

# DESARROLLO DE APLICACIONES CON BASES DE DATOS

Licenciatura en Informática  
Trabajo Práctico 4

**Prof. Titular Disciplinar: Silvia Laura Castelli**  
**Prof. Titular Experto: Ricardo Ramón Daubrowsky**  
**Alumno: Pablo Alejandro Hamann**  
**Legajo: VINF010782**  
**Año: 2025**

## Tabla de contenido

Introducción .....	1
Repositorio en GitHub .....	1
Consigna .....	1
Actividad 1: Crear una base de datos denominada DWPedidos .....	1
Actividad 2: Crear las siguientes tablas en la base de datos DWPedidos .....	2
Actividad 3: Cargar de las tablas .....	2
Sección 0: Tareas previas a la creación del esquema y configuración del entorno .....	2
Borrar esquema (usuario) si existiera .....	2
Crear el esquema y darle privilegios .....	3
Definición de funciones personalizadas .....	3
Asignación de permisos .....	4
Sección 1: Sentencias de creación de la base de datos DWPedidos .....	4
Modelado dimensional y el modelo estrella .....	4
Dimensión de Fechas .....	4
Dimensión de Productos .....	5
Dimensión de Clientes .....	5
Tabla de hechos de Pedidos .....	5
Sección 2: Detalle de proced. almacenados p/carga de datos en DWPedidos (Proceso ETL) .....	5
Procedimiento para cargar DIMFechas .....	6
Procedimiento para cargar DIMClientes .....	6
Procedimiento para cargar DIMProductos .....	7
Procedimiento para cargar FACTPedidos .....	7
Procedimiento ejecutar_ETLMaestroDW .....	8
Ejecución del procedimiento ETL Maestro .....	8
Sección 3: Extra .....	8
Captura de pantalla primera ejecución de sentencias SQL .....	8
Captura de pantalla segunda ejecución de sentencias SQL .....	9
Captura de pantalla de los datos en DWPEDIDOS.DIMClientes .....	10
Captura de pantalla de los datos en DWPEDIDOS.DIMFechas .....	10
Captura de pantalla de los datos en DWPEDIDOS.DIMProductos .....	10
Captura de pantalla de los datos en DWPEDIDOS.FACTPedidos .....	10

## Introducción

Este documento corresponde al desarrollo de las consignas planteadas en el Trabajo Práctico 4, y retoma a partir de las tareas realizadas en todos los TPs anteriores.

## Repositorio en GitHub

Todo lo producido, tanto para este presente TP, como para el anterior, se encuentra en un repositorio en GitHub creado para el cursado de esta materia. Allí, se mantienen actualizadas tanto las actividades prácticas como los TPs y cualquier otro tipo de actividad que implique desarrollo (de documentación, programación, etc.), que se dé durante el cursado de la materia. El repositorio se puede acceder mediante el siguiente enlace:

---

<https://github.com/linkstat/dabd/tree/main>

---

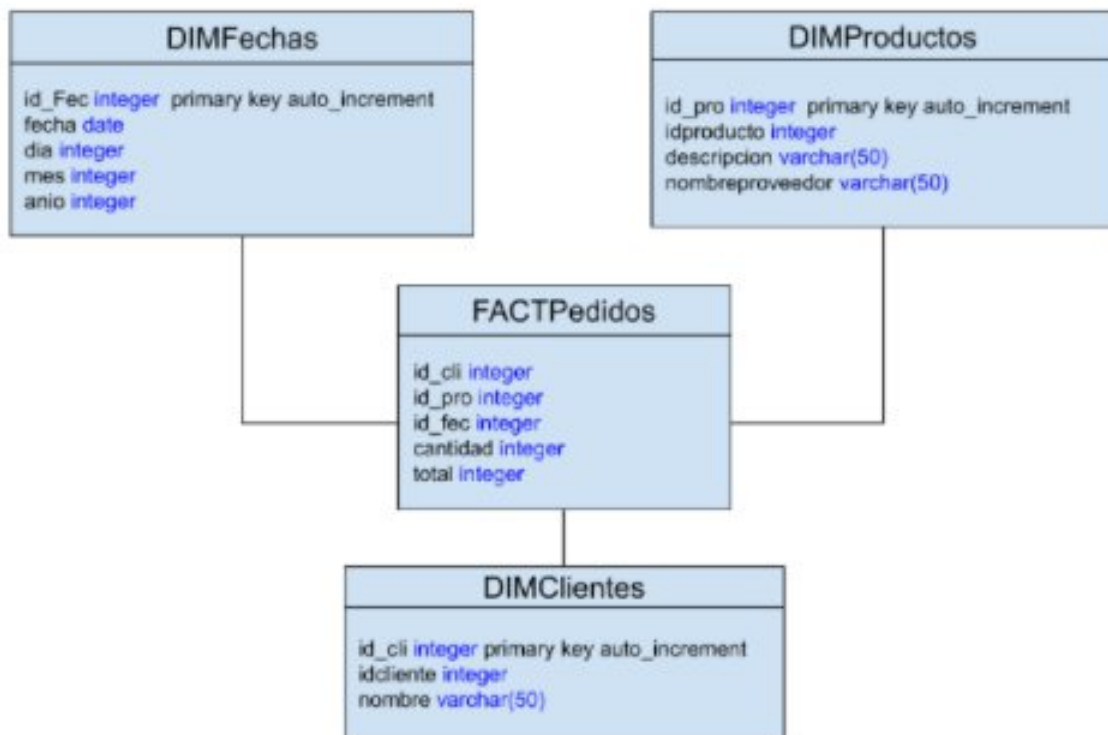
Archivos principales del proyecto **dabd** (este proyecto):

- Este documento en formato PDF:
  - <https://github.com/linkstat/dabd/raw/refs/heads/main/docs/HAMANN-PABLO-ALEJANDRO-TP4.pdf>
- Este documento en formato DOCX de Word:
  - <https://github.com/linkstat/dabd/raw/refs/heads/main/docs/HAMANN-PABLO-ALEJANDRO-TP4.docx>
- Archivo de script SQL que debe ejecutar el usuario SYSTEM (DBO):
  - [https://raw.githubusercontent.com/linkstat/dabd/refs/heads/main/sql/HAMANN-PABLO-ALEJANDRO-TP4-\[SYSTEM\].sql](https://raw.githubusercontent.com/linkstat/dabd/refs/heads/main/sql/HAMANN-PABLO-ALEJANDRO-TP4-[SYSTEM].sql)
- Archivo de script SQL que debe ejecutar el usuario DWPEDIDOS (DW):
  - [https://raw.githubusercontent.com/linkstat/dabd/refs/heads/main/sql/HAMANN-PABLO-ALEJANDRO-TP4-\[DWPEDIDOS\].sql](https://raw.githubusercontent.com/linkstat/dabd/refs/heads/main/sql/HAMANN-PABLO-ALEJANDRO-TP4-[DWPEDIDOS].sql)

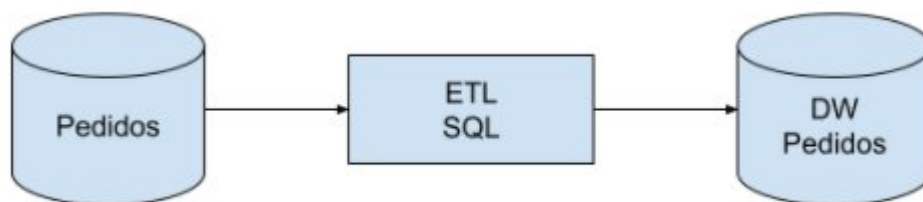
## Consigna

El objetivo de este trabajo individual consiste en desarrollar un proceso ETL (extracción, transformación y carga) utilizando lenguaje SQL, para completar la base de datos del datawarehouse de la organización.

### Actividad 1: Crear una base de datos denominada DWPedidos

**Actividad 2: Crear las siguientes tablas en la base de datos DWPedidos****Actividad 3: Cargar de las tablas...**

Cargar las tablas de la base de datos DWPedidos con los datos de la base de datos de PEDIDOS, mediante procedimientos almacenados desarrollados en SQL

**Sección 0: Tareas previas a la creación del esquema y configuración del entorno****Borrar esquema (usuario) si existiera**

```
BEGIN
  EXECUTE IMMEDIATE 'DROP USER DWPEDIDOS CASCADE';
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -1918 THEN RAISE; END IF; -- ORA-01918: usuario no existe
```

```
END;  
/
```

### Crear el esquema y darle privilegios

```
BEGIN  
  EXECUTE IMMEDIATE q'[  
    CREATE USER DWPEDIDOS IDENTIFIED BY dabdTP4  
    DEFAULT TABLESPACE USERS  
    TEMPORARY TABLESPACE TEMP  
    QUOTA UNLIMITED ON USERS  
  ]';  
  EXECUTE IMMEDIATE 'GRANT CONNECT, RESOURCE TO DWPEDIDOS';  
END;  
/
```

### Definición de funciones personalizadas

```
-- Función para convertir (y almacenar) UUID en RAW(16)  
CREATE OR REPLACE FUNCTION uuid_to_raw(p_uuid IN VARCHAR2)  
  RETURN RAW DETERMINISTIC AS  
BEGIN  
  RETURN HEXTORAW(REPLACE(p_uuid,'-',''));  
END;  
/  
  
-- Función para recuperar y convertir de nuevo a UUID  
CREATE OR REPLACE FUNCTION raw_to_uuid(p_raw IN RAW)  
  RETURN VARCHAR2 DETERMINISTIC AS  
  v_hex VARCHAR2(32) := RAWTOHEX(p_raw);  
BEGIN  
  RETURN LOWER(  
    SUBSTR(v_hex,1,8)||'-'||  
    SUBSTR(v_hex,9,4)||'-'||  
    SUBSTR(v_hex,13,4)||'-'||  
    SUBSTR(v_hex,17,4)||'-'||  
    SUBSTR(v_hex,21,12)  
  );  
END;  
/
```

### Asignación de permisos

Para que el usuario DWPEDIDOS consulte datos de las tablas del esquema PEDIDOS, necesitamos:

- A. Permisos de SELECT en las tablas de PEDIDOS
- B. Permisos de ejecución de funciones.
- C. Permisos de creación de sinónimos para comodidad (así no escribimos PEDIDOS.<tabla> siempre)

```
-- A. Dar permisos de SELECT:
GRANT SELECT ON PEDIDOS.Clientes TO DWPEDIDOS;
GRANT SELECT ON PEDIDOS.Proveedores TO DWPEDIDOS;
GRANT SELECT ON PEDIDOS.Productos TO DWPEDIDOS;
GRANT SELECT ON PEDIDOS.Pedidos TO DWPEDIDOS;
GRANT SELECT ON PEDIDOS.DetallePedidos TO DWPEDIDOS;

-- B. Dar permisos de EXECUTE:
GRANT EXECUTE ON uuid_to_raw TO DWPEDIDOS;
GRANT EXECUTE ON raw_to_uuid TO DWPEDIDOS;

-- C. Dar permiso para crear sinónimos
GRANT CREATE SYNONYM TO DWPEDIDOS;
```

## Sección 1: Sentencias de creación de la base de datos DWPedidos

Sentencias de creación de las tablas del DW (modelo estrella, PK RAW(16)/UUID).

### Modelado dimensional y el modelo estrella

En los data warehouses, **no todas las tablas son iguales**:

- **Tablas de hechos:**
  - Registran los eventos principales del negocio, con métricas cuantificables.
  - Ejemplo: ventas, pedidos, transacciones.
  - Suele tener muchas filas, y claves foráneas a dimensiones.
- **Tablas de dimensiones:**
  - Describen los contextos de los hechos, o los “ejes” de análisis.
  - Ejemplo: productos, clientes, fechas, sucursales, empleados.
  - Suele tener pocos registros y campos descriptivos.

En el **modelo estrella**, la tabla central es la **tabla de hechos** y las otras son **dimensiones** que se conectan por claves foráneas.

### Dimensión de Fechas

```
-- Dimensión de Fechas
CREATE TABLE DIMFechas (
  id_Fec      RAW(16) PRIMARY KEY,
```

```
    fecha      DATE,  
    dia        NUMBER,  
    mes        NUMBER,  
    anio       NUMBER  
);
```

### Dimensión de Productos

```
-- Dimensión de Productos  
CREATE TABLE DIMProductos (  
    id_pro      RAW(16) PRIMARY KEY,  
    idproducto  RAW(16),  
    descripcion VARCHAR2(255),  
    nombreproveedor VARCHAR2(100)  
);
```

### Dimensión de Clientes

```
-- Dimensión de Clientes  
CREATE TABLE DIMClientes (  
    id_cli      RAW(16) PRIMARY KEY,  
    idcliente   RAW(16),  
    nombre      VARCHAR2(255)  
);
```

### Tabla de hechos de Pedidos

```
-- Tabla de hechos de Pedidos (FACTORIAL)  
CREATE TABLE FACTPedidos (  
    id_cli      RAW(16),  
    id_pro      RAW(16),  
    id_fec      RAW(16),  
    cantidad    NUMBER,  
    total       NUMBER(10,2),  
    FOREIGN KEY (id_cli) REFERENCES dim_clientes(id_cli),  
    FOREIGN KEY (id_pro) REFERENCES dim_productos(id_pro),  
    FOREIGN KEY (id_fec) REFERENCES dim_fechas(id_fec)  
);
```

## Sección 2: Detalle de proced. almacenados p/carga de datos en DWPedidos (Proceso ETL)

Conviene encapsular cada carga en un procedimiento almacenado.

Aquí la idea es que en vez de ejecutar directamente el `INSERT INTO ... SELECT ...`, se desarrolle un procedimiento almacenado (por ejemplo, `cargar_FACTPedidos`).

Así, cada vez que se quiera refrescar la tabla, solo ejecutaremos el procedimiento y listo. Esto se aplica tanto a la **tabla de hechos** como a cada **dimensión**.

#### Procedimiento para cargar DIMFechas

```
-- A. Procedimiento para cargar DIMFechas
CREATE OR REPLACE PROCEDURE cargar_DIMFechas IS
BEGIN
    -- Primero eliminamos datos existentes, para evitar duplicados en recargas
    DELETE FROM DIMFechas;

    -- Cargamos fechas únicas desde Pedidos
    INSERT INTO DIMFechas (id_fec, fecha, dia, mes, anio)
    SELECT
        SYS_GUID(),
        fecha,
        EXTRACT(DAY FROM fecha),
        EXTRACT(MONTH FROM fecha),
        EXTRACT(YEAR FROM fecha)
    FROM (SELECT DISTINCT fecha FROM PEDIDOS.Pedidos);

    COMMIT;
END cargar_DIMFechas;
/
```

#### Procedimiento para cargar DIMClientes

```
-- B. Procedimiento para cargar DIMClientes
CREATE OR REPLACE PROCEDURE cargar_DIMClientes IS
BEGIN
    DELETE FROM DIMClientes;

    INSERT INTO DIMClientes (id_cli, idcliente, nombre)
    SELECT
        SYS_GUID(),
        idcliente,
        apellido || ', ' || nombres
    FROM PEDIDOS.Clientes;

    COMMIT;
```



```
END cargar_DIMClientes;  
/
```

### Procedimiento para cargar DIMProductos

```
-- C. Procedimiento para cargar DIMProductos  
CREATE OR REPLACE PROCEDURE cargar_DIMProductos IS  
BEGIN  
    DELETE FROM DIMProductos;  
  
    INSERT INTO DIMProductos (id_pro, idproducto, descripcion, nombreproveedor)  
    SELECT  
        SYS_GUID(),  
        p.idproducto,  
        p.descripcion,  
        prov.nombreproveedor  
    FROM PEDIDOS.Productos p  
    JOIN PEDIDOS.Proveedores prov ON p.idproveedor = prov.idproveedor;  
  
    COMMIT;  
END cargar_DIMProductos;  
/
```

### Procedimiento para cargar FACTPedidos

```
-- D. Procedimiento para cargar FACTPedidos  
CREATE OR REPLACE PROCEDURE cargar_FACTPedidos IS  
BEGIN  
    DELETE FROM FACTPedidos;  
  
    INSERT INTO FACTPedidos (id_cli, id_pro, id_fec, cantidad, total)  
    SELECT  
        dc.id_cli,  
        dp.id_pro,  
        df.id_fec,  
        det.cantidad,  
        det.cantidad * det.preciounitario  
    FROM  
        PEDIDOS.DetallePedidos det  
    JOIN PEDIDOS.Pedidos ped ON det.numeropedido = ped.numeropedido  
    JOIN DIMFechas df ON ped.fecha = df.fecha  
    JOIN DIMClientes dc ON ped.idcliente = dc.idcliente
```

```
JOIN DIMProductos dp ON det.idproducto = dp.idproducto;

COMMIT;
END cargar_FACTPedidos;
/
```

#### Procedimiento `ejecutar_ETLMaestroDW`

Es un procedimiento que invoca a todos los anteriores, en orden.

```
-- Procedimiento para cargar todos los procedimientos ordenadamente
CREATE OR REPLACE PROCEDURE ejecutar_ETLMaestroDW IS
BEGIN
    cargar_DIMFechas;
    cargar_DIMClientes;
    cargar_DIMProductos;
    cargar_FACTPedidos;
END;
/
```

#### Ejecución del procedimiento ETL Maestro

```
-- Ejecutar el ETL Maestro
EXEC ejecutar_ETLMaestroDW;
```

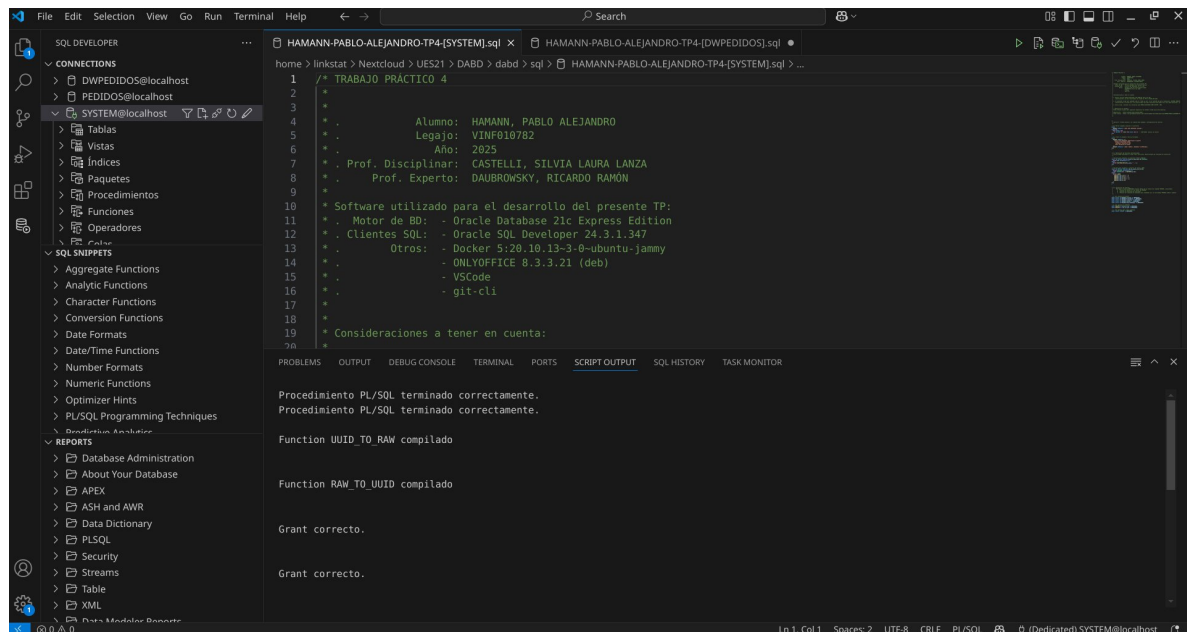
### Sección 3: Extra

Usando la instancia Oracle XE (instalada usando docker, como se explicó en el TP2), ahora configuré la extensión de Oracle SQL Developer en Visual Studio Code, lo que convirtió al editor de texto en una interfaz moderna, amigable y cómoda para trabajar sobre el motor de base de datos de Oracle.

Primero, me conecté como usuario SYSTEM (DBO), probé los comandos que guardé en el primer archivo de script (`HAMANN-PABLO-ALEJANDRO-TP4-[SYSTEM].sql`); allí verificamos que la ejecución de las sentencias fue exitosa.

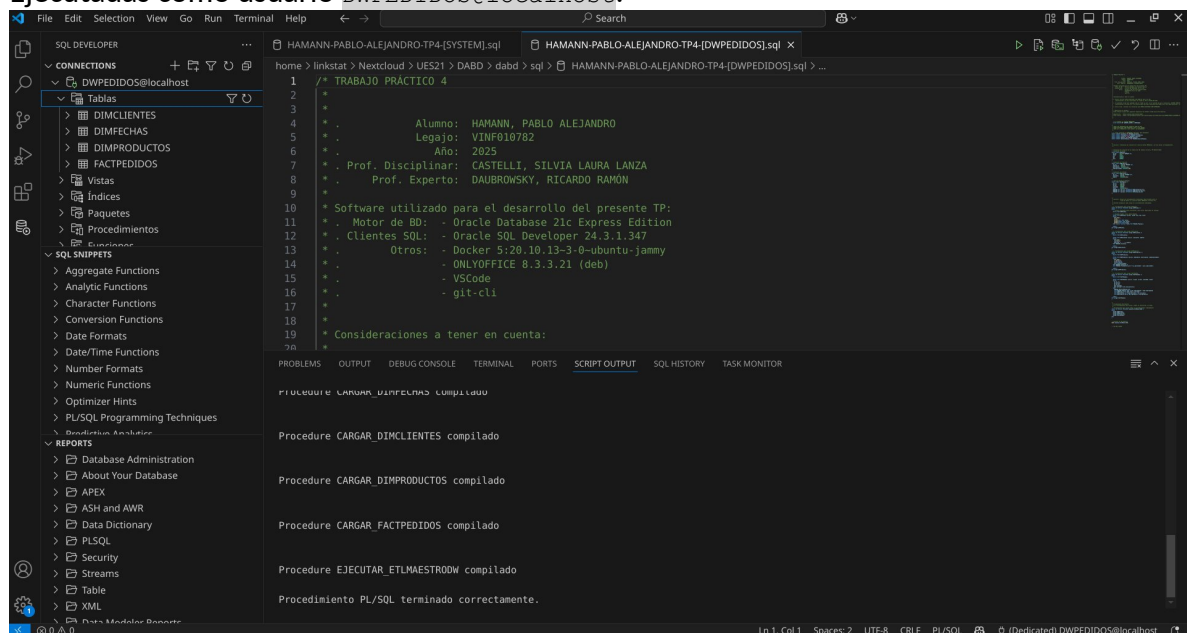
#### Captura de pantalla primera ejecución de sentencias SQL

Ejecutadas como usuario `SYSTEM@localhost:`

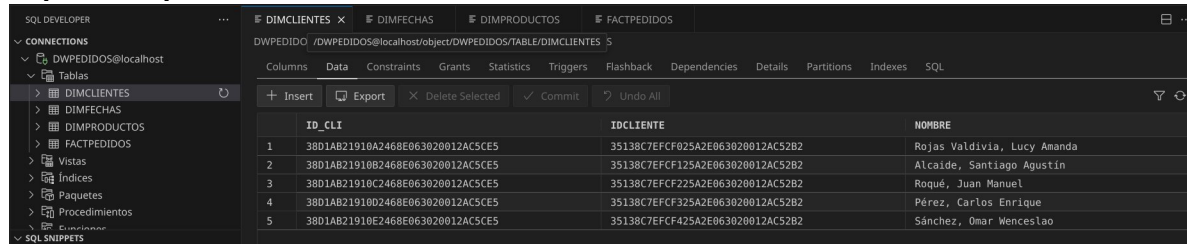


## Captura de pantalla segunda ejecución de sentencias SQL

Ejecutadas como usuario **DWPEDIDOS@localhost**:



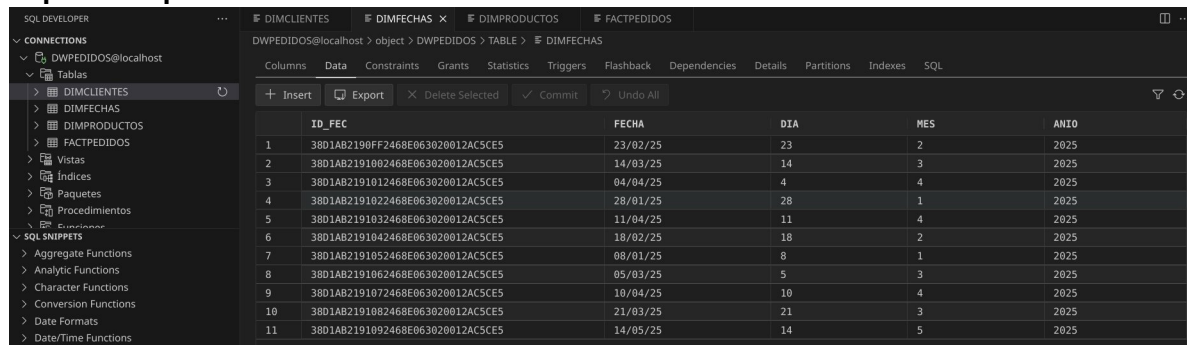
## Captura de pantalla de los datos en DWPEDIDOS.DIMClientes



The screenshot shows the SQL Developer interface with the 'DIMCLIENTES' table selected. The table contains 5 rows of data. The columns are ID\_CLI, IDCLIENTE, and NOMBRE.

ID_CLI	IDCLIENTE	NOMBRE
1	35138C7EFCF025A2E063020012AC52B2	Rojas Valdivia, Lucy Amanda
2	35138C7EFCF125A2E063020012AC52B2	Alcaide, Santiago Agustín
3	35138C7EFCF225A2E063020012AC52B2	Roqué, Juan Manuel
4	35138C7EFCF325A2E063020012AC52B2	Pérez, Carlos Enrique
5	35138C7EFCF425A2E063020012AC52B2	Sánchez, Omar Wenceslao

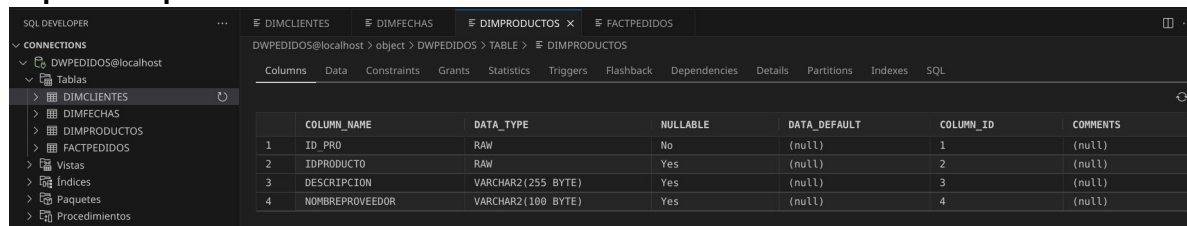
## Captura de pantalla de los datos en DWPEDIDOS.DIMFechas



The screenshot shows the SQL Developer interface with the 'DIMFECHAS' table selected. The table contains 11 rows of data. The columns are ID\_FEC, FECHA, DIA, MES, and ANIO.

ID_FEC	FECHA	DIA	MES	ANIO
1	23/02/25	23	2	2025
2	14/03/25	14	3	2025
3	04/04/25	4	4	2025
4	28/01/25	28	1	2025
5	11/04/25	11	4	2025
6	18/02/25	18	2	2025
7	08/01/25	8	1	2025
8	05/03/25	5	3	2025
9	10/04/25	10	4	2025
10	21/03/25	21	3	2025
11	14/05/25	14	5	2025

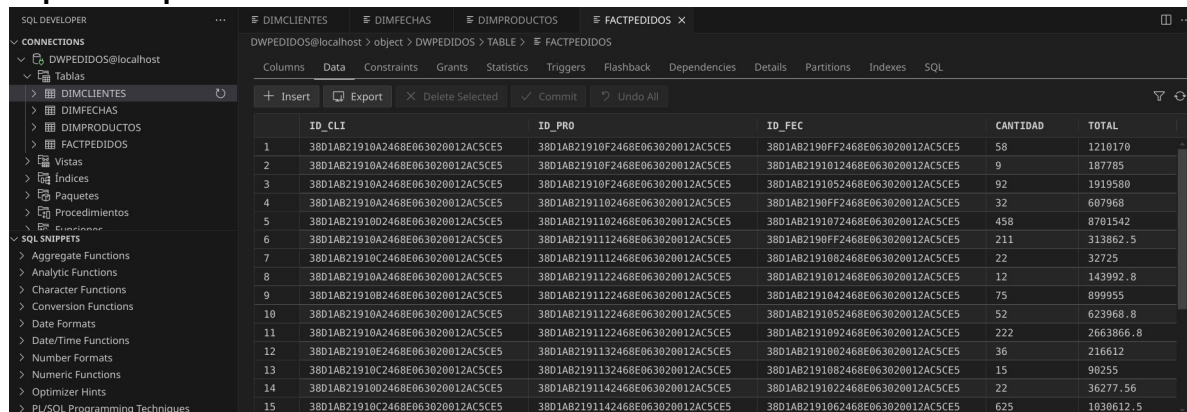
## Captura de pantalla de los datos en DWPEDIDOS.DIMProductos



The screenshot shows the SQL Developer interface with the 'DIMPRODUCTOS' table selected. The table contains 4 rows of data. The columns are COLUMN\_NAME, DATA\_TYPE, NULLABLE, DATA\_DEFAULT, COLUMN\_ID, and COMMENTS.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
ID_PRO	RAW	No	(null)	1	(null)
IDPRODUCTO	RAW	Yes	(null)	2	(null)
DESCRIPCION	VARCHAR2(255 BYTE)	Yes	(null)	3	(null)
NOMBREPROVEEDOR	VARCHAR2(100 BYTE)	Yes	(null)	4	(null)

## Captura de pantalla de los datos en DWPEDIDOS.FACTPedidos



The screenshot shows the SQL Developer interface with the 'FACTPEDIDOS' table selected. The table contains 15 rows of data. The columns are ID\_CLI, ID\_PRO, ID\_FEC, CANTIDAD, and TOTAL.

ID_CLI	ID_PRO	ID_FEC	CANTIDAD	TOTAL
38D1AB21910A2468E063020012AC5CE5	38D1AB21910F2468E063020012AC5CE5	38D1AB2190FF2468E063020012AC5CE5	58	1210170
38D1AB21910A2468E063020012AC5CE5	38D1AB21910F2468E063020012AC5CE5	38D1AB2191012468E063020012AC5CE5	9	187785
38D1AB21910A2468E063020012AC5CE5	38D1AB21910F2468E063020012AC5CE5	38D1AB2191052468E063020012AC5CE5	92	1919580
38D1AB21910A2468E063020012AC5CE5	38D1AB219102468E063020012AC5CE5	38D1AB2190FF2468E063020012AC5CE5	32	607968
38D1AB2191002468E063020012AC5CE5	38D1AB219102468E063020012AC5CE5	38D1AB2191072468E063020012AC5CE5	458	8701542
38D1AB21910A2468E063020012AC5CE5	38D1AB2191112468E063020012AC5CE5	38D1AB2190FF2468E063020012AC5CE5	211	313862.5
38D1AB21910C2468E063020012AC5CE5	38D1AB2191112468E063020012AC5CE5	38D1AB2191082468E063020012AC5CE5	22	32725
38D1AB21910A2468E063020012AC5CE5	38D1AB2191122468E063020012AC5CE5	38D1AB2191012468E063020012AC5CE5	12	143992.8
38D1AB21910B2468E063020012AC5CE5	38D1AB2191122468E063020012AC5CE5	38D1AB2191042468E063020012AC5CE5	75	899955
38D1AB21910A2468E063020012AC5CE5	38D1AB2191122468E063020012AC5CE5	38D1AB2191052468E063020012AC5CE5	52	623968.8
38D1AB21910A2468E063020012AC5CE5	38D1AB2191122468E063020012AC5CE5	38D1AB2191092468E063020012AC5CE5	222	2663866.8
38D1AB21910E2468E063020012AC5CE5	38D1AB2191132468E063020012AC5CE5	38D1AB2191002468E063020012AC5CE5	36	216612
38D1AB21910C2468E063020012AC5CE5	38D1AB2191132468E063020012AC5CE5	38D1AB2191082468E063020012AC5CE5	15	90255
38D1AB2191002468E063020012AC5CE5	38D1AB2191142468E063020012AC5CE5	38D1AB2191022468E063020012AC5CE5	22	36277.56
38D1AB21910C2468E063020012AC5CE5	38D1AB2191142468E063020012AC5CE5	38D1AB2191062468E063020012AC5CE5	625	1030612.5