

BDOR GESTIÓN DE VENTAS

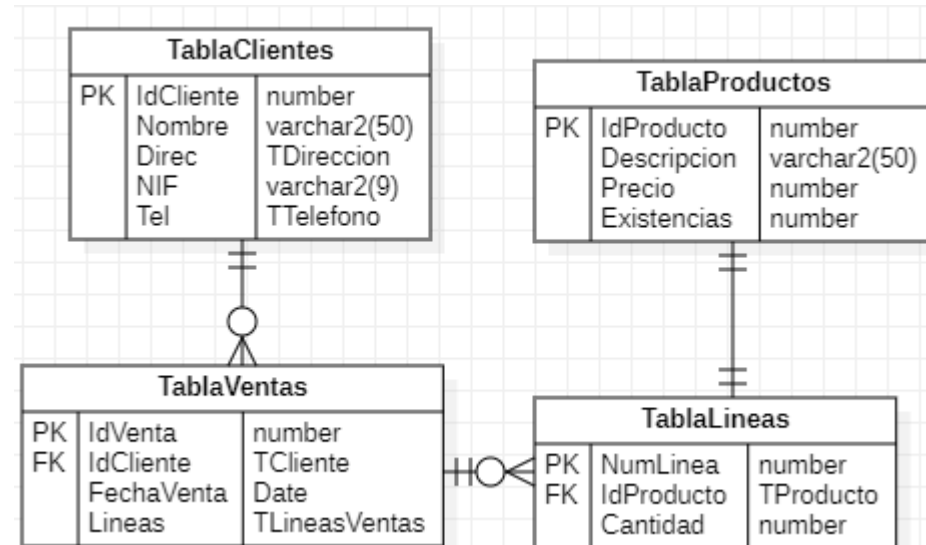
Alberto Colmenar

La práctica consiste en crear un modelo de bases de datos objeto-relacional para la gestión de ventas.

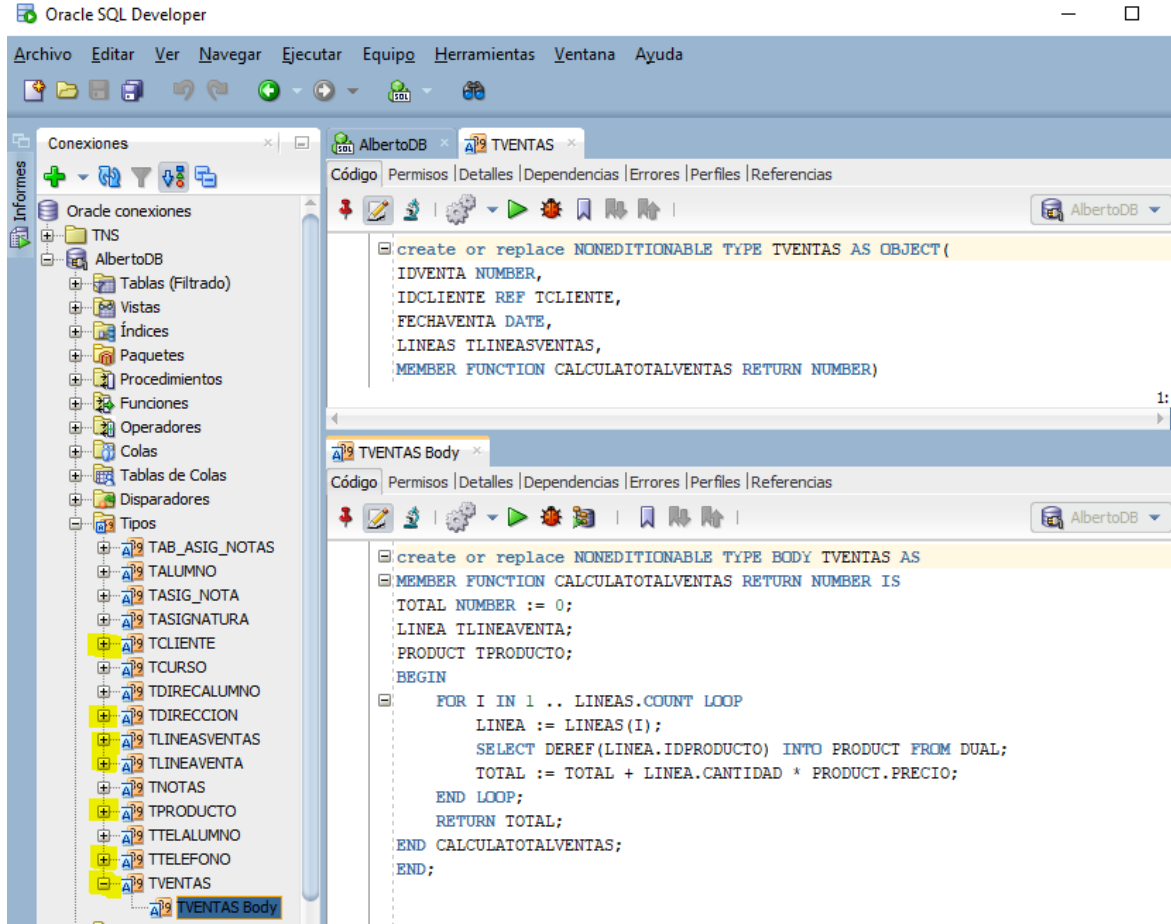
Inicialmente creamos los tipos y las tablas (como se muestra en el diagrama).
El diagrama muestra los atributos y como se relacionan las distintas tablas. Los clientes hacen varias ventas, las ventas tienen líneas (tabla anidada) y una línea tiene un producto.

Introducimos datos en las tablas e hicimos unas consultas para probarlas.

Al final de la práctica creamos distintos procedimientos y funciones con la finalidad de mostrar distintos datos, especificados por parámetro, de las tablas.



Practica 4.1 Tipos creados para la BDOR Gestion de ventas.

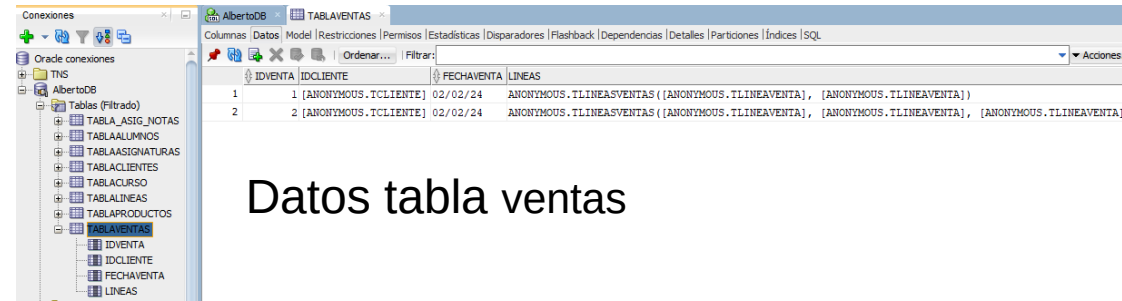


Creación del tipo ventas

Cuerpo de la función calculatotalventas

Tipos creados en amarillo

Práctica 4.1 Datos almacenados en las tablas.



Conexiones

Oracle conexiones

TNS

AlbertoDB

Tablas (Filtrado)

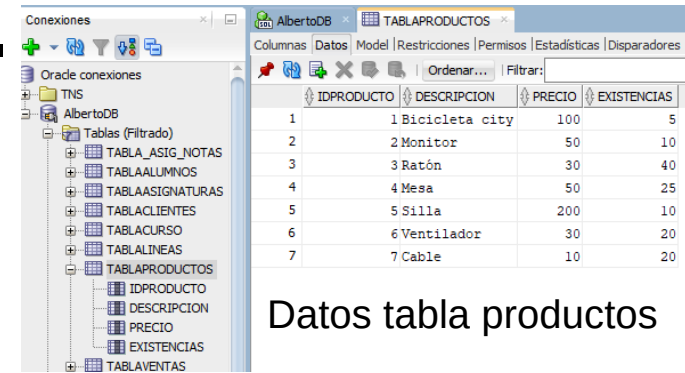
- TABLA_ASIG_NOTAS
- TABLAALUMNOS
- TABLAASIGNATURAS
- TABLACLIENTES
- TABLACURSO
- TABLALINEAS
- TABLAPRODUCTOS
- TABLAVENTAS**
- IDVENTA
- IDCLIENTE
- FECHAVENTA
- LINEAS

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores | Flashback | Dependencias | Detalles | Particiones | Índices | SQL

Ordenar... | Filtrar: | Acciones...

IDVENTA	IDCLIENTE	FECHAVENTA	LINEAS
1	[ANONYMOUS.TCLIENTE]	02/02/24	ANONYMOUS.TLINEASVENTAS ([ANONYMOUS.TLINEAVENTA], [ANONYMOUS.TLINEAVENTA])
2	[ANONYMOUS.TCLIENTE]	02/02/24	ANONYMOUS.TLINEASVENTAS ([ANONYMOUS.TLINEAVENTA], [ANONYMOUS.TLINEAVENTA])

Datos tabla ventas



Conexiones

Oracle conexiones

TNS

AlbertoDB

Tablas (Filtrado)

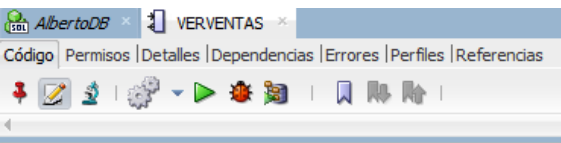
- TABLA_ASIG_NOTAS
- TABLAALUMNOS
- TABLAASIGNATURAS
- TABLACLIENTES
- TBLACURSO
- TABLALINEAS
- TABLAPRODUCTOS**
- IDPRODUCTO
- DESCRIPCION
- PRECIO
- EXISTENCIAS

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores

Ordenar... | Filtrar:

IDPRODUCTO	DESCRIPCION	PRECIO	EXISTENCIAS
1	Bicicleta city	100	5
2	Monitor	50	10
3	Ratón	30	40
4	Mesa	50	25
5	Silla	200	10
6	Ventilador	30	20
7	Cable	10	20

Datos tabla productos



AlbertoDB x VERVENTAS x

Código | Permisos | Detalles | Dependencias | Errores | Perfiles | Referencias

Conectando a la base de datos AlbertoDB.

Ejecutando: IdeConnections%23AlbertoDB.jpr - Log

Conectando a la base de datos AlbertoDB.

NUMERO DE VENTA: 1 FECHA DE VENTA: 02/02/24

NOMBRE DEL CLIENTE: Luis García

DIRECCION: C/Dalias,22

1-Bicicleta city: 100 * 3 = 300

2-Monitor: 50 * 1 = 50

TOTAL VENTA: 350

El proceso ha terminado.

Desconectando de la base de datos AlbertoDB.

Datos tabla anidada de ventas usando el procedimiento del ejercicio 9 de esta práctica



Conexiones

Oracle conexiones

TNS

AlbertoDB

Tablas (Filtrado)

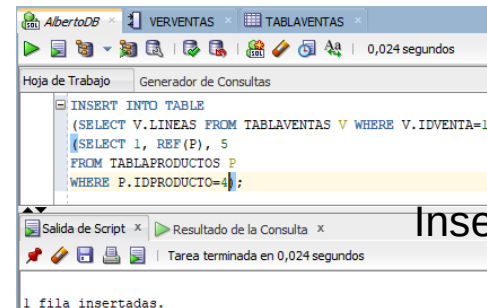
- TABLA_ASIG_NOTAS
- TABLAALUMNOS
- TABLAASIGNATURAS
- TABLACLIENTES**
- IDCLIENTE
- NOMBRE
- DIREC
- NIF
- TEL

Columnas | Datos | Model | Restricciones | Permisos | Estadísticas | Disparadores | Flashback | Dependencias | Detalles | Particiones | Índices | SQL

Ordenar... | Filtrar:

IDCLIENTE	NOMBRE	DIREC	NIF	TEL
1	Luis García	[ANONYMOUS.TDIRECCION]	34343434L	ANONYMOUS.TTELEFONO('950234577', '67899
2	Roberto García	[ANONYMOUS.TDIRECCION]	5656565L	ANONYMOUS.TTELEFONO('59634577', '463290

Datos tabla clientes



AlbertoDB x VERVENTAS x TABLAVENTAS x

0,024 segundos

Hoja de Trabajo | Generador de Consultas

```
INSERT INTO TABLE
(SELECT V.LINEAS FROM TABLAVENTAS V WHERE V.IDVENTA=1)
(SELECT 1, REP(P), 5
FROM TABLAPRODUCTOS P
WHERE P.IDPRODUCTO=4);
```

Salida de Script x Resultado de la Consulta x

Tarea terminada en 0,024 segundos

1 fila insertadas.

Insertión una venta

Practica 4.1 Consultas sobre las tablas de la BDOR

Selecciona número de línea, la descripción del producto, la cantidad pedida y el precio de todos los productos vendidos para la venta con idventa = 2

The screenshot shows the AlbertoDB interface with a SQL query in the 'Generador de Consultas' tab. The query is: `SELECT NUMLINEA, DEREf(IDPRODUCTO).DESCRIPCION, CANTIDAD, DEREf(IDPRODUCTO).PRECIO FROM THE (SELECT V.LINEAS FROM TABLAVENTAS V WHERE V.IDVENTA = 2)`. The 'Salida de Script' tab shows the results in a table format.

NUMLINEA	DEREF(IDPRODUCTO).DESCRIPCION	CANTIDAD	DEREF(IDPRODUCTO).PRECIO
2	Mesa	3	50
3	Silla	7	200
1	Bicicleta city	1	100

Selecciona el nombre, calle, población, codpostal de todos los clientes de la provincia de Madrid.

The screenshot shows the AlbertoDB interface with a SQL query in the 'Generador de Consultas' tab. The query is: `SELECT C.DIREC.CALLE, C.DIREC.POBLACION, C.DIREC.CODPOSTAL FROM TABLACLIENTES C WHERE C.DIREC.POBLACION = 'Madrid'`. The 'Salida de Script' tab shows the message: 'no se ha seleccionado ninguna fila'.

Como no tenía clientes de Madrid he hecho la misma query para Sevilla

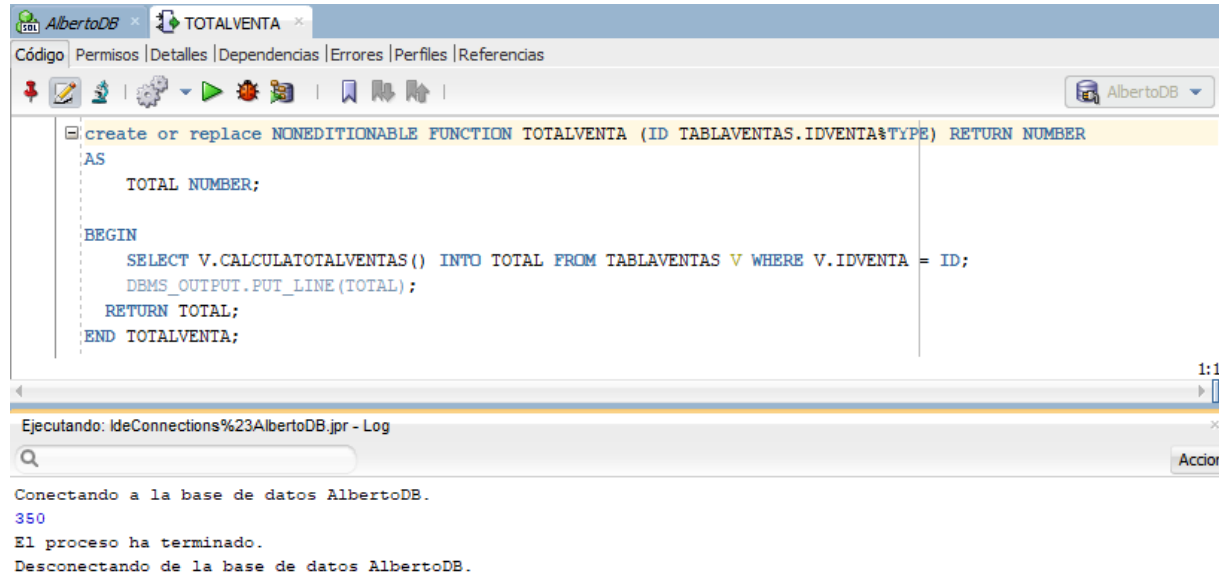
The screenshot shows the AlbertoDB interface with a SQL query in the 'Generador de Consultas' tab. The query is: `SELECT C.DIREC.CALLE, C.DIREC.POBLACION, C.DIREC.CODPOSTAL FROM TABLACLIENTES C WHERE C.DIREC.POBLACION = 'Sevilla'`. The 'Salida de Script' tab shows the results in a table format.

DIREC.CALLE	DIREC.POBLACION	DIREC.CODPOSTAL
C/Dalias, 22	Sevilla	44330

Práctica 4.1 Código y ejecución de la función del ej. 11

11. Crea una función que reciba un identificador de venta y retorne el total de la venta. Realiza un bloque PL/SQL anónimo que haga uso de la función.

Utiliza la función `calculatotalventas()` de la tabla `ventas`. Esta función calcula el total pero como nos interesa solo las ventas del id pasado por parámetro se lo escribimos en el `where` de nuestra `select`. El resultado lo metemos en una variable previamente creada, la mostramos y la devolvemos con el `return`.



The screenshot displays the AlbertoDB IDE interface. The main editor window shows the following PL/SQL code:

```
create or replace NONEDITIONABLE FUNCTION TOTALVENTA (ID TABLAVENTAS.IDVENTA%TYPE) RETURN NUMBER
AS
    TOTAL NUMBER;

BEGIN
    SELECT V.CALCULATOTALVENTAS() INTO TOTAL FROM TABLAVENTAS V WHERE V.IDVENTA = ID;
    DBMS_OUTPUT.PUT_LINE(TOTAL);
    RETURN TOTAL;
END TOTALVENTA;
```

Below the code editor, the execution log is visible, showing the following messages:

```
Ejecutando: ldeConnections%23AlbertoDB.jpr - Log
Conectando a la base de datos AlbertoDB.
350
El proceso ha terminado.
Desconectando de la base de datos AlbertoDB.
```

Práctica 4.1 Código y ejecución del procedimiento ej. 14

14. Crea una función que devuelva el total de artículos vendidos de un artículo determinado cuyo id sea un parámetro de entrada a la función.

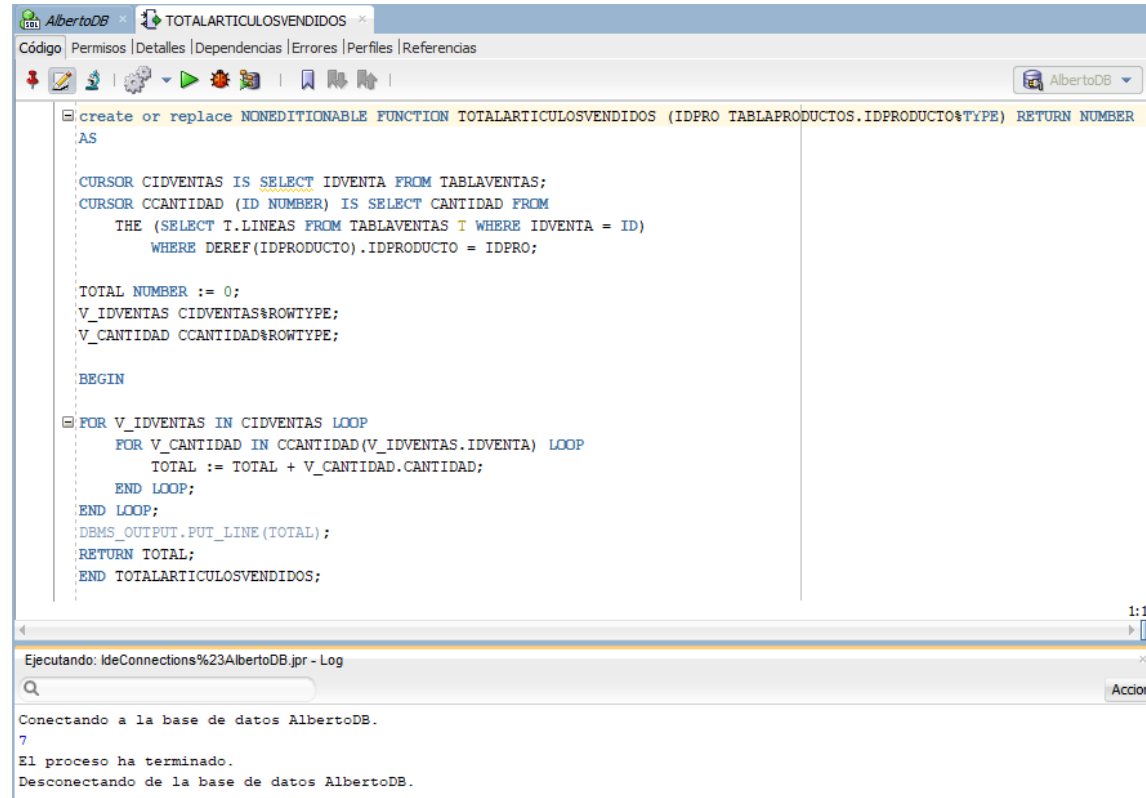
Guardamos en un cursor todas las idVenta de la tabla ventas para recorrerlo después.

Usamos otro cursor para guardar las cantidades del producto, que nos pasan como parámetro, de las ventas de la tabla anidada.

Después recorreremos el primer cursor para ir por todas las ventas y dentro el otro cursor para coger las cantidades e ir las sumando a una variable total.

Al final mostramos el total por consola y lo retornamos.

La prueba se realizó para el producto con id 5



The screenshot shows an IDE window titled 'AlbertoDB' with a tab for 'TOTALARTICULOSVENDIDOS'. The 'Código' (Code) tab is active, displaying the following PL/SQL code:

```
create or replace NONEDITIONABLE FUNCTION TOTALARTICULOSVENDIDOS (IDPRO TABLAPRODUCTOS.IDPRODUCTO%TYPE) RETURN NUMBER
AS
CURSOR CIDVENTAS IS SELECT IDVENTA FROM TABLAVENTAS;
CURSOR CCANTIDAD (ID NUMBER) IS SELECT CANTIDAD FROM
    THE (SELECT T.LINEAS FROM TABLAVENTAS T WHERE IDVENTA = ID)
    WHERE Deref(IDPRODUCTO).IDPRODUCTO = IDPRO;

TOTAL NUMBER := 0;
V_IDVENTAS CIDVENTAS%ROWTYPE;
V_CANTIDAD CCANTIDAD%ROWTYPE;

BEGIN
FOR V_IDVENTAS IN CIDVENTAS LOOP
    FOR V_CANTIDAD IN CCANTIDAD(V_IDVENTAS.IDVENTA) LOOP
        TOTAL := TOTAL + V_CANTIDAD.CANTIDAD;
    END LOOP;
END LOOP;
DEMS_OUTPUT.PUT_LINE(TOTAL);
RETURN TOTAL;
END TOTALARTICULOSVENDIDOS;
```

Below the code editor, a console window titled 'Ejecutando: IdeConnections%23AlbertoDB.jpr - Log' shows the execution output:

```
Conectando a la base de datos AlbertoDB.
7
El proceso ha terminado.
Desconectando de la base de datos AlbertoDB.
```

Práctica 4.1 Código y ejecución del procedimiento ej. 15

15. Crea un procedimiento que actualice el precio de un artículo determinado donde el id y el nuevo precio serán introducidos como parámetros de entrada.

Nos pasan por parámetro el id del producto que se quiere modificar y el nuevo precio. Estos parámetros los introducimos en un update y modificamos el precio. Esta prueba ha sido ejecutada para el producto con id 2 y nuevo precio de 100 como vemos en las imágenes de los datos.

The screenshot displays the SQL Server Enterprise Manager interface with three main windows:

- Left Window (TABLAPRODUCTOS):** A table with columns IDPRODUCTO, DESCRIPCION, PRECIO, and EXISTENCIAS. The data is as follows:

IDPRODUCTO	DESCRIPCION	PRECIO	EXISTENCIAS
1	1Bicicleta city	100	5
2	2Monitor	50	10
3	3Ratón	30	40
4	4Mesa	50	25
5	5Silla	200	10
6	6Ventilador	30	20
7	7Cable	10	20

An orange arrow points to the row where IDPRODUCTO is 2.
- Middle Window (MODIFICARPRECIO):** A script editor showing the following SQL code:

```
create or replace NONEDITIONABLE PROCEDURE MODIFICARPRECIO (ID TABLAPRODUCTOS.IDPRODUCTO%TYPE, NUEVOPRECIO TABLAPRODUCTOS.PRECIO%TYPE)
AS
BEGIN
    UPDATE TABLAPRODUCTOS
    SET PRECIO = NUEVOPRECIO
    WHERE IDPRODUCTO = ID;
END MODIFICARPRECIO;
```
- Right Window (TABLAPRODUCTOS):** The same table as the left window, but with the price for IDPRODUCTO 2 updated to 100. An orange arrow points to this row.

IDPRODUCTO	DESCRIPCION	PRECIO	EXISTENCIAS
1	1Bicicleta city	100	5
2	2Monitor	100	10
3	3Ratón	30	40
4	4Mesa	50	25
5	5Silla	200	10
6	6Ventilador	30	20
7	7Cable	10	20

At the bottom, a status bar shows the execution log:

```
Ejecutando: IdeConnections%23AlbertoDB.jpr - Log
Conectando a la base de datos AlbertoDB.
El proceso ha terminado.
Desconectando de la base de datos AlbertoDB.
```