

# Diseño y construcción de una base de datos pluviométricos diarios, apta para una rápida captura del campo de precipitación de una tormenta

Martín Burgues, Daniel Barrera y  
Margarita Portapila

# Introducción

- Utilidad y necesidad de una base de datos pluviométricos georreferenciados
  - aplicaciones operativas:
    - pronóstico hidrológico,
    - seguimiento de humedad del suelo y el nivel de la superficie freática mediante técnicas de balance hídrico,
    - validación de los modelos que estiman precipitación
  - diferentes fuentes
    - Secretaría de Agricultura Ganadería, Pesca y Alimentación SAGPyA
    - INTA
    - agencias provinciales de recursos hídricos
    - corporaciones de acopiadores de cereales

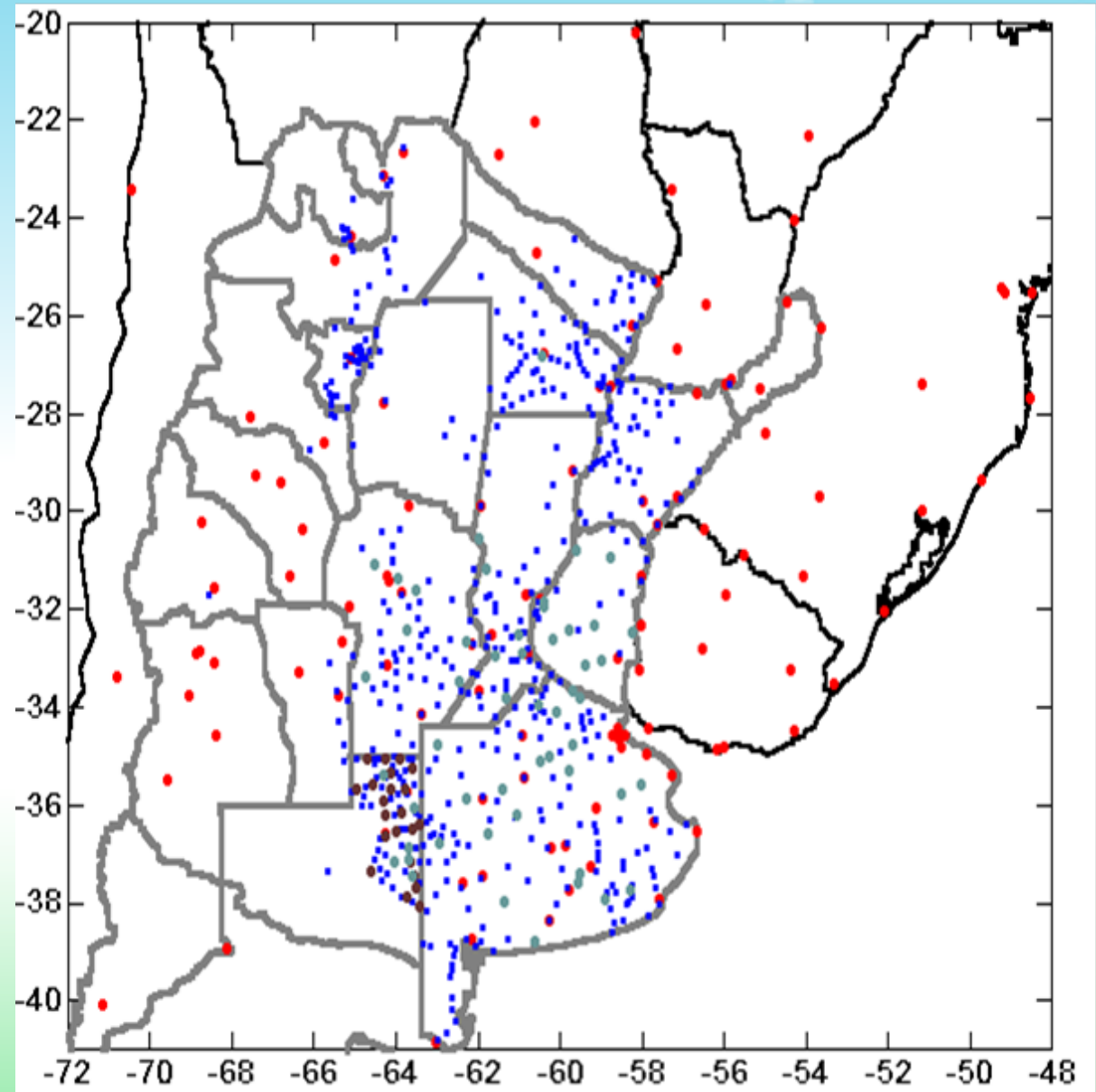
# Introducción

- Valores puntuales y areales de precipitación. Relación con las características de la red pluviométrica
  - ubicación de los centros de máxima precipitación
  - técnicas basadas en imágenes satelitales
  - recopilación de los datos a tiempo
  - interpretación de los ingresos

# Redes pluviométricas disponibles

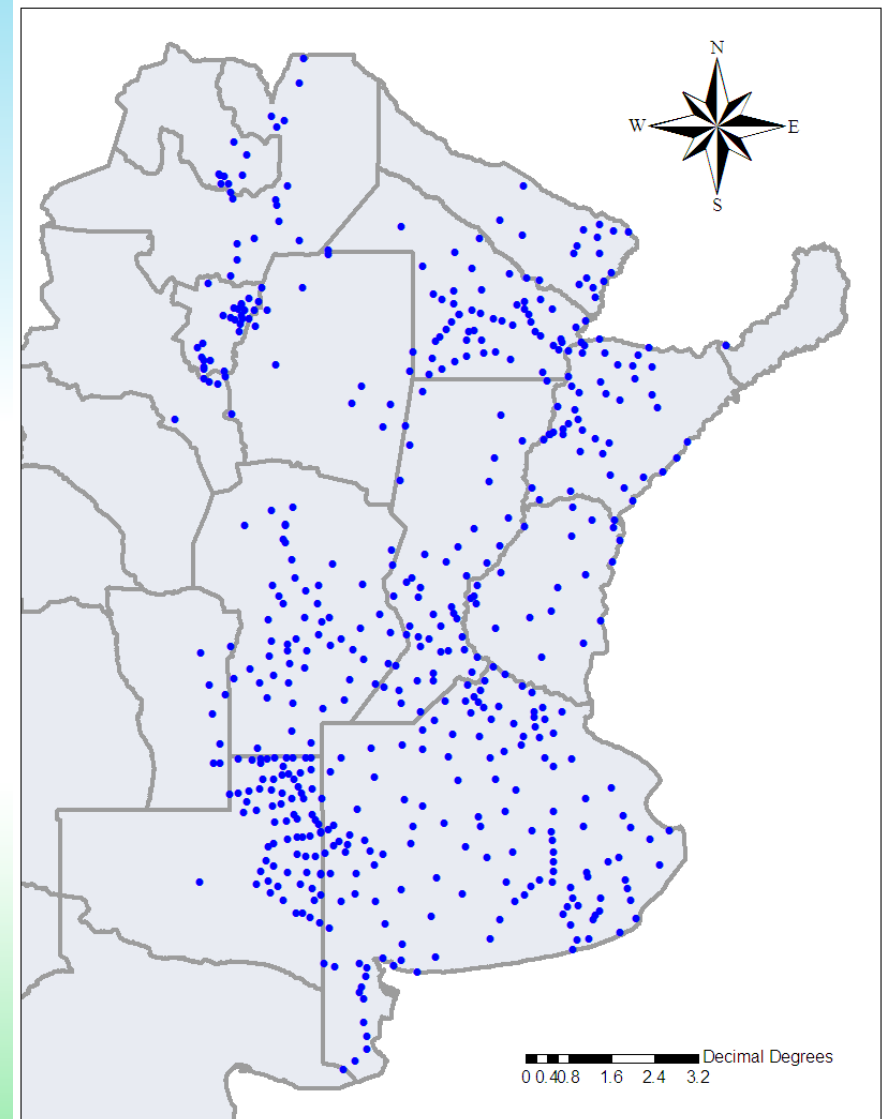
Se muestran con distintos colores las estaciones con información proveniente de distintas fuentes:

- rojo (SMN)
- azul (SAGPyA)
- marrón (INTA)
- verde (Federación de Acopiadores de Cereales)



# Área de estudio

Área de mayor producción agrícola de la república Argentina. 500 pluviómetros activos que recopila la SAGPyA diariamente.



# Software utilizado

- **SQL** Structured Query Language

Lenguaje estandarizado de consultas a base de datos de forma programatica,

- consta de un lenguaje de definición DDL (CREATE, ALTER, DROP, TRUNCATE, ...)
- lenguaje de manipulación de datos DML (SELECT, INSERT, UPDATE, DELETE)

- **SQLite**

Motor de base de datos, de código abierto y libre simple y muy utilizado en dispositivos de limitada capacidad de procesamiento.

# Software utilizado

- **Python 2.7**

Lenguaje de programación de alto nivel, moderno creado a principio de los 90, es un lenguaje de scripts originalmente inspirado en Perl.

- Algunas características:

- es interpretado,
- con tipado dinámico,
- fuertemente tipado,
- multiplataforma,
- orientado a objetos,
- y con algunas características de programación funcional

# Software utilizado

- **ORM Object Relational Mapping**  
Objeto de programación que brinda servicios de mapeo entre clases de un programa orientado a objetos y los objetos de la base de datos
- **SQLAlchemy**  
ORM que interactúa con python.
- **pyExcelerator**  
Objeto de software encargado de encapsular el funcionamiento de las hojas de calculo excel, para interactuar con un programa.

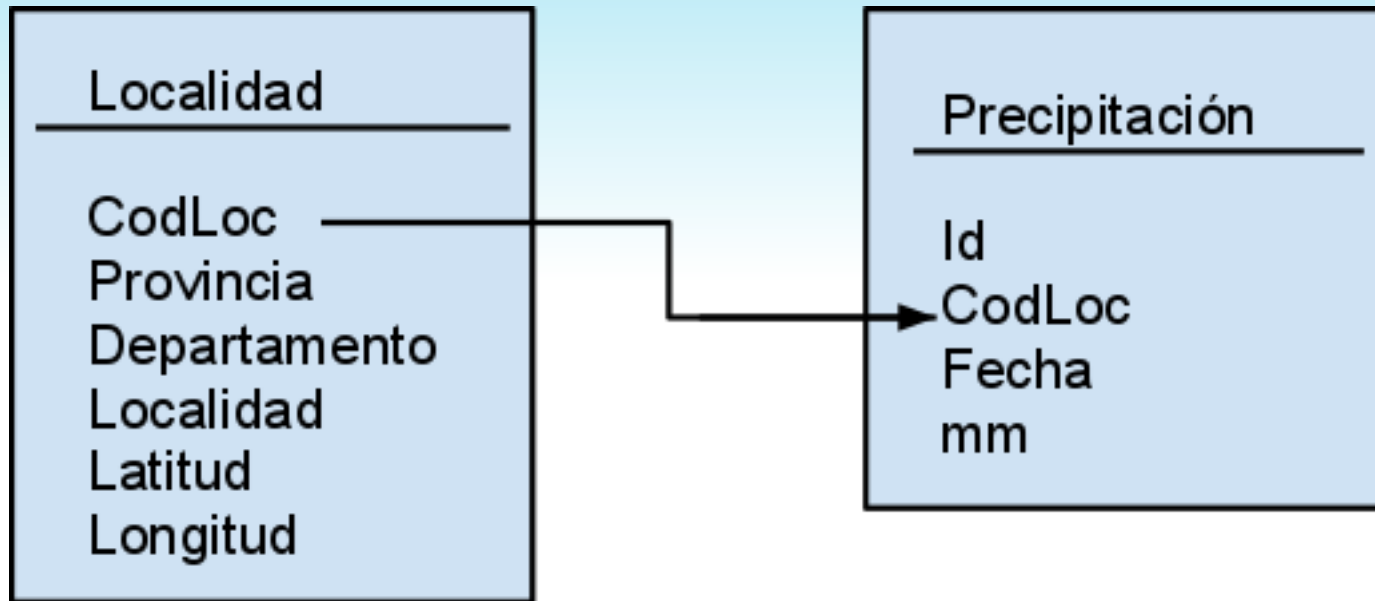


# Diseño de la base de datos

- diseño relacional
- basado en entidades físicas (Localidad, Precipitación)
- normalización 3FN
  - 1FN
    - atributos indivisibles
    - clave primaria sin nulos
    - dependencia funcional de los atributos a su clave principal
    - independencia de los registros
    - todos los registros tienen la misma cantidad de atributos
  - 2FN
    - dependencia funcional completa
  - 3FN
    - no existen dependencias transitivas

# Diseño de la base de datos

## Diagrama entidad-relación



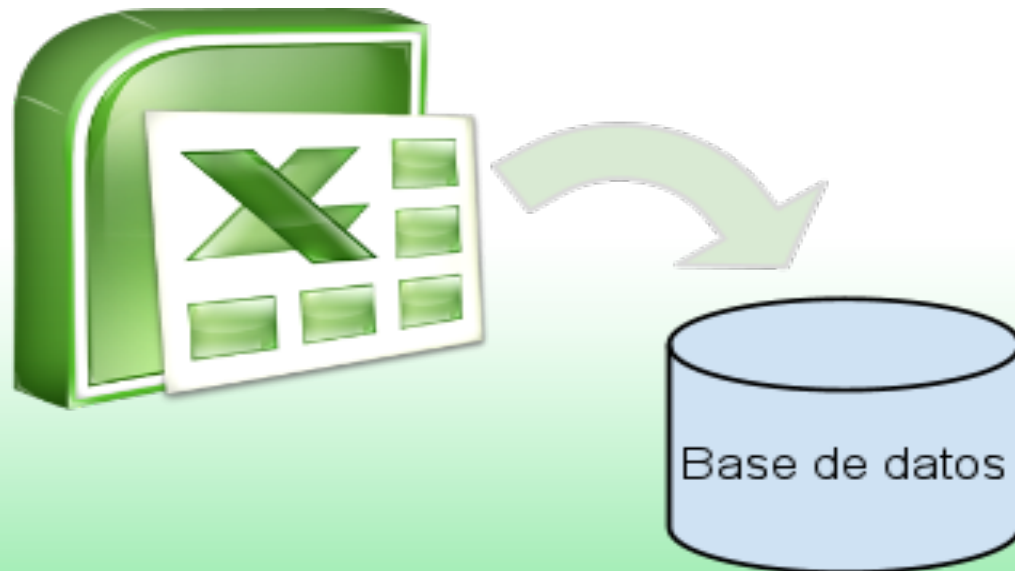
de este esquema se desprende un diagrama de clases similar que da lugar al ORM para administrar la base de datos

# Estructura de la aplicación

- **ReadLoc**

Carga los datos de la tabla de localidades georreferenciadas para llenar la entidad Localidad en la BD incluyendo los datos de latitud y longitud.

Con este paso, cualquier dato relacionado con la Localidad es factible de ser automáticamente georreferenciado.



# Estructura de la aplicación

- **ReadSAGPyA:**

Este módulo se encarga de leer masivamente los datos de las planillas semanales de la SAGPyA previo análisis de la información y normalización de los datos.

## **Normalización:**

- capitalización
- abreviaturas comunes
- acentos y caracteres raros
- diccionario de sinónimos

# Estructura de la aplicación

- **Consultar**

Extrae los datos de la base de acuerdo a los criterios pedidos por medio del lenguaje SQL.

```
SELECT localidad,  
    latitud,  
    longitud,  
    p.fecha,  
    p.mm  
FROM localidad AS l INNER JOIN precipitacion AS p  
    ON l.CodLoc = p.CodLoc  
WHERE l.localidad = 'Soldini'  
    AND p.fecha BETWEEN '2007-10-01' AND '2007-10-31'  
    AND p.mm > 0  
ORDER BY p.fecha
```

# Estructura de la aplicación

<b>Loc.</b>	<b>Lat.</b>	<b>Long.</b>	<b>Fecha</b>	<b>mm</b>
Soldini,	-60.75,	-33.02,	2007-10-02,	17.0
Soldini,	-60.75,	-33.02,	2007-10-09,	16.0
Soldini,	-60.75,	-33.02,	2007-10-11,	29.0
Soldini,	-60.75,	-33.02,	2007-10-20,	6.0
Soldini,	-60.75,	-33.02,	2007-10-26,	23.0
Soldini,	-60.75,	-33.02,	2007-10-31,	55.0

# Otras consultas

- Promedio de precipitaciones
- Total de precipitaciones mensuales / anuales
- Seleccionar localidades en una zona geográfica filtrada por su lat/long
- Comparar muestras de localidades cercanas
- Comparación de períodos de tiempo
- etc...

# Conclusiones

- Este trabajo permite concentrar esfuerzos de las recopilaciones de datos realizadas por la SAGPyA en una completa base de datos pluviométricos que pueden ser consultados empleando múltiples criterios y obteniendo los resultados de la consulta casi instantáneamente.
- Se utilizó PYTHON como lenguaje de alto nivel que cuenta con cantidad de paquetes de objetos de software para manejo de base de datos, ORM, extracción de datos de planillas Excel, manejo de cadenas de texto necesarias para la normalización de los datos de entrada.



# Conclusiones

- Como la base de datos cuenta con información pluviométrica y las coordenadas geográficas de las estaciones se pueden realizar consultas que devuelvan tablas con registros georreferenciados que pueden ser utilizadas directamente en sistemas de información geográfica.
- La utilidad de esta base de datos aumentará a medida que se vayan incorporando más estaciones pluviométricas, provenientes de SAGPyA como de otras fuentes.

**Preguntas?**

**Muchas gracias!**

**Contacto:**  
*Martín Burgues*  
*[lccmartin@gmail.com](mailto:lccmartin@gmail.com)*



~ fin ~