefono_contacto VARCHAR(20),
  edad INTEGER,
  genero VARCHAR(20),
  disponibilidad_evento VARCHAR(100)
);


CREATE TABLE ofrece (
    id_servicio INTEGER,
    id_contrato INTEGER,
    FOREIGN KEY(id_servicio) REFERENCES Servicio(id),
    FOREIGN KEY (id_contrato) REFERENCES Contrato(id)
);

CREATE TABLE asigna (
    id_evento INTEGER,
    id_meseros INTEGER,
    FOREIGN KEY (id_evento) REFERENCES Evento(id),
    FOREIGN KEY (id_meseros) REFERENCES meseros(id)
);
```

Carga de datos históricos:

Utilizando las instrucciones "INSERT" y “VALUES” , se agregarán los datos históricos en las tablas correspondientes:

INSERT INTO Contrato (id, nombre_contratante, telefono_contratante, fecha_requerimiento, descripcion_opcional, motivo_cancelacion, metodo_pago, monto_pagado, estado_cumplimiento, id_vendedor, id_evento)

VALUES (1, 'Pedro Daniel', '42468971', '15 de Enero 2022','Cliente solicita el paquete P2', 'NULL', 'Tarjeta MasterCard', '16000','Completado', 2, 1);

INSERT INTO Servicio (id, nombre, descripcion, costo, meseros_asignados)

VALUES (1, 'P1','Primer paquete de servicios', '25,000', 25);

INSERT INTO Evento(id, fecha_evento, hora_evento, lugar_evento, tipo_evento, tiempo_evento)

VALUES (1,'Sábado 15 de Enero', '4 Pm', 'Islas las cascadas','Boda','9 horas');

insert into meseros (id, nombre, dpi, telefono_contacto, edad, genero, disponibilidad_evento)

values (1, 'Josue Lopez', '1245879325', '52687941', '20', 'Masculino', 'Disponible');

INSERT INTO ofrece(ID_SERVICIO, id_contrato)

VALUES (2,1);

INSERT INTO asigna(ID_EVENTO, ID_MESEROS)

```
VALUES (6,11);


INSERT INTO Vendedor(id, nombre_vendedor)
values (1,'Denis López');
```

Restricciones y consultas adicionales:

```
-- A Realizar una consulta para ver los contratos completos en enero de cualquier año
SELECT Contrato.estado_cumplimiento
FROM Contrato
WHERE strftime('%m', fecha_requerimiento) = '01' AND estado_cumplimiento = 'Completado';


-- B Consulta para contar los contratos de cada servicio
SELECT Servicio.descripcion, COUNT(Contrato.id_vendedor) AS veces_contratado
FROM Servicio
LEFT JOIN Contrato ON Servicio.id = Contrato.id_vendedor
GROUP BY Servicio.descripcion;


--C Realizar una consulta donde me indique los contratos cuyo estado no sea completado
SELECT *
FROM Contrato
WHERE Contrato.estado_cumplimiento <> 'Completado';
```

--D Realizar una consulta indicando los contratos realizados por cada vendedor

```
SELECT v.nombre_vendedor, COUNT(c.id) AS total_contratos

FROM vendedor v

INNER JOIN contrato c ON v.id = c.id_vendedor

GROUP BY v.nombre_vendedor;
```

--E Realizar una consulta cuyo resultado sean los meseros que atendieron un evento en un determinado mes

```
SELECT strftime('%m', fecha_evento) AS mes, COUNT(*) AS cantidad_eventos

FROM Evento

GROUP BY mes;
```

-- F Consulta para obtener los meseros que atendieron un evento en un mes específico

```
SELECT Contrato.id_evento

FROM Contrato

WHERE strftime('%m', fecha_requerimiento) = '05';
```

--G Realizar una consulta donde me indique el numero de eventos atendido por cada mesero

```
SELECT m.nombre AS nombre_mesero, COUNT(e.id) AS numero_eventos_atendidos
```

```
FROM meseros m

LEFT JOIN Evento e ON m.id = e.id

GROUP BY m.nombre;


-- H Consulta para obtener los eventos realizados por mes

SELECT fecha_requerimiento

FROM Contrato

WHERE strftime('%m', id_evento) = 'Enero';


-- I Consulta del día de la semana en el que se realizan más eventos

SELECT strftime('%w', fecha_evento) AS dia_semana, COUNT(*) AS cantidad_eventos

FROM Evento

GROUP BY dia_semana

ORDER BY cantidad_eventos DESC

LIMIT 1;


-- J Resumen de eventos con nombre del contratante, fecha de completado, monto y vendedor

SELECT c.nombre_contratante, e.fecha_evento, c.monto_pagado, v.nombre_vendedor

FROM Contrato c

JOIN Vendedor v ON c.id_vendedor = v.id

JOIN Evento e ON c.id_evento = e.id;
```

```
-- K Actualizar el valor inicial del servicio P2 de 16,000 a 16,500

UPDATE Servicio

SET costo = '16,500'

WHERE nombre = 'P2';

select * from Servicio


--L Debido al costo de los productos se deberá incrementar los servicios mayores a 15,000 suben un 10%

UPDATE Servicio

SET costo = CAST(costo AS FLOAT) * 1.1

WHERE CAST(costo AS FLOAT)>15000;

select *from Servicio

-- L Actualizar los servicios mayores a 15,000 con un incremento del 10%

UPDATE Servicio

SET costo = costo * 1.1

WHERE costo > 15000;


-- M Actualizar el nombre del servicio P4 a D4

UPDATE Servicio
```

```
SET nombre = 'D4'

WHERE nombre = 'P4';

--Actualizar el nombre del servicio P5 a D5

UPDATE Servicio

SET nombre = 'D5'

WHERE nombre = 'P5';


-- N  El servicio p6 será desactivado

UPDATE Servicio

SET nombre = 'Desactivado'

WHERE nombre = 'P6';


-- O  Indicar cual fue la fecha en la que se vendió el ultimo servicio p6

SELECT MAX(c.fecha_requerimiento) AS fecha_ultimo_p6

FROM Contrato c

JOIN Servicio s ON c.id_evento = s.id

WHERE s.nombre = 'P6';


--P Indicar cual es el genero de los meseros que más atienden los eventos

SELECT m.genero, COUNT(*) AS cantidad_eventos

FROM meseros m
```

```
JOIN asigna a ON m.id = a.id_meseros

GROUP BY m.genero

ORDER BY cantidad_eventos DESC

LIMIT 1;

--Q Realizar una consulta que me indique el numero de horas trabajadas por los meseros en un mes

SELECT m.nombre, SUM(strftime('%H', e.tiempo_evento)) AS horas_trabajadas

FROM meseros m

JOIN asigna a ON m.id = a.id_meseros

JOIN Evento e ON a.id_evento = e.id

WHERE strftime('%m', e.fecha_evento) = '01'

GROUP BY m.nombre;


--R Realizar una tabla que me diga el número de horas trabajadas por los meseros el año anterior, los datos del mesero que se solicitan son, nombre, dpi, edad y estado

SELECT m.nombre, m.dpi, m.edad, m.nombre, SUM(strftime('%H', e.tiempo_evento)) AS horas_trabajadas

FROM meseros m

JOIN asigna a ON m.id = a.id_meseros

JOIN Evento e ON a.id_evento = e.id

WHERE strftime('%Y', e.fecha_evento) = strftime('%Y', date('now', '-1 year'))

GROUP BY m.nombre, m.dpi, m.edad, m.nombre, m.dpi, m.edad, m.nombre;
```

Glosario

AS: se utiliza para asignar un alias o nombre alternativo a una columna o tabla en una consulta.

AUTOINCREMENT: se utiliza en las bases de datos para generar automáticamente valores únicos y crecientes en una columna.

CREATE TABLE: se utiliza para crear una nueva tabla en una base de datos.

COUNT(*): Cuenta el número total de filas en una tabla, sin importar los valores de las columnas.

DDL: Es un conjunto de instrucciones en SQL (Structured Query Language) utilizadas para definir y administrar la estructura de una base de datos. Las instrucciones DDL se utilizan para crear, modificar y eliminar tablas, índices, vistas, restricciones y otros objetos de la base de datos.

DELETE: Se utiliza para eliminar filas específicas de una tabla. Permite eliminar registros que cumplan con una condición determinada.

DROP TABLE: Elimina las tablas ya creadas.

FROM: especifica la tabla o tablas de las cuales deseas seleccionar los datos en una consulta. Indica la fuente de datos sobre la cual se realizará la operación de selección.

GROUP BY: se encarga de agrupar los registros según el dato proporcionado.

INSERT: se utiliza para insertar nuevos registros en una tabla existente. Permite agregar datos a una tabla especificando los valores que se deben insertar en cada columna.

INTEGER: se utiliza para almacenar números enteros sin decimales.

JOIN: se utiliza en para combinar registros de dos o más tablas basándose en una condición de unión. Aquí tienes algunos ejemplos de cómo se utiliza la cláusula JOIN.

LIMIT 1: se usa para obtener solo el primer resultado.

ORDER BY: se utiliza para ordenar los resultados en orden descendente según el total.

ON: La cláusula "ON" se utiliza en SQL junto con la instrucción "JOIN" para especificar la condición de combinación entre las tablas que participan en la operación de unión.

PRIMARY KEY: es una parte fundamental del diseño de la base de datos, ya que proporciona una forma rápida y eficiente de acceder y relacionar los registros en una tabla.

REFERENCES: se utiliza para establecer una relación de clave externa (foreign key) entre dos tablas en una base de datos. La cláusula "REFERENCES" se utiliza en conjunto con la cláusula "FOREIGN KEY" para definir la relación entre las columnas de las tablas.

RENAME TO: Renombra la nueva tabla creada.

SELECT * FROM: Verifica que las tablas tengan sus atributos que se hayan realizado correctamente.

SELECT MAX: se utiliza para obtener el valor máximo de una columna en una consulta. Aquí tienes un ejemplo de cómo se usa.

SELECT MIN: se utiliza en para obtener el valor mínimo de una columna en una consulta. Aquí tienes un ejemplo de cómo se usa.

SET: La cláusula "SET" en SQL se utiliza para asignar nuevos valores a una o varias columnas en una operación de actualización.

SUM: para calcular la suma total de los montos que cumplen con las condiciones establecidas.

UPDATE: Se utiliza para modificar los datos existentes en una o varias filas de una tabla. Permite actualizar los valores de una o varias columnas en función de una condición especificada.

VARCHAR: se utiliza para almacenar cadenas de caracteres de longitud variable.

VALUES: se utiliza junto con la instrucción "INSERT" para especificar los valores que se van a insertar en las columnas correspondientes de una tabla. Permite proporcionar los datos que se desean agregar a una tabla en una única instrucción "INSERT".

WHERE: se usa para filtrar los registros de una tabla en función de una condición específica. Permite especificar una expresión lógica que debe evaluarse para cada fila de la tabla y solo se seleccionarán aquellas filas que cumplan con la condición especificada.

Conclusiones

La base de datos catering facilitará la gestión de eventos y contratos, mejorando la eficiencia operativa de Multiservicios CGT. Al contar con una estructura de datos adecuada, se podrá acceder rápidamente a la información relevante, agilizando la planificación, la asignación de recursos y la coordinación de actividades, lo cual es esencial en un negocio de catering.

La implementación de la base de datos catering permitirá a Multiservicios CGT ofrecer servicios más personalizados a sus clientes. Al tener acceso a información detallada sobre preferencias, historial de eventos y solicitudes especiales, la empresa podrá adaptar sus propuestas y ofrecer experiencias únicas y satisfactorias, mejorando así la fidelidad y satisfacción de los clientes.

La implementación de una base de datos entidad-relación (ER) para el negocio de catering de Multiservicios CGT es una estrategia acertada para optimizar la gestión de información. Este enfoque permitirá organizar de manera eficiente los datos relacionados con clientes, eventos, contratos, pedidos y pagos, brindando una visión integral del negocio.

Recomendaciones

Realizar pruebas exhaustivas de la base de datos antes de su implementación completa. Esto implica validar la estructura de la base de datos, realizar pruebas de carga y rendimiento, y verificar la precisión y coherencia de los datos almacenados. Las pruebas rigurosas ayudarán a identificar posibles errores o deficiencias antes de que la base de datos esté en pleno funcionamiento.

Proporcionar capacitación y soporte adecuados al personal que utilizará la base de datos catering. Esto incluye la formación sobre la estructura y la funcionalidad de la base de datos, así como el desarrollo de procedimientos y pautas para su correcto uso. El personal debe comprender cómo interactuar con la base de datos de manera eficiente y correcta para aprovechar al máximo su potencial y evitar errores costosos.

Implementar medidas de seguridad y privacidad sólidas para proteger la información confidencial almacenada en la base de datos catering. Esto puede incluir la aplicación de políticas de acceso basadas en roles, el cifrado de datos sensibles y la realización de copias de seguridad periódicas para garantizar la disponibilidad y la integridad de los datos.

Bibliografía

jon mircha.(2023, 21 de marzo).Curso modelo de base de datos.**https://www.youtube.com/watch?v=aFqHVE_Y_YU&feature=youtu.be**