SQL Moderno para Ingeniería Agroindustrial

Semana 04: Integración Bash + SQL + Python

Ingeniería de Datos

11 de enero de 2026

"Los datos son el fertilizante de la agricultura moderna."

¿Por qué SQL en 2026?

El problema con los CSV/Excel:

- Se cargan completos en RAM.
- Difícil cruzar información (VLOOKUP es lento).
- Sin control de tipos de datos.

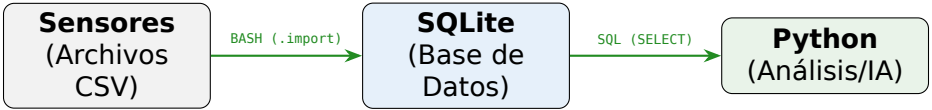
La solución SQL:

- **Eficiente:** Procesa en disco, no en RAM.
- **Relacional:** Une tablas complejas fácilmente.
- **Estándar:** Funciona igual en una Raspberry Pi que en la Nube.

En el Agro

Los sensores generan datos 24/7. SQL es el silo donde guardamos esa cosecha digital.

El Pipeline de Datos



- **Bash:** El cargador rápido (Ingesta).
- **SQL:** El almacén ordenado (Persistencia).
- **Python:** El cerebro analítico (Insights).

Capítulo I: Estructurando el Campo

Antes de sembrar datos, debemos preparar la tierra (Crear Tablas).

DDL - Data Definition Language

```sql
-- Crear la bitácora de sensores
CREATE TABLE sensores (
    id INTEGER PRIMARY KEY,
    zona TEXT NOT NULL,      -- Ej: 'Lote Norte'
    fecha DATETIME,
    temperatura REAL,        -- Grados Celsius
    humedad REAL             -- Porcentaje %
);
```

Capítulo I: Estructurando el Campo

Antes de sembrar datos, debemos preparar la tierra (Crear Tablas).

DDL - Data Definition Language

```sql
-- Crear la bitácora de sensores
CREATE TABLE sensores (
    id INTEGER PRIMARY KEY,
    zona TEXT NOT NULL,      -- Ej: 'Lote Norte'
    fecha DATETIME,
    temperatura REAL,        -- Grados Celsius
    humedad REAL             -- Porcentaje %
);
```

Consultando datos críticos:

```sql
SELECT zona, fecha
FROM sensores
WHERE humedad > 80 AND temperatura > 25;
-- Esto detecta condiciones ideales para hongos
```

Capítulo II: Ingesta Masiva con Bash

Situación: Tienes 50 archivos CSV de sensores. No vas a hacer `INSERT` manual. Usamos `sqlite3` en la terminal:

Comando de Terminal (ETL Express)

```
# 1. Definir modo CSV
# 2. Importar archivo -> tabla
sqlite3 finca.db << EOF
.mode csv
.import datos_crudos.csv sensores
EOF
```

**Esto es 100 veces más rápido que un bucle for en Python.*

Capítulo III: Conexión con Pandas

Una vez los datos están en SQL, Python solo trae lo necesario.

```python
import sqlite3
import pandas as pd

conn = sqlite3.connect('finca.db')

# BUENA PRÁCTICA: Filtrar en SQL, no en Pandas
query = """
    SELECT zona, AVG(temperatura) as media
    FROM sensores
    WHERE fecha >= '2024-01-01'
    GROUP BY zona
"""

df = pd.read_sql(query, conn)
print(df)
```

Ética: Soberanía de Datos

¿Dónde viven los datos?

Si usamos Google Sheets o AWS, dependemos de internet y corporaciones extranjeras.

Ventajas de SQLite en el Agro:

1. **Local:** Funciona “ Offline” en la finca.
2. **Propiedad:** El archivo ‘.db‘ es fácil de copiar y respaldar.
3. **Auditabilidad:** El agricultor puede ver qué se registra.

Taller: Monitor de Cultivos

Objetivo: Construir el Pipeline completo.

1. **Generar Caos:** Ejecuta `generar_sensores.py` para crear datos sucios.
2. **Ingesta (Bash):** Crea la DB e importa el CSV automáticamente.
3. **Lógica (SQL):** Encuentra los "Días de Estrés Hídrico"(Humedad $< 30\,\%$).
4. **Reporte (Python):** Grafica los lotes más afectados.

¡Manos a la obra!