Ejercicio – SQL + Propuesta:

Carrier

CarrierID	Name	Capacity
1	Carrier A	10000
2	Carrier B	10000
3	Carrier C	3000

Cantidad de envíos.

Zona	Mes	Cantidad de envíos
AMBA	1	40000
Bs As	1	50000
Resto	1	60000

COSTOS

CarrierID	Zona	Costo	Tiempo de Entrega
1	AMBA	$10	3
1	Bs As	$20	5
1	Resto	$50	7
2	AMBA	$15	2
2	Bs As	$19	4
2	Resto	$55	6
3	AMBA	$20	1

- El costo total del mes 1 y por zona es:
  - AMBA: 1.800.000
  - Bs As: 1.950.000
  - Resto: 6.300.000

Query utilizada:

```
USE Voolkia

GO


create function func_GetCantidadEnvios(@mes int, @zona varchar(50))
returns int
as
	begin return (
		SELECT
			Voolkia.dbo.Cantidad_de_envios.Cantidad_de_envios
			AS CANTIDAD_ENVIOS
			FROM Voolkia.dbo.Cantidad_de_envios
			WHERE Voolkia.dbo.Cantidad_de_envios.Mes=@mes AND
Voolkia.dbo.Cantidad_de_envios.Zona=@zona
		);
end

GO


create function func_CalcularCostoTotal(@zona varchar(30), @cantidad_envios int)
returns int
as
	begin return (
		SELECT
```

```
                    SUM(CONVERT(INT, SUBSTRING(Voolkia.dbo.Costos.Costo, 2, 3))) *
@cantidad_envios
                    AS COSTO_AMBA
                    FROM Voolkia.dbo.Costos
                    WHERE Voolkia.dbo.Costos.Zona = @zona
                );
end

GO


DECLARE @Cantidad_envios INT
DECLARE @Costo_total_por_zona int

SET @Cantidad_envios = Voolkia.dbo.func_GetCantidadEnvios(1, 'AMBA');
SET @Costo_total_por_zona    = Voolkia.dbo.func_CalcularCostoTotal('AMBA',
@Cantidad_envios);

PRINT 'AMBA ' + CONVERT(varchar(30), @Costo_total_por_zona)

SET @Cantidad_envios = Voolkia.dbo.func_GetCantidadEnvios(1, 'Bs As');
SET @Costo_total_por_zona    = Voolkia.dbo.func_CalcularCostoTotal('Bs As',
@Cantidad_envios);

PRINT 'Bs As ' + CONVERT(varchar(30), @Costo_total_por_zona)

SET @Cantidad_envios = Voolkia.dbo.func_GetCantidadEnvios(1, 'Resto');
SET @Costo_total_por_zona    = Voolkia.dbo.func_CalcularCostoTotal('Resto',
@Cantidad_envios);

PRINT 'Resto ' + CONVERT(varchar(30), @Costo_total_por_zona)
```

Ejercicio – Lectura y comprensión Script Básico grails

Lectura y comprensión Script básico grails

- A tu entender, que se busca obtener como output del script?

```
import me.*;
def upsPullTrkService = ctx.getBean('upsPullTrkService')
def s = Shipment.get(27528954729)
def tn = s.trackingNumber
def trackingData = upsPullTrkService.getTrkEvents([tn])
trackingData.each { td ->
  println "-----------------------------------------------"
  println "${td.sucursal} - ${td.eventDate} - ${td.description}"
}
"Done"
```

- Se busca imprimir por pantalla cada sucursal, fecha de evento y descripción por cada dato de seguimiento del envió que se obtuvo anteriormente (Shipment.get(<id>)). Una vez terminado el recorrido, imprime “Done”.

Ejercicio - Script básico bash:

```
#!/bin/bash
users_id=(71665538 66146765 132961968 15096445 172753273 54152646)
for users_id in ${users_id[*]}
do
        curl=$(curl -s "api.mercadolibre.com/users/$users_id/shipping_preferences" | jq -c
'.services')
        echo "$users_id: $curl"
done
```

a- Obtiene las preferencias de envió de los usuarios especificados

b- Lineas:
   1- Se indica que el script debe ser interpretado por un Shell de bash
   2- Se declara un **ARRAY** con ids de usuarios
   3- Se declara un ciclo **FOR**
   4- Da inicio a la sentencia del ciclo **FOR** con la palabra reservada **DO**
   5- Se le asigna en una **VARIABLE** llamada curl la respuesta de la petición que se realiza a través de curl. En esta línea misma con “ **| jq –c ‘.services’** “ obtenemos lo que contenga la propiedad services del objeto de respuesta.
   6- Imprime por pantalla el id del usuario + la respuesta de la petición asignada en la línea anterior
   7- Una vez finalizado el ciclo, se cierra la sentencia con la palabra reservada **DONE**

c- Se imprimen 6 lineas. (La longitud de “**users_id**”)