NUTRIARG

Dietética

Tienda de alimentos naturales y saludables

Curso: SQL

Comisión: 34960

Profesor: Redondo, Camilo

Tutor: Ovejero, Cristian

Alumno: Carabajal, Marcelo

Año: 2022

Índice

1. Introducción……Pág. 3
2. Objetivo……Pág. 3
3. Situación problemática……Pág. 3
4. Modelo de negocio……Pág. 3
5. Diagrama de Entidad Relación……Pág. 4
6. Inserción de datos por importación……Pág. 4
7. Descripción de tablas……Pág. 5
8. Informe de vistas……Pág. 7
9. Informe de Funciones……Pág. 8
10. Informe de Stored Procedures (S.P.)……Pág. 8
11. Informe de Triggers……Pág. 9
12. Transaction……Pág. 10
13. Backup……Pág. 10
14. Herramientas utilizadas……Pág. 10

Proyecto Final “Nutriarg” Dietética

SQL - CODERHOUSE

1. Introducción

En este proyecto veremos el resultado de lo aprendido en el curso creando una base de datos relacional basada en un modelo de negocios e-commerce.

2. Objetivo

El objetivo de este proyecto es brindar una solución a la información de la tienda e-commerce diseñando una base de datos relacional donde desarrollaremos objetos que permitan el mantenimiento de la misma e implementaremos consultas SQL que permitan la generación de informes para que el usuario responsable del negocio pueda analizar la información almacenada de una manera más sencilla y eficiente.

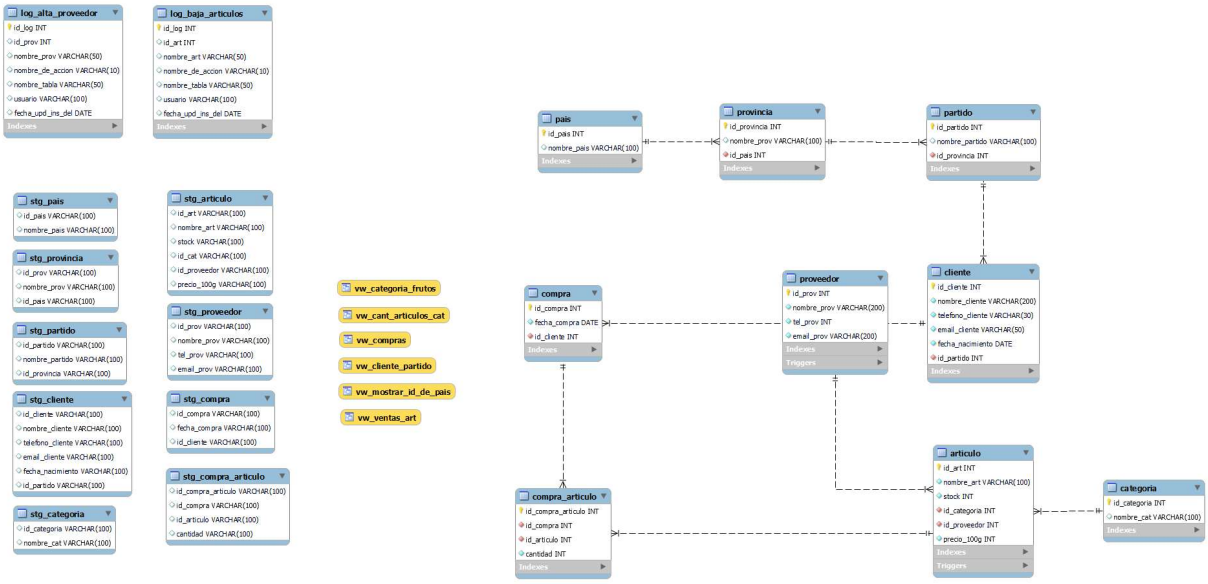
3. Situación problemática

El problema que se planteó fue la administración de modelo de negocio de la dietética que necesita actualizar la forma en la que guardan los registros los clientes como así también la de los países, provincias, partidos, clientes, categorías, artículos, proveedores, compras y ventas que concurren a la compañía, esto serviría para tener información más rápida mediante reportes de como el de compras o el de ventas.

4. Modelo de negocio

El modelo de negocio con el que se trabajará es una dietética e-commerce que apunta a automatizar toda su información de ahora en adelante, como así también tener registros de todas sus áreas.

5. Diagrama de Entidad Relación



6. Inserción de datos por importación

Pasos de inserción por importación de archivo CSV:

1) Creamos el archivo CSV con todos los registros necesarios como se detalla en el DER.

2) Creamos la base de datos/schema y las tablas con los comandos CREATE.

3) A la tabla que vamos a utilizar para que se importe le ponemos el nombre STAGE al comienzo para que se pueda importar de forma correcta. Ej: “CREATE TABLE STG_CLIENTE;”. El tipo de dato que usamos para todos los campos es VARCHAR.

4) Ingresamos al panel de la tabla, posicionándonos con el cursor del mouse en la tabla, y

realizamos click secundario para la importación por archivo. En este caso tengo 4 (cliente, articulo, proveedor y pedido). Hacer la importación por cada uno.

5) Una vez que hicimos la importación a la table STAGE (en el ejemplo, STG_CLIENTE), debemos copiar el contenido a la tabla CLIENTE, pero con los tipos de datos que corresponda en cada campo. Utilizamos el comando: “insert into cliente select * from

stg_cliente".

6) Realizar lo mismo para las demás tablas.

7. Descripción de tablas

En las siguientes imágenes estaremos viendo cada tabla y su contenido.

- País: esta tabla representa la cantidad de países que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_art	int	NO	PRI	NULL	
nombre_art	varchar(100)	NO		NULL	
stock	int	NO		NULL	
id_categoria	int	NO	MUL	NULL	
id_proveedor	int	NO	MUL	NULL	
precio_100g	int	NO		NULL	

- Provincia: esta tabla representa la cantidad de provincias (dentro del país) que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_provincia	int	NO	PRI	NULL	
nombre_prov	varchar(100)	YES		NULL	
id_pais	int	NO	MUL	NULL	

- Partido: esta tabla representa la cantidad de partidos (dentro de la provincia) que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_partido	int	NO	PRI	NULL	
nombre_partido	varchar(100)	YES		NULL	
id_provincia	int	NO	MUL	NULL	

- Cliente: esta tabla representa la cantidad de clientes dados de alta que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_cliente	int	NO	PRI	NULL	
nombre_cliente	varchar(200)	NO		NULL	
telefono_cliente	varchar(30)	NO		NULL	
email_cliente	varchar(50)	NO		NULL	
fecha_nacimiento	date	NO		NULL	
id_partido	int	NO	MUL	NULL	

- Proveedor: esta tabla representa la cantidad de proveedores dados de alta que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_prov	int	NO	PRI	NULL	
nombre_prov	varchar(200)	NO		NULL	
tel_prov	int	NO		NULL	
email_prov	varchar(200)	NO		NULL	

- Categoría: esta tabla representa la cantidad de categorías disponibles que se encuentran disponibles para los productos en la empresa.

Field	Type	Null	Key	Default	Extra
id_categoria	int	NO	PRI	NULL	
nombre_cat	varchar(100)	YES		NULL	

- Artículo: esta tabla representa la cantidad de artículos dados de alta que se encuentran disponibles en la empresa.

Field	Type	Null	Key	Default	Extra
id_art	int	NO	PRI	NULL	
nombre_art	varchar(100)	NO		NULL	
stock	int	NO		NULL	
id_categoria	int	NO	MUL	NULL	
id_proveedor	int	NO	MUL	NULL	
precio_100g	int	NO		NULL	

- Compra: esta tabla representa la cantidad de compras realizadas en la empresa.

	Field	Type	Null	Key	Default	Extra
▶	id_compra	int	NO	PRI	NULL	
	fecha_compra	date	NO		NULL	
	id_cliente	int	NO	MUL	NULL	

- Compra-Articulo: esta tabla representa la cantidad de compras-artículos realizadas en la empresa.

	Field	Type	Null	Key	Default	Extra
▶	id_compra_articulo	int	NO	PRI	NULL	
	id_compra	int	NO	MUL	NULL	
	id_articulo	int	NO	MUL	NULL	
	cantidad	int	NO		NULL	

8. Informe de vistas

VW_CLIENTE_PARTIDO

Muestra todos los clientes y su partido y, a su vez, también el ID del partido.

Tablas que la componen: CLIENTE y PARTIDO.

VW_VENTAS_ART

Muestra la cantidad de ventas por artículo, en el ejemplo de la tabla, se visualiza el artículo 2.

Tablas que la componen: COMPRA_ARTICULO

VW_COMPRAS

Muestra a los clientes que realizaron alguna compra en orden ascendente.

Tablas que la componen: COMPRA y CLIENTE

VW_CATEGORIA_FRUTOS

Se creó para mostrar la categoría donde aparezca "frutos", haciendo referencia con el comando LIKE a “frutos secos”.

Tablas que la componen: CATEGORIA

VW_MOSTRAR_ID_DE_PAIS

Muestra que país corresponde al ID = 3.

Tablas que la componen: PAIS

VW_CANT_ARTICULOS_CAT

Muestra la cantidad de artículos que hay para la categoría frutas deshidratadas.

Tablas que la componen: ARTICULO y CATEGORIA

9. Informe de Funciones

Se desarrollaron las siguientes funciones para facilitar algunos cálculos según sea necesario:

FN_EDAD

Fue diseñada para para detallar la edad de un cliente.

```
SELECT FN_EDAD(3); -- ejemplo cliente con id 3
```

FN_DESCUENTO

Función para obtener, por ejemplo, un descuento del 20% del total de la compra.

```
select fn_descuento (0.2, 1) as total_compra_descuento;
```

10. Informe de Stored Procedures (S.P.)

Se desarrollaron los siguientes procedimientos para la organización de tablas y registros:

SP_ORDENAR

El primer S.P. permite indicar a través de un parámetro el campo de ordenamiento de una tabla y mediante un segundo parámetro, si el orden es descendente o ascendente.

Tablas con las que interactúa: ARTICULO

```
CALL SP_ORDENAR (@PARAM_ORDER ,@PARAM_ASC_DESC);
```

SP_AGREGAR_PROVEEDOR

Fue diseñado para agregar un PROVEEDOR nuevo con estado (si retorno 0 = existe, no se agrega y si retorna 1 = no existe y lo agrega).

Tablas con las que interactúa: PROVEEDOR

```
CALL SP_AGREGAR_PROVEEDOR(@p_id_prov, @p_nombre_prov,
@p_tel_prov, @p_email_prov, @p_notificacion, @p_estado);

SELECT @p_notificacion, @p_estado;
```

11. Informe de Triggers

Para los triggers se crearon tablas de auditoría dentro del mismo esquema de la base de datos del proyecto, en los que se registra la información determinada por el trigger.

log_baja_articulos

Se creó para dar de baja un artículo y almacenarlo en mi tabla “log_baja_articulos” para poder hacer un seguimiento desde el área de IT.

```
select * from log_baja_articulos;
```

log_alta_proveedor

Se creó para dar de alta a un nuevo proveedor en nuestros registros.

```
select * from log_alta_proveedor;
```

12. Transaction

Para ejecutar la transacción, en primer lugar, se deshabilita la función autocommit y sql_safe_updates (“seteando” ambas en 0, como lo indica la imágen).

```
set @@autocommit = 0;
set sql_safe_updates = 0;
```

Luego, se indica el comienzo de una transacción sobre la tabla que realicemos las modificaciones, inserciones o borrado con el siguiente comando:

```
start transaction;
```

Para que se revierta la última acción (post start transaction), utilizamos:

```
rollback;
```

13. Backup

Se realiza backup sobre el contenido de todas las tablas del proyecto, las mismas son:

1. País
2. Provincia
3. Partido
4. Cliente
5. Proveedor
6. Categoría
7. Artículo
8. Compra
9. Compra-Articulo

14. Herramientas utilizadas

- MySQL Workbench.
- Stack Overflow.
- app.diagrams.net: esta página fue utilizada para crear el diagrama de entidad relación (DER).
- Microsoft Excel: utilizamos este programada para la normalización de datos a insertar en las tablas.
- Microsoft Word: se utilizó para el armado de este informe.
- Adobe Acrobat: el mismo se utilizad para visualizar este informe