UNIVERSIDAD
DE GRANADA

2023/2024

PRÁCTICA FINAL

BASES DE DATOS DISTRIBUIDAS

Javier Martín Alcalde
Teresa de Jesús Muñoz Rodríguez

Índice

1. Descripción del problema

Una compañía vinícola española desea implementar una base de datos distribuida para gestionar, lo más eficientemente posible, la distribución de vinos españoles por todo el territorio nacional. La compañía dispone de cuatro delegaciones territoriales: Madrid (localidad 1), Barcelona (localidad 2), La Coruña (localidad 3) y Granada (localidad 4). La delegación de Madrid suministra vinos a clientes de la región centro de la península (Castilla-León, Castilla-La Mancha, Aragón, Madrid, La Rioja). La delegación de Barcelona suministra vinos a clientes del Levante Español (Cataluña, Baleares, País Valenciano y Murcia). La delegación de La Coruña suministra vinos a clientes del Norte de España (Galicia, Asturias, Cantabria, País Vasco y Navarra). Y, por último, la delegación de Sevilla suministra vinos a clientes de la zona Sur del país (Andalucía, Extremadura, Canarias, Ceuta y Melilla).Cada delegación realiza sus labores de gestión a través de sucursales ubicadas en diferentes ciudades de cada autonomía. Cada sucursal solamente puede distribuir directamente vinos producidos en las comunidades autónomas que pertenecen a su delegación. Un cliente puede solicitar suministros de cualquier vino, pero siempre tendrá que hacerlo a través de sucursales de la delegación a la que pertenece su comunidad autónoma. Si una sucursal no puede suministrar directamente un vino a un cliente porque su delegación no lo distribuye, esta sucursal solicitará el pedido de vino a cualquier sucursal de la delegación que sí lo distribuya. Por ejemplo, un cliente de Andalucía requiere suministros de vinos de La Rioja. Este cliente tendrá que solicitar su pedido a cualquier sucursal de la delegación de Granada. Puesto que esta delegación no distribuye vinos de la Rioja, la sucursal a la que se haya dirigido el cliente tendrá que solicitar el pedido a una sucursal de la delegación de Madrid que es la encargada de la distribución de los vinos de la Rioja.

Los principales requisitos de funcionamiento y de almacenamiento de la compañía se describen con la siguiente lista de especificaciones:

LOS CLIENTES de la compañía se identifican mediante un código de cliente y, para cada uno de ellos, se almacena el DNI (o CIF), nombre del cliente, dirección o domicilio social del cliente, tipo de cliente, comunidad autónoma en la que reside, los distintos suministros de vino que la distribuidora le va a proporcionar (o le ha proporcionado) y las sucursales que van a tramitar(o han tramitado) cada uno de esos suministros. Cada suministro consiste en un tipo de vino, cantidad que se va a suministrar (número de botellas) y la fecha en la que se solicitó el suministro. A efectos de almacenamiento, la solicitud de dos (o más) suministros del mismo vino en la misma fecha y requeridos a la misma sucursal por el mismo cliente, se consideran como un único suministro en el cual se suman las cantidades que se van a suministrar (o se han suministrado).

LOS EMPLEADOS de la compañía se identifican mediante un código de empleado, código que mantendrán mientras trabajen en dicha compañía independientemente de la sucursal en la que estén destinados en un momento determinado. La administración de la distribuidora almacena para cada empleado DNI, nombre del empleado, fecha de comienzo de contrato con la compañía (NO la fecha de comienzo de trabajo en una sucursal), salario, dirección y sucursal de destino. Cada empleado, en un momento dado, sólo puede estar

asignado en una sucursal (no se mantendrá registro de las distintas sucursales a las que haya estado destinado el empleado).

LOS VINOS se identifican mediante un código de vino y, para cada uno de ellos, se almacena la marca del vino, año de la cosecha, denominación de origen (si la tiene), graduación, viñedo de procedencia, comunidad autónoma, cantidad producida (número de botellas), cantidad en stock (número de botellas) y productor. En cada momento, la cantidad en stock será la cantidad producida menos la cantidad total suministrada. Inicialmente, la cantidad en stock es la cantidad producida.

LOS PRODUCTORES se identifican mediante un código de productor y, para cada uno de ellos, se almacena el DNI (o CIF), nombre del productor, dirección o domicilio social del productor y los vinos que produce.

LAS SUCURSALES se identifican mediante un código de sucursal y, para cada una de ellas, se almacena el nombre de la sucursal, ciudad en la que se encuentra, comunidad autónoma en la que se ubica, director de la sucursal y los pedidos que ha realizado a otras sucursales para tramitar los suministros solicitados por sus clientes. Cada pedido consiste, en un tipo de vino, cantidad que se solicita y fecha del pedido. A efectos de almacenamiento, dos (o más) pedidos del mismo vino en la misma fecha y solicitados por la misma sucursal, se consideran como un único pedido en el cual se suman las cantidades solicitadas.

Las aplicaciones, tanto de actualización como de consulta, que usan datos de sucursales (incluidos datos de empleados, clientes, suministros, vinos que distribuye, etc.),se pueden generar en cualquier localidad, pero referencian con una probabilidad del 95% a sucursales cercanas.

2. Práctica 1: Diseño Conceptual y Lógico Global de la Base de Datos

2.1. Diagrama Entidad-Relación

2.1.1. Descripción General:

Este diagrama representa las entidades y relaciones clave en un sistema de distribución de una compañía vinícola, incluyendo empleados, vinos, productores, clientes y sucursales.

Entidades:

- PRODUCTOR (codPr, DNI, nombre, direccion)
- EMPLEADO (codEm, DNI, nombre, fechaContrato, salario, direccion)
- VINO (codVin, marca, anioCosecha, denominacionOrigen, graduacion, viniedoProcedencia, cantStock, cantProducida, comAutonoma)
- CLIENTE (codCl, DNI, nombre, direccion, tipo, comAutonoma)
- SUCURSAL (codSu, codDirector, nombre, ciudad, comAutonoma)

Relaciones entre Entidades:

- Vino - Productor (Binaria)
- Sucursal - Vino (Binaria)
- Empleado - Sucursal (Binaria)
- Cliente - Sucursal (Binaria)
- Sucursal - Sucursal (Unaria)

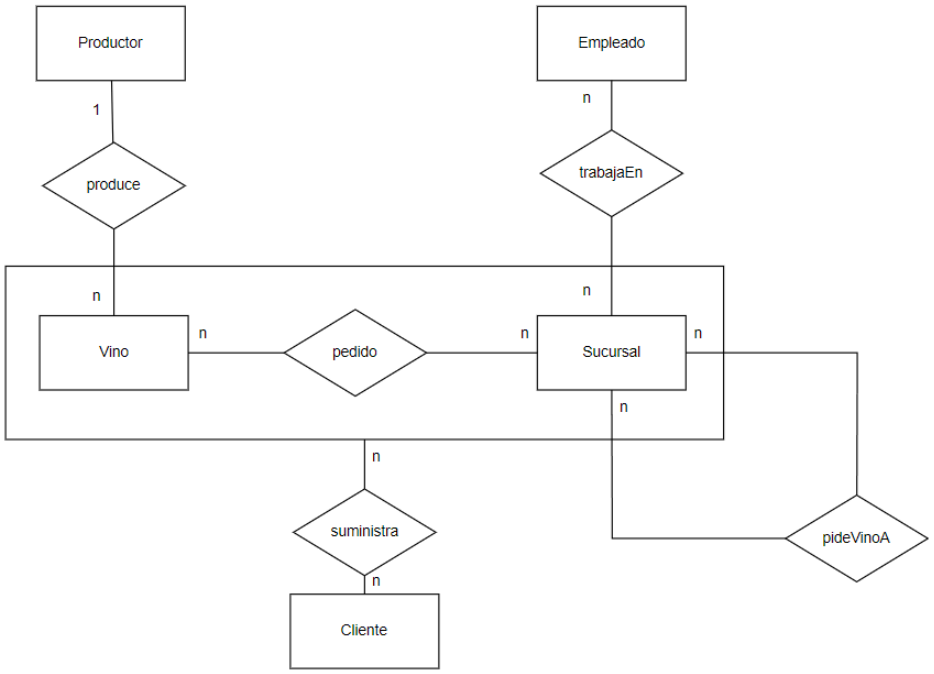
Cardinalidades:

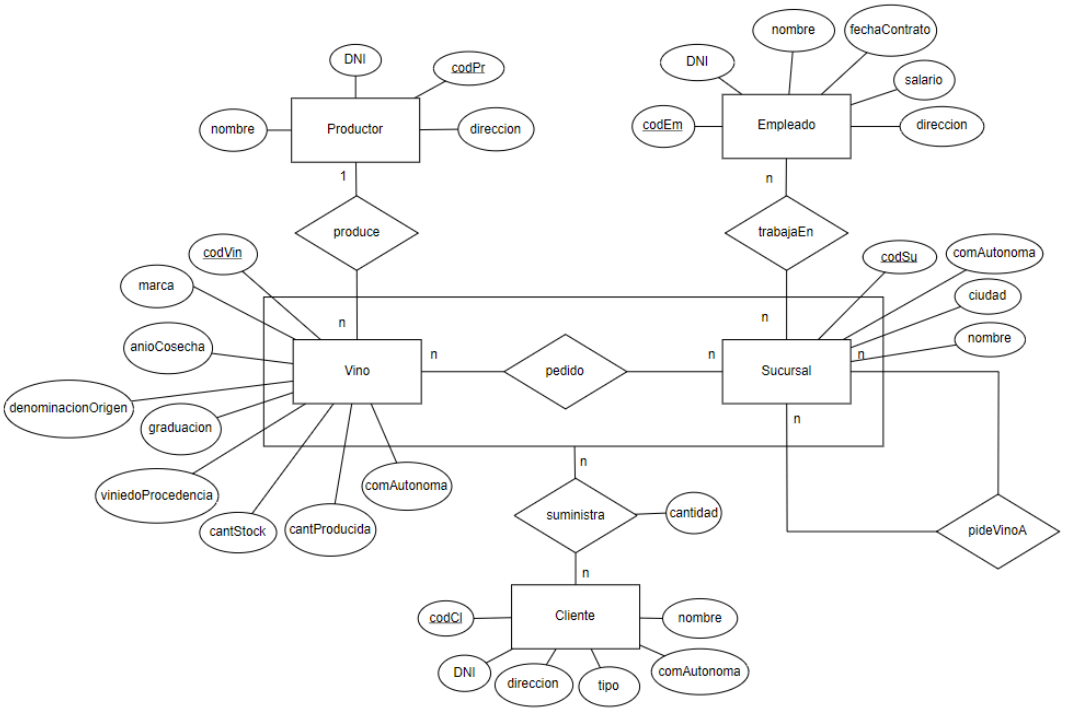
- Vino - Productor (N:1)
- Sucursal - Vino (N:N)
- Empleado - Sucursal (N:1)
- Cliente - Sucursal (N:N)
- Sucursal - Sucursal (N:N)

Claves Primarias:

- Productor: Clave Primaria = codPr
- Empleado: Clave Primaria = codEm
- Vino: Clave Primaria = codVin
- Cliente: Clave Primaria = codCl
- Sucursal: Clave Primaria = codSu

2.1.2. Diagrama E/R:

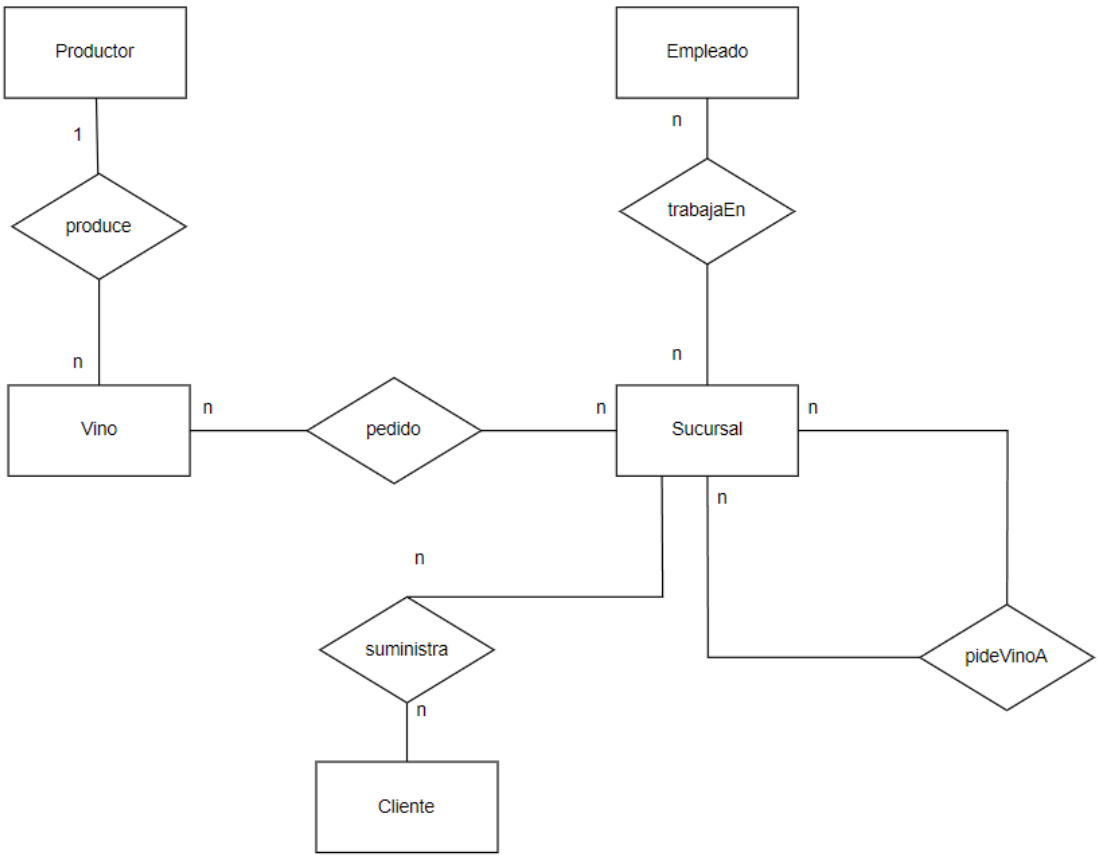




2.1.3. Alternativas de Diseño::

Una alternativa a este diseño consistiría en eliminar la agregación entre Vinos y Sucursal, la cuál contaría con el inconveniente de que la relación "pedido" entre Cliente y Sucursal naturalmente implica solicitar uno o más tipos de vinos a una o varias sucursales. En otras palabras, la solicitud de vinos está implícitamente vinculada a la relación "pedido". Por lo que la agregación nos proporcionaba una manera de modelar y entender más claramente la conexión entre Sucursal y Vinos en el contexto de la relación de pedido entre Cliente y Sucursal.

Por lo que, aunque técnicamente sería posible eliminar la agregación entre Sucursal y Vinos, mantenerla ayuda a expresar de manera más clara la relación entre estas entidades en el contexto de los pedidos realizados por los clientes a las sucursales. La agregación actúa como una estructura conceptual que facilita la comprensión del comportamiento del sistema.



2.2. Diseño Lógico:

Una vez comentado el diagrama E/R, realizaremos el diseño del modelo lógico de la base de datos, para esto haremos el paso a tablas del modelo de datos relacional.

2.2.1. Descripción de las tablas y los atributos:

Productor:

- Identificador de productor: Dato de tipo numérico que identifica a cada productor dentro del sistema.
- DNI: Dato de tipo alfanumérico compuesto por 8 números y 1 letra.
- Nombre: Dato de tipo String
- Dirección: Dato de tipo String

Empleado:

- Identificador de empleado: Dato de tipo numérico que identifica a cada empleado dentro del sistema.
- DNI: Dato de tipo alfanumérico compuesto por 8 números y 1 letra.
- Nombre: Dato de tipo String
- Fecha del contrato: Dato de tipo Date. Se refiere al inicio del contrato con la compañía, que es distinta a la de inicio de contrato con la sucursal.
- Salario: Dato de tipo numérico en coma flotante
- Dirección: Dato de tipo String

Vino:

- Identificador de vino: Dato de tipo numérico que identifica a cada vino dentro del sistema.
- Marca: Dato de tipo String
- Año de la cosecha: Dato de tipo numérico compuesto por 4 dígitos.
- Denominación de origen: Dato de tipo String
- Graduación: Dato de tipo String
- Viñedo de procedencia: Dato de tipo String
- Cantidad en stock: Dato de tipo numérico que hace alusión al número de botellas de vino que hay en stock
- Cantidad producida: Dato de tipo numérico que hace alusión al total de botellas de vino que se han producido
- Comunidad autónoma: Dato de tipo String

Cliente:

- Identificador de cliente: Dato de tipo numérico que identifica a cada cliente dentro del sistema.

- DNI: Dato de tipo alfanumérico compuesto por 8 números y 1 letra.
- Nombre: Dato de tipo String
- Dirección: Dato de tipo String
- Tipo: Dato de tipo String que hace alusión al tipo de cliente
- Comunidad autónoma: Dato de tipo String

Sucursal:

- Identificador de sucursal: Dato de tipo numérico que identifica a cada sucursal dentro del sistema.
- Identificador de director: Dato de tipo numérico que identifica al director de la sucursal
- Nombre: Dato de tipo String
- Ciudad: Dato de tipo String
- Comunidad autónoma: Dato de tipo String

Produce: Relación entre las entidades Productor y Vino

- Identificador del productor: Dato de tipo numérico que hace referencia al identificador del productor que va a producir el vino
- Identificador del vino: Dato de tipo numérico que hace referencia al identificador del vino que va a ser producido.

TrabajaEn: Relación entre las entidades Empleado y Sucursal

- Identificador de empleado: Dato de tipo numérico que hace referencia al identificador del empleado que trabaja en dicha sucursal
- Identificador de sucursal: Dato de tipo numérico que hace referencia al identificador de la sucursal en la que trabaja el empleado en cuestión

Pedido: Relación entre las entidades Vino y Sucursal

- Identificador de vino: Dato de tipo numérico que hace referencia al identificador del vino que se va a pedir
- Identificador de sucursal: Dato de tipo numérico que hace referencia al identificador de la sucursal a la que se le va a hacer el pedido

Suministra: Relación entre las entidades Cliente y Sucursal

- Identificador de cliente: Dato de tipo numérico que hace referencia al identificador del cliente al que se le va a suministrar el vino
- Identificador de vino: Dato de tipo numérico que hace referencia al identificador del vino que se va a suministrar
- Identificador de sucursal: Dato de tipo numérico que hace referencia al identificador de la sucursal que va a suministrar el vino.
- Cantidad: Dato de tipo numérico entero que hace referencia al número de botellas de vino que se van a suministrar.

PideVinoA: Relación entre las entidades Sucursal (pide) y Sucursal (suministra)

- Identificador de sucursal que pide: Dato de tipo numérico que hace referencia al identificador de la sucursal que va a solicitar el vino (que solicita el pedido)
- Identificador de la sucursal que suministra: Dato de tipo numérico que hace referencia al identificador de la sucursal que va a suministrar el vino (que recibe la solicitud del pedido)
- Identificador del pedido: Dato de tipo numérico que identifica cada pedido
- Identificador de vino: Dato de tipo numérico que hace referencia al identificador del vino que se va a pedir
- Cantidad: Dato de tipo numérico entero que hace alusión al número de botellas de vino que se va a pedir
- Fecha del pedido: Dato de tipo Date que hace alusión a la fecha en la que se solicita el pedido.

2.2.2. Conjunto de Tablas:

1) PRODUCTOR (codPr, DNI, nombre, direccion)
2) EMPLEADO (codEm, DNI, nombre, fechaContrato, salario, direccion)
3) VINO (codVin, marca, anioCosecha, denominacionOrigen, graduacion, viniedoProcedencia, cantStock, cantProducida, comAutonoma)
4) CLIENTE (codCl, DNI, nombre, direccion, tipo, comAutonoma)
5) SUCURSAL (codSu, codDirector, nombre, ciudad, comAutonoma)
6) PRODUCE (codPr (CE1), codVin(CE3))
7) TRABAJAEN (codEm (CE2), codSu(CE5))
8) PEDIDO (codVin(CE3), codSu (CE5))
9) SUMINISTRA (codCl (CE4), codVin, codSu (CE8), cantidad)
10) PIDEVINOA (codSu1(CE5), codSu2 (CE5), codPed, codVin(CE12), cantidad, fechaPedido)

11) (3) - (6)
12) VINO (codVin, codPr(CE1) marca, anioCosecha, denominacionOrigen, graduacion, viniedoProcedencia, cantStock, cantProducida, comAutonoma)

13) (2) - (7)
    EMPLEADO (codEm, codSu(CE5), DNI, nombre, fechaContrato, salario, direccion)

3. Práctica 2: Diseño de la Fragmentación y de la Asignación

3.1. Fragmentación y Asignación

Sabemos que las aplicaciones de actualización y consulta que utilizan datos de los sucursales se generan en cualquier localidad pero referencian con una probabilidad del 0.95 a sucursales cercanas. Por esta razón vamos a fragmentar de manera horizontal primaria la relación SUCURSAL según el atributo comAutonoma. También vamos a fragmentar las relaciones VINO y CLIENTE por el mismo atributo.

El conjunto P de predicados es el siguiente:

P = {comAutonoma="Castilla-León", comAutonoma="Castilla-La Mancha", comAutonoma="Aragón", comAutonoma="Madrid", comAutonoma="La Rioja", comAutonoma="Cataluña", comAutonoma="Baleares", comAutonoma="País Valenciano", comAutonoma="Murcia", comAutonoma="Galicia", comAutonoma="Asturias", comAutonoma="Cantabria", comAutonoma="País Vasco", comAutonoma="Navarra", comAutonoma="Andalucía", comAutonoma="Extremadura", comAutonoma="Canarias", comAutonoma="Ceuta", comAutonoma="Melilla"}

Usaremos la siguiente notación para facilitar los pasos siguientes:

p1 ≡ comAutonoma= "Castilla-León"
p2 ≡ comAutonoma= "Castilla-La Mancha"
p3 ≡ comAutonoma= "Aragón"
p4 ≡ comAutonoma= "Madrid"
p5 ≡ comAutonoma= "La Rioja"
p6 ≡ comAutonoma= "Cataluña"
p7 ≡ comAutonoma= "Baleares"
p8 ≡ comAutonoma= "País Valenciano"
p9 ≡ comAutonoma= "Murcia"
p10 ≡ comAutonoma= "Galicia"
p11 ≡ comAutonoma= "Asturias"
p12 ≡ comAutonoma= "Cantabria"
p13 ≡ comAutonoma= "País Vasco"
p14 ≡ comAutonoma= "Navarra"
p15 ≡ comAutonoma= "Andalucía"
p16 ≡ comAutonoma= "Extremadura"
p17 ≡ comAutonoma= "Canarias"
p18 ≡ comAutonoma= "Ceuta"
p19 ≡ comAutonoma= "Melilla"

Por tanto, tenemos $2^{19} = 524288$ expresiones conjuntivas. Por simplicidad solo vamos a identificar en detalle un subconjunto a partir del cual se puede entender cómo se realizará la evaluación del resto:

- y1 = p1 ∧ p2 ∧ p3 ∧ p4 ∧ p5 ∧ p6 ∧ p7 ∧ p8 ∧ p9 ∧ p10 ∧ p11 ∧ p12 ∧ p13 ∧ p14 ∧ p15 ∧ p16 ∧ p17 ∧ p18 ∧ p19 → Falso
- y2 = p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y3 = ¬p1 ∧ p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y4 = ¬p1 ∧ ¬p2 ∧ p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y5 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y6 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y7 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y8 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y9 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y10 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y11 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y12 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y13 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y14 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y15 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y16 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y17 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Verdadero
- y18 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ p17 ∧ ¬p18 ∧ ¬p19 → Verdadero

- y19 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ p18 ∧ ¬p19 → Verdadero
- y20 = ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ p19 → Verdadero
- y21 = p1 ∧ p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧¬ p19 → Falso
- y21 = p1 ∧ ¬p2 ∧ p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Faso
- …
- y524288= ¬p1 ∧ ¬p2 ∧ ¬p3 ∧ ¬p4 ∧ ¬p5 ∧ ¬p6 ∧ ¬p7 ∧ ¬p8 ∧ ¬p9 ∧ ¬p10 ∧ ¬p11 ∧ ¬p12 ∧ ¬p13 ∧ ¬p14 ∧ ¬p15 ∧ ¬p16 ∧ ¬p17 ∧ ¬p18 ∧ ¬p19 → Falso

Las expresiones conjuntivas de y2 a y20 son verdaderas porque solo hay un predicado verdadero. Las demás expresiones se evalúan falsas ya que una sucursal no puede encontrarse en dos o más comunidades autónomas a la vez. Por último, la expresión y524288 también es falsa debido a que hemos declarado todas las comunidades autónomas que hay en nuestro país por lo que todas las sucursales deben pertenecer a una comunidad.

Los esquemas de fragmentación serían los siguientes:

- $Sucursal1 = SL_{y2}(Sucursal)$
- $Sucursal2 = SL_{y3}(Sucursal)$
- …
- $Sucursal19 = SL_{y20}(Sucursal)$

- $Cliente1 = SL_{y2}(Cliente)$
- $Cliente2 = SL_{y3}(Cliente)$
- …
- $Cliente19 = SL_{y20}(Cliente)$

- $Vino1 = SL_{y2}(Vino)$
- $Vino2 = SL_{y3}(Vino)$
- …
- $Vino19 = SL_{y20}(Vino)$

Además, vamos a fragmentar de manera horizontal derivada la relación Empleado y Pedido a partir de la relación horizontal que hemos realizado de Sucursal y la relacion Productor a partir de la fragmentación de Vino.

- $Empleado1 = SJ_{codSu=codSu}(Sucursal1)$
- $Empleado2 = SL_{codSu=codSu}(Sucursal2)$

- …
- $Empleado19 = SL_{codSu=codSu}(Sucursal3)$

- $Pedido1 = SJ_{codSu=codSu}(Sucursal1)$
- $Pedido2 = SL_{codSu=codSu}(Sucursal2)$
- …
- $Pedido19 = SL_{codSu=codSu}(Sucursal3)$

- $Suministro1 = SJ_{codSu=codSu}(Sucursal1)$
- $Suministro2 = SL_{codSu=codSu}(Sucursal2)$
- …
- $Suministro19 = SL_{codSu=codSu}(Sucursal19)$

- $Productor1 = SJ_{codVin=codVin}(Vino1)$
- $Productor2 = SL_{codVin=codVin}(Vino2)$
- …
- $Productor19 = SL_{codVin=codVin}(Vino19)$

La relación PideVinoA no va a ser fragmentada debido a la inexistencia de un criterio lógico para su realización.

3.2. Replicación:

En cuanto a la estrategia de replicación, nos centraremos principalmente en la estabilidad de los datos de las tablas. Esto significa que buscaremos tener copias de datos en diversas ubicaciones, priorizando aquellas tablas cuyos datos experimenten cambios mínimos o poco frecuentes.

En este contexto, procederemos a replicar la tabla PideVinoA en cada localidad. Esta decisión se fundamenta en la necesidad de tener los datos de PideVinoA disponibles en cada ubicación, lo cual es esencial para facilitar el acceso al flujo de pedidos entre sucursales. Se presume que esta tabla posee una baja volatilidad, ya que las operaciones asociadas, como realizar nuevos pedidos o cancelarlos, no serán tan frecuentes ni dinámicas como las descritas en la fragmentación anterior.

Por todo esto, la elección de replicar la tabla PideVinoA parece ser coherente con las consideraciones mencionadas.

Localidad asociada a la base de datos:

1 -> Madrid

2 -> Barcelona
3 -> La Coruña
4 -> Granada

3.3. Asignación de Fragmentos, Réplicas y Relaciones:

Después de establecer la fragmentación, procederemos a asignar los diversos fragmentos a cada ubicación, considerando la división geográfica solicitada en el enunciado. La asignación se realizará como sigue:

Localidad de **Madrid**:
- Sucursal1, Sucursal2, Sucursal3, Sucursal4, Sucursal5
- Cliente1, Cliente2, Cliente3,Cliente4, Cliente5
- Vino1, Vino2, Vino3, Vino4, Vino5
- Empleado1, Empleado2, Empleado3, Empleado4, Empleado5
- Pedido1, Pedido2, Pedido3, Pedido4, Pedido5
- Productor1, Productor2, Productor3, Productor4, Productor5
- Suministro1, Suministro2, Suministro3, Suministro4, Suministro5

Localidad de **Barcelona**:
- Sucursal6, Sucursal7, Sucursal8, Sucursal9
- Cliente6, Cliente7, Cliente8,Cliente9
- Vino6, Vino7, Vino8, Vino9
- Empleado6, Empleado7, Empleado8, Empleado9
- Pedido6, Pedido7, Pedido8, Pedido9
- Productor6, Productor7, Productor8, Productor9
- Suministro6, Suministro7, Suministro8, Suministro9

Localidad de **La Coruña**:
- Sucursal10, Sucursal11, Sucursal12, Sucursal13, Sucursal14
- Cliente10, Cliente11, Cliente12,Cliente13, Cliente14
- Vino10, Vino11, Vino12, Vino13, Vino14
- Empleado10, Empleado11, Empleado12, Empleado13, Empleado14
- Pedido10, Pedido11, Pedido12, Pedido13, Pedido14
- Productor10, Productor11, Productor12, Productor13, Productor14
- Suministro10, Suministro11, Suministro12, Suministro13, Suministro14

Localidad de **Granada**:
- Sucursal15, Sucursal16, Sucursal17, Sucursal18, Sucursal19
- Cliente15, Cliente16, Cliente17,Cliente18, Cliente19
- Vino15, Vino16, Vino17, Vino18, Vino19
- Empleado15, Empleado16, Empleado17, Empleado18, Empleado19
- Pedido15, Pedido16, Pedido17, Pedido18, Pedido19
- Productor15, Productor16, Productor17, Productor18, Productor19
- Suministro15, Suministro16, Suministro17, Suministro18, Suministro19

3.4. Restricciones de Integridad:

Para la implementación de las restricciones de nuestra base de datos hemos intentado priorizar siempre el uso de claves externas para referenciar las demás tablas siempre que ha sido necesario. En los demás casos, hemos implementado una serie de disparadores, los cuales muestran un mensaje de alerta por pantalla para indicar con mayor claridad el tipo de error.

3.5. Implementación de Actualizaciones

Hemos implementado un procedimiento para cada actualización requerida, llamando en algunos a otros procedimientos creados anteriormente para evitar la repetición de código.

3.6. Código de Creación de Tablas y Vistas

En este apartado solo aparece el código de la localidad 1, ya que para las demás localidades el código es el mismo solo que cambiando el índice por el de la localidad.

3.6.1. Tablas

```
CREATE TABLE SUCURSAL1(
    CODSU NUMBER PRIMARY KEY,
    NOMBRE VARCHAR2(30),
    CIUDAD VARCHAR2(30),
    COMAUTONOMA VARCHAR2(30) NOT NULL,
    CODDIRECTOR INT UNIQUE
);

CREATE TABLE PRODUCTOR1(
    CODPR INT PRIMARY KEY,
    DNI VARCHAR(10) NOT NULL UNIQUE,
    NOMBRE VARCHAR2(30),
    DIRECCION VARCHAR(30)
);
```

```sql
CREATE TABLE VINO1(
    CODVIN INT PRIMARY KEY,
    MARCA VARCHAR(20),
    ANIOCOSECHA INT NOT NULL,
    DENOMINACION_ORIGEN VARCHAR(20),
    GRADUACION NUMBER(4,2),
    VINIEDOPROCEDENCIA VARCHAR(20),
    CANTSTOCK INT,
    CANTPRODUCIDA INT,
    COMAUTONOMA VARCHAR(20) NOT NULL,
    CODPR INT,
    CONSTRAINT CE1 FOREIGN KEY (CODPR) REFERENCES PRODUCTOR1(CODPR)
);


CREATE TABLE EMPLEADO1(
    CODEM INT PRIMARY KEY,
    CODSU INT,
    DNI VARCHAR(10) NOT NULL UNIQUE,
    NOMBRE VARCHAR(20),
    FECHACONTRATO DATE NOT NULL,
    SALARIO INT NOT NULL,
    DIRECCION VARCHAR(30),
    CONSTRAINT CE2 FOREIGN KEY (CODSU) REFERENCES SUCURSAL1(CODSU)
);
```

```sql
CREATE TABLE CLIENTE1(
    CODCL INT PRIMARY KEY,
    DNI VARCHAR2(10) NOT NULL UNIQUE,
    NOMBRE VARCHAR2(20),
    DIRECCION VARCHAR2(30),
    TIPO VARCHAR2(20) CHECK (TIPO IN ('A', 'B', 'C')),
    COMAUTONOMA VARCHAR2(20)
);


CREATE TABLE PEDIDO1(
    CODVIN INT,
    CODSU INT,
    CANTIDAD INT,
    CONSTRAINT CE5 FOREIGN KEY (CODVIN) REFERENCES VINO1(CODVIN),
    CONSTRAINT CE6 FOREIGN KEY (CODSU) REFERENCES SUCURSAL1(CODSU)
);
```

```
CREATE TABLE SUMINISTRO1(
    CODCL INT,
    CODVIN INT,
    CODSU INT,
    CANTIDAD INT,
    FECHASUMINISTRO DATE,
    CONSTRAINT CE4 FOREIGN KEY (CODCL) REFERENCES CLIENTE1(CODCL),
    CONSTRAINT CE8 FOREIGN KEY (CODSU) REFERENCES SUCURSAL1(CODSU)
);


CREATE TABLE PIDEVINOA (
    CODPED INT PRIMARY KEY,
    CODSU1 INT NOT NULL,
    CODSU2 INT NOT NULL,
    CODVIN INT NOT NULL,
    CANTIDAD INT NOT NULL,
    FECHAPEDIDO DATE
);

ALTER TABLE SUCURSAL1 ADD CONSTRAINT CE10 FOREIGN KEY (CODDIRECTOR) REFERENCES EMPLEADO1(CODEM);
```

Nota aclaratoria:

Puesto que la tabla SUCURSAL tiene una variable externa a la tabla EMPLEADO y, a su vez, la tabla EMPLEADO también referencia a la tabla SUCURSAL, hemos creado primero la tabla SUCURSAL definiendo el atributo CODDIRECTOR y tras crear la tabla EMPLEADO hemos modificado dicho atributo empleando lo siguiente:

```
ALTER TABLE SUCURSAL1 ADD CONSTRAINT CE10 FOREIGN KEY (CODDIRECTOR) REFERENCES EMPLEADO1(CODEM);
```

3.6.2. Vistas

```
CREATE VIEW SucursalView AS
SELECT * FROM SUCURSAL1
UNION
SELECT * FROM kdekilo2.SUCURSAL2
UNION
SELECT * FROM kdekilo3.SUCURSAL3
UNION
SELECT * FROM kdekilo4.SUCURSAL4;

CREATE VIEW ProductorView AS
SELECT * FROM PRODUCTOR1
UNION
SELECT * FROM kdekilo2.PRODUCTOR2
UNION
SELECT * FROM kdekilo3.PRODUCTOR3
UNION
SELECT * FROM kdekilo4.PRODUCTOR4;

CREATE VIEW VinoView AS
SELECT * FROM VINO1
UNION
SELECT * FROM kdekilo2.VINO2
UNION
SELECT * FROM kdekilo3.VINO3
UNION
SELECT * FROM kdekilo4.VINO4;

CREATE VIEW EmpleadoView AS
SELECT * FROM EMPLEADO1
UNION
SELECT * FROM kdekilo2.EMPLEADO2
UNION
SELECT * FROM kdekilo3.EMPLEADO3
UNION
SELECT * FROM kdekilo4.EMPLEADO4;

CREATE VIEW ClienteView AS
SELECT * FROM CLIENTE1
UNION
SELECT * FROM kdekilo2.CLIENTE2
UNION
SELECT * FROM kdekilo3.CLIENTE3
UNION
SELECT * FROM kdekilo4.CLIENTE4;
```

```sql
CREATE VIEW PedidoView AS
SELECT * FROM PEDIDO1
UNION
SELECT * FROM kdekilo2.PEDIDO2
UNION
SELECT * FROM kdekilo3.PEDIDO3
UNION
SELECT * FROM kdekilo4.PEDIDO4;

CREATE VIEW SuministroView AS
SELECT * FROM SUMINISTRO1
UNION
SELECT * FROM kdekilo2.SUMINISTRO2
UNION
SELECT * FROM kdekilo3.SUMINISTRO3
UNION
SELECT * FROM kdekilo4.SUMINISTRO4;

CREATE VIEW PideVinoAView AS
SELECT * FROM PIDEVINOA;
```

4. Práctica 5: Implementación de Consultas

```sql
-- CONSULTAS:

-- 1. "Listar los clientes (nombre y dirección) de Andalucía o Castilla-La Mancha y las
-- sucursales (nombre y ciudad), a los que se le ha suministrado vino de marca "Tablas
-- de Daimiel" entre el 1 de Enero de 2023 y el 1 de septiembre de 2023.
SET SERVEROUTPUT ON;
SELECT
    DISTINCT c.NOMBRE AS "Nombre Cliente",
    c.DIRECCION AS "Dirección Cliente",
    s.NOMBRE AS "Nombre Sucursal",
    s.CIUDAD AS "Ciudad Sucursal"
FROM
    ClienteView c
JOIN
    SuministroView sm ON c.CODCL = sm.CODCL
JOIN
    SucursalView s ON sm.CODSU = s.CODSU
JOIN
    VinoView v ON sm.CODVIN = v.CODVIN
WHERE
    (c.COMAUTONOMA = 'Andalucia' OR c.COMAUTONOMA = 'Castilla-La Mancha')
    AND v.MARCA = 'Tablas de Daimiel'
    AND sm.FECHASUMINISTRO BETWEEN TO_DATE('2023-01-01', 'YYYY-MM-DD') AND TO_DATE('2023-09-01', 'YYYY-MM-DD');
```

```sql
-- 2. Dado por teclado el código de un productor: "Listar la marca, el año de cosecha de
-- cada uno de los vinos producidos por él y la cantidad total suministrada de cada uno
-- de ellos a clientes de las comunidades autónomas de Baleares, Extremadura o País
-- Valenciano.
SET SERVEROUTPUT ON;
DECLARE
    codigo_productor_input INT;
BEGIN
    -- Solicitar el código de un productor por teclado
    DBMS_OUTPUT.PUT_LINE('Introduce el código de un productor: ');
    codigo_productor_input := &codigo_productor_input;

    -- Consulta SQL para listar la marca, el año de cosecha y la cantidad total suministrada
    FOR vino_info IN (
        SELECT
            v.MARCA AS Marca,
            v.ANIOCOSECHA AS AnioCosecha,
            SUM(s.CANTIDAD) AS CantidadTotalSuministrada
        FROM
            VinoView v
        JOIN
            SuministroView s ON v.CODVIN = s.CODVIN
        JOIN
            ClienteView c ON s.CODCL = c.CODCL
        WHERE
            v.CODPR = codigo_productor_input AND
            (c.COMAUTONOMA IN ('Baleares', 'Extremadura', 'País Valenciano') OR c.COMAUTONOMA IS NULL)
        GROUP BY
            v.MARCA, v.ANIOCOSECHA
    ) LOOP
    DBMS_OUTPUT.PUT_LINE('Marca: ' || vino_info.Marca || ', Año de Cosecha: ' || vino_info.AnioCosecha ||
                         ', Cantidad Total Suministrada: ' || vino_info.CantidadTotalSuministrada);
    END LOOP;
END;
```

```sql
-- 3. Dado por teclado el código de una sucursal: "Listar el nombre de cada uno de sus
-- clientes, su tipo y la cantidad total vino de Rioja o Albariño que se le ha suministrado
-- a cada uno de ellos (solamente deberán aparecer aquellos clientes a los que se les ha
-- suministrado vinos con esta denominación de origen).
SET SERVEROUTPUT ON;
DECLARE
    codigo_sucursal_input INT;
BEGIN
    -- Solicitar el código de una sucursal por teclado
    DBMS_OUTPUT.PUT_LINE('Introduce el código de una sucursal: ');
    codigo_sucursal_input := &codigo_sucursal_input;

    -- Consulta SQL para listar clientes, su tipo y la cantidad total de vino suministrado
    FOR cliente_info IN (
        SELECT
            c.NOMBRE AS "Nombre del Cliente",
            c.TIPO AS "Tipo de Cliente",
            SUM(s.CANTIDAD) AS "Cantidad Total Suministrada"
        FROM
            ClienteView c
        JOIN
            SuministroView s ON c.CODCL = s.CODCL
        JOIN
            VinoView v ON s.CODVIN = v.CODVIN
        WHERE
            s.CODSU = codigo_sucursal_input
            AND (v.DENOMINACION_ORIGEN = 'Rioja' OR v.DENOMINACION_ORIGEN = 'Albariño')
        GROUP BY
            c.NOMBRE, c.TIPO
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('Nombre del Cliente: ' || cliente_info."Nombre del Cliente" ||
                             ', Tipo de Cliente: ' || cliente_info."Tipo de Cliente" ||
                             ', Cantidad Total Suministrada: ' || cliente_info."Cantidad Total Suministrada");
    END LOOP;
END;
```

Bibliografía

[1] Guión de la práctica.
[2] Página de PRADO de la asignatura