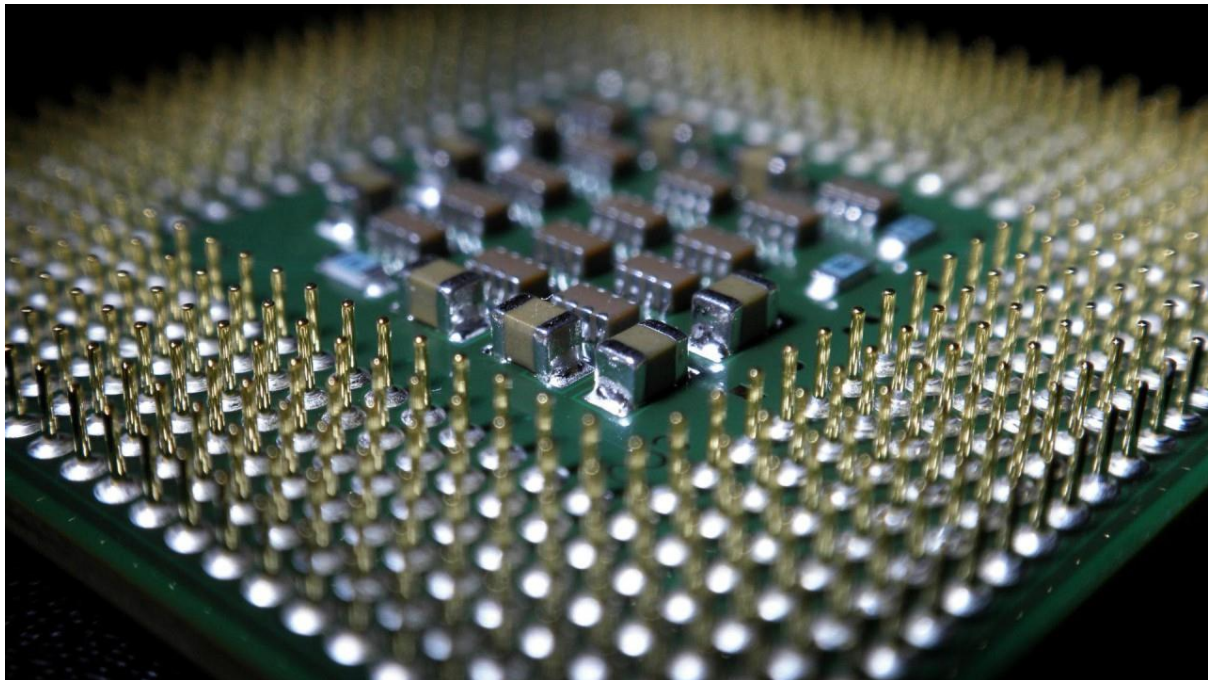


UNIVERSIDAD AUTÓNOMA DE CHIHUAHUA

CÓMPUTO PARALELO Y DISTRIBUIDO



Proyecto 2a. Evaluación



Nombres:

331206 - Miguel Ángel Abundis Medina

353198 - Diego Alejandro Martínez González

348411 - Ramón Reyna García

Profesor: De Lira Montes Jose Saul




Link Repositorio Github:

<https://github.com/DiegoMTZGlz/proy2CPD>

Scripts:

Para empezar tenemos dos contenedores, ambos con la base de datos de ORACLE, los cuales están dentro de una red de docker llamada oracle_net

```
PS C:\Users\aspir> docker network create oracle_net
1aa606d4348d13363109b504758f7e9a461beabdb16d6a31377b42c8bdb5b031
PS C:\Users\aspir> docker images
```

Image	Server-B	Server-A
		
7318dc7b65a1	99d7de4b9a8	
container-registry.o	container-registry.o	
Running	Running	
N/A	N/A	
1522:1521	1521:1521	
1 second ago	0 seconds ago	
■	■	
:	:	
■	■	

El primer script a ejecutar fue el de la base de datos OE, el cual tal cual es corrido en la base de datos del servidor A



BDD_SOE.sql

luego creamos un datalink hacia el server B

```
CREATE DATABASE LINK SBL
CONNECT TO C##SERVERB IDENTIFIED BY ADMIN
USING '172.20.0.3:1521/FREE';
```

luego fragmentamos la tabla de customers creando otra tabla que contenga solo los de región A y B

```
CREATE TABLE customers_fragmented_db1
PARTITION BY RANGE (REGION) (
  PARTITION region_a VALUES LESS THAN ('C'),
  PARTITION region_b VALUES LESS THAN (MAXVALUE)
)
AS SELECT * FROM customers WHERE REGION IN ('A', 'B');
```

creamos la vista para mostrar todos los customers tanto del servidor A como del B

```
CREATE OR REPLACE VIEW global_customers AS
SELECT * FROM customers_db2;
UNION ALL
SELECT * FROM CUSTOMERS_FRAGMENTED_DB1;
```

al igual que en el servidor A en el B ejecutamos el script OE pero con la diferencia que la única operación extra es crear su fragment con los customers de región C y D

```
CREATE TABLE customers_fragmented_db2
PARTITION BY RANGE (REGION) (
  PARTITION region_c VALUES LESS THAN ('D'),
  PARTITION region_d VALUES LESS THAN (MAXVALUE)
)
AS SELECT * FROM customers WHERE REGION IN ('C', 'D');
```

A partir de ahora todos los script son creados en el servidor A, de los cuales empezamos con la capa de transparencia:

```
CREATE SYNONYM customers_db2 FOR customers_fragmented_db2@SBL;
CREATE SYNONYM order_items_db2 FOR order_items@SBL;
CREATE SYNONYM orders_db2 FOR orders@SBL;
CREATE SYNONYM product_information_db2 FOR product_information@SBL;
```

Creamos los triggers los cuales al ejecutarse una operación en el servidor A se replicará en el servidor B a excepción de la tabla de customers

```
L > TRIGGERS-A.sql
1 CREATE OR REPLACE TRIGGER replicate_orders_insert
2 AFTER INSERT ON orders
3 FOR EACH ROW
4 BEGIN
5   INSERT INTO orders_db2 (
6     ORDER_ID, ORDER_DATE, ORDER_MODE, CUSTOMER_ID,
7     ORDER_STATUS, ORDER_TOTAL, SALES_REP_ID, PROMOTION_ID
8   ) VALUES (
9     :NEW.ORDER_ID, :NEW.ORDER_DATE, :NEW.ORDER_MODE, :NEW.CUSTOMER_ID,
10    :NEW.ORDER_STATUS, :NEW.ORDER_TOTAL, :NEW.SALES_REP_ID, :NEW.PROMOTION_ID
11  );
12 END replicate_orders_insert;
13 /
```

```

CREATE OR REPLACE TRIGGER replicate_orders_update
AFTER UPDATE ON orders
FOR EACH ROW
BEGIN
    UPDATE orders_db2 SET
        ORDER_DATE = :NEW.ORDER_DATE,
        ORDER_MODE = :NEW.ORDER_MODE,
        CUSTOMER_ID = :NEW.CUSTOMER_ID,
        ORDER_STATUS = :NEW.ORDER_STATUS,
        ORDER_TOTAL = :NEW.ORDER_TOTAL,
        SALES_REP_ID = :NEW.SALES_REP_ID,
        PROMOTION_ID = :NEW.PROMOTION_ID
    WHERE ORDER_ID = :OLD.ORDER_ID;
END replicate_orders_update;
/

```

```

CREATE OR REPLACE TRIGGER replicate_orders_delete
AFTER DELETE ON orders
FOR EACH ROW
BEGIN
    DELETE FROM orders_db2 WHERE ORDER_ID = :OLD.ORDER_ID;
END replicate_orders_delete;
/

```

```

CREATE OR REPLACE TRIGGER replicate_orders_delete
AFTER DELETE ON orders
FOR EACH ROW
BEGIN
    DELETE FROM orders_db2 WHERE ORDER_ID = :OLD.ORDER_ID;
END replicate_orders_delete;
/

CREATE OR REPLACE TRIGGER replicate_order_items_insert
AFTER INSERT ON order_items
FOR EACH ROW
BEGIN
    INSERT INTO order_items_db2 (
        ORDER_ID, LINE_ITEM_ID, PRODUCT_ID, UNIT_PRICE, QUANTITY
    ) VALUES (
        :NEW.ORDER_ID, :NEW.LINE_ITEM_ID, :NEW.PRODUCT_ID, :NEW.UNIT_PRICE, :NEW.QUANTITY
    );
END replicate_order_items_insert;
/

```

Después tenemos el script que crea los procedimientos almacenados, que es básicamente el CRUD para cada tabla que tenemos exceptuando customers:

```
L > CRUDDOrders-A.sql
1  CREATE OR REPLACE PROCEDURE create_order(
2      p_order_id IN NUMBER,
3      p_order_date IN TIMESTAMP,
4      p_order_mode IN VARCHAR2,
5      p_customer_id IN NUMBER,
6      p_order_status IN NUMBER,
7      p_order_total IN NUMBER,
8      p_sales_rep_id IN NUMBER,
9      p_promotion_id IN NUMBER
10 )
11 IS
12 BEGIN
13     INSERT INTO orders (
14         ORDER_ID, ORDER_DATE, ORDER_MODE, CUSTOMER_ID,
15         ORDER_STATUS, ORDER_TOTAL, SALES_REP_ID, PROMOTION_ID
16     ) VALUES (
17         p_order_id, p_order_date, p_order_mode, p_customer_id,
18         p_order_status, p_order_total, p_sales_rep_id, p_promotion_id
19     );
20     COMMIT;
21 END create_order;
22 /
23
24 CREATE OR REPLACE FUNCTION read_order(
25     p_order_id IN NUMBER
26 ) RETURN SYS_REFCURSOR
27 IS
28     v_cursor SYS_REFCURSOR;
29 BEGIN
30     OPEN v_cursor FOR
31         SELECT * FROM orders WHERE ORDER_ID = p_order_id;
32     RETURN v_cursor;
33 END read_order;
34 /
```

```

CREATE OR REPLACE PROCEDURE list_orders(
    p_cursor OUT SYS_REFCURSOR
)
IS
BEGIN
    OPEN p_cursor FOR
        SELECT * FROM orders;
END list_orders;
/

CREATE OR REPLACE PROCEDURE update_order(
    p_order_id IN NUMBER,
    p_order_date IN TIMESTAMP,
    p_order_mode IN VARCHAR2,
    p_customer_id IN NUMBER,
    p_order_status IN NUMBER,
    p_order_total IN NUMBER,
    p_sales_rep_id IN NUMBER,
    p_promotion_id IN NUMBER
)
IS
BEGIN
    UPDATE orders SET
        ORDER_DATE = p_order_date,
        ORDER_MODE = p_order_mode,
        CUSTOMER_ID = p_customer_id,
        ORDER_STATUS = p_order_status,
        ORDER_TOTAL = p_order_total,
        SALES_REP_ID = p_sales_rep_id,
        PROMOTION_ID = p_promotion_id
    WHERE ORDER_ID = p_order_id;
    COMMIT;
END update_order;
/

```

etc.

Para seguir con el CRUD de customers qué es un poco diferente ya que contiene condicionales para saber si se almacenará un usuario en el servidor A o B, así como su edición:

```
CREATE OR REPLACE PROCEDURE create_customer(  
    p_customer_id IN NUMBER,  
    p_cust_first_name IN VARCHAR2,  
    p_cust_last_name IN VARCHAR2,  
    p_credit_limit IN NUMBER,  
    p_cust_email IN VARCHAR2,  
    p_income_level IN VARCHAR2,  
    p_region IN VARCHAR2  
)  
IS  
BEGIN  
    IF p_region IN ('A', 'B') THEN  
        INSERT INTO customers_fragmented_db1 (  
            CUSTOMER_ID, CUST_FIRST_NAME, CUST_LAST_NAME,  
            CREDIT_LIMIT, CUST_EMAIL, INCOME_LEVEL, REGION  
        ) VALUES (  
            p_customer_id, p_cust_first_name, p_cust_last_name,  
            p_credit_limit, p_cust_email, p_income_level, p_region  
        );  
    ELSIF p_region IN ('C', 'D') THEN  
        INSERT INTO customers_db2 (  
            CUSTOMER_ID, CUST_FIRST_NAME, CUST_LAST_NAME,  
            CREDIT_LIMIT, CUST_EMAIL, INCOME_LEVEL, REGION  
        ) VALUES (  
            p_customer_id, p_cust_first_name, p_cust_last_name,  
            p_credit_limit, p_cust_email, p_income_level, p_region  
        );  
    ELSE  
        RAISE_APPLICATION_ERROR(-20001, 'Región no válida. Las regiones válidas son A, B, C o D.');
```

así sucesivamente con todas las operaciones CRUD

Funcionamiento:

tenemos una ventana la cual contiene varias opciones, en ellas tenemos las 4 tablas al superior y debajo sus operaciones crud correspondientes

Proyecto 2

CUSTOMERS ORDERS ITEMS PRODUCTS

AGREGAR (+) BUSCAR (?) ACTUALIZAR (*) ELIMINAR (-)

ID Cliente:

Nombre:

Apellido:

Crédito:

Correo:

Ingresos:

Región:

CREAR LIMPIAR

en el apartado de BUSCAR tenemos toda la lista de registros de la tabla en la base de datos

Proyecto 2

CUSTOMERS ORDERS ITEMS PRODUCTS

AGREGAR (+) BUSCAR (?) ACTUALIZAR (*) ELIMINAR (-)

ID	Nombre	Descripción	Categoría	Peso	Garantía	ID Proveedor	Estatus	Precio	Precio Mi	URL
1787	CPU D300	Dual CPU @ 300Mh	15	1	3 0	102097	orderable	101	90.0	http://www.supp-102
2439	CPU D400	Dual CPU @ 400Mh	15	1	3 0	102092	orderable	123	105.0	http://www.supp-102
1788	CPU D600	Dual CPU @ 600Mh	15	1	5 0	102067	orderable	178	149.0	http://www.supp-102
2375	GP 1024x768	Graphics Processor,	15	1	0 9	102063	orderable	78	69.0	http://www.supp-102
2411	GP 1280x1024	Graphics Processor,	15	1	1 0	102061	orderable	98	78.0	http://www.supp-102
1769	GP 800x600	Graphics processor,	15	1	0 6	102050	orderable	48	None	http://www.supp-102
2049	MB - S300	PC type motherboar	15	2	1 0	102082	obsolete	55	47.0	http://www.supp-102
2751	MB - S450	PC type motherboar	15	2	1 0	102072	orderable	66	54.0	http://www.supp-102
3112	MB - S500	PC type motherboar	15	2	1 6	102086	orderable	77	66.0	http://www.supp-102
2752	MB - S550	PC type motherboar	15	2	1 6	102086	orderable	88	76.0	http://www.supp-102

Cargar/Actualizar Productos

para ACTUALIZAR tenemos un campo de ID el cual ingresamos un id a modificar y en el botón buscar nos llenará los otros campos con la información de ese id

The screenshot shows a web application window titled 'Proyecto 2'. At the top, there are four tabs: 'CUSTOMERS', 'ORDERS', 'ITEMS', and 'PRODUCTS'. Below these tabs are four buttons: 'AGREGAR (+)', 'BUSCAR (?)', 'ACTUALIZAR (*)', and 'ELIMINAR (-)'. The 'ACTUALIZAR (*)' button is highlighted. The form contains the following fields and values:

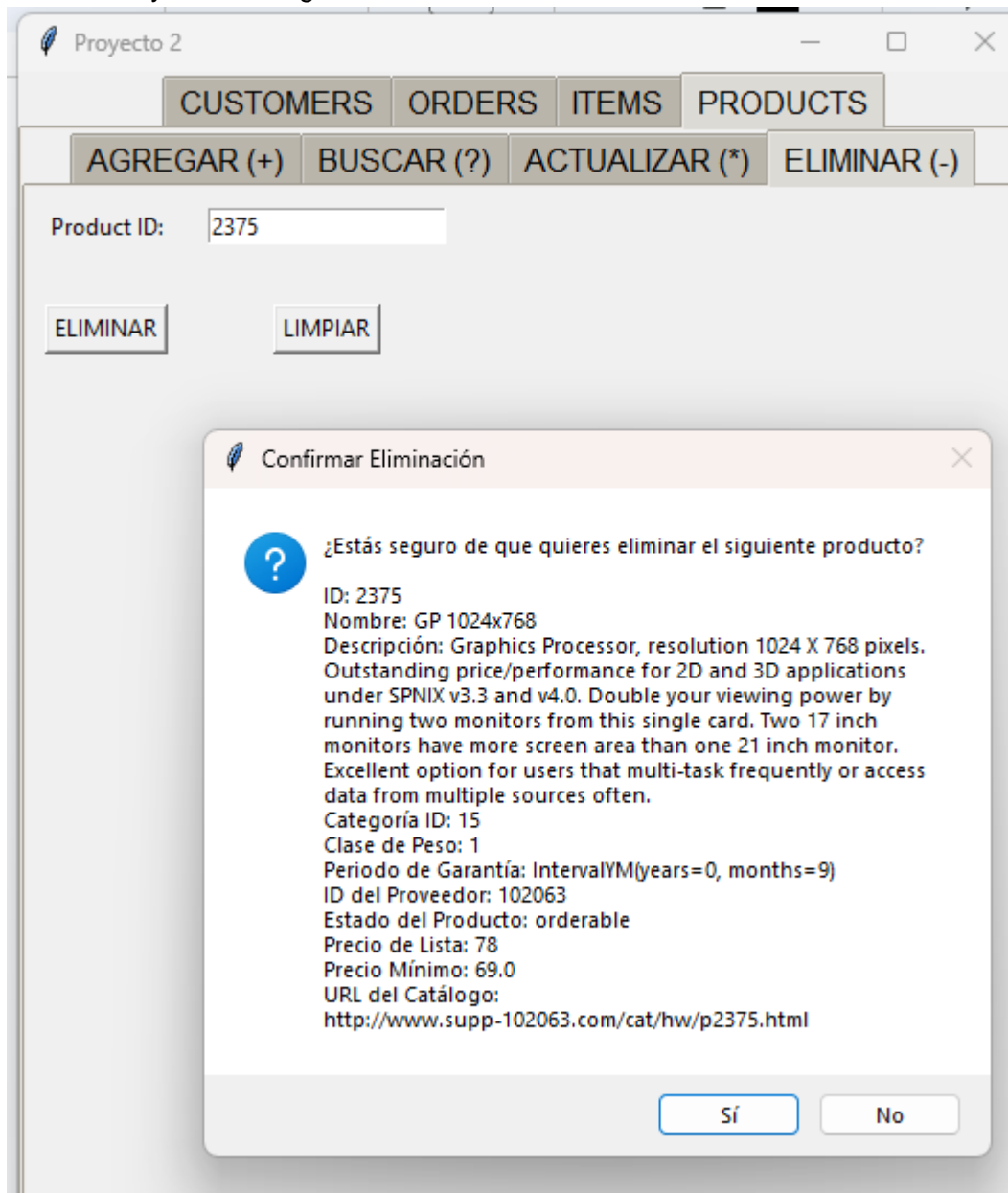
Field	Value
Product ID:	2375
Product Name:	GP 1024x768
Product Description:	Graphics Processor, res
Category ID:	15
Weight Class:	[Dropdown menu]
Warranty Period:	0 9
Supplier ID:	102063
Product Status:	orderable
List Price:	78
Min Price:	69.0
Catalog URL:	http://www.supp-1020

At the bottom of the form, there are two buttons: 'ACTUALIZAR' and 'LIMPIAR'.

por último para eliminar tenemos solo el campo de ID, con el cuál buscará el usuario

The screenshot shows the same web application window, but now the 'ELIMINAR (-)' button is highlighted. The form is simplified, containing only the 'Product ID:' field, which is currently empty. Below the field are two buttons: 'ELIMINAR' and 'LIMPIAR'.

Al querer eliminar nos saldrá un recuadro con los datos del campo a eliminar, de no ser así saltará la leyenda de registro no encontrado



Las mismas operaciones son similares para las otras tablas solo cambiando algunos inputs

Todos los campos han sido validados con sus correspondientes tipo de datos y longitud