UNIVERSIDAD TECNOLÓGICA DE CHIHUAHUA

Tecnologías de la Información: Desarrollo y Gestión de Software

I.4. Reporte de investigación de los lenguajes y bibliotecas para análisis y procesamiento de datos

IDGS81N - Kevin Iván Aguirre Silva
Extracción de Conocimiento en Bases de Datos - Ing. Luis Enrique Mascote Cano

Chihuahua, Chih., 23 de septiembre de 2025

ÍNDICE

INTRODUCCIÓN

En el campo del análisis de datos, el conocimiento de los lenguajes de programación y sus bibliotecas resulta fundamental para aprovechar al máximo el potencial de la información. Cada lenguaje aporta características únicas: Python, con su tipado dinámico y versatilidad, ofrece herramientas como Pandas para manipulación de datos y TensorFlow para machine learning; R se distingue por su potencia estadística y bibliotecas como dplyr y tidyr, que facilitan la transformación y limpieza de datos; Scala, gracias a su integración con Apache Spark, es clave en entornos de big data y procesamiento distribuido, complementado por librerías numéricas como Breeze; SQL se mantiene esencial para la gestión de bases de datos relacionales y el acceso eficiente a grandes volúmenes de información, apoyado por ORMs como SQLAlchemy o Sequelize; y Julia, con su rendimiento cercano al de lenguajes de bajo nivel, se ha consolidado en computación científica mediante bibliotecas como DataFrames.jl y Flux.jl.

Conocer estos lenguajes y sus ecosistemas no solo facilita la manipulación, exploración y modelado de datos, sino que también permite seleccionar la herramienta adecuada según la naturaleza del problema, ya sea estadístico, científico, empresarial o de inteligencia artificial. Así, el dominio de estas tecnologías se convierte en un pilar estratégico para generar soluciones eficientes, reproducibles y escalables en la ciencia de datos moderna.

OBJETIVOS DEL REPORTE

- Analizar la importancia de conocer los lenguajes de programación más utilizados en el análisis de datos, destacando sus características principales.
- Identificar y describir las bibliotecas y frameworks clave de cada lenguaje (Python, R, Scala, SQL y Julia) que facilitan la manipulación, transformación, visualización y modelado de datos.
- Comparar los ámbitos de aplicación de los distintos lenguajes, mostrando cómo cada uno se adapta a contextos específicos como estadística, big data, machine learning, computación científica o gestión de bases de datos.
- Mostrar ejemplos prácticos de uso básico (“Hola datos”) que permitan ilustrar la facilidad de integración de cada lenguaje con datos reales.

- Resaltar la relevancia de seleccionar adecuadamente las herramientas de programación y análisis para optimizar procesos de exploración, procesamiento y toma de decisiones basadas en datos.

PYTHON

Descripción general

Python es un lenguaje de programación interpretado y de tipado dinámico, lo que significa que no necesita compilación previa y que las variables no requieren declaración explícita de tipo, ya que el tipo se evalúa en tiempo de ejecución. Además, es multiparadigma, soportando programación estructurada, orientada a objetos y funcional. Su ámbito principal de uso incluye ciencia de datos, desarrollo de aplicaciones y desarrollo back-end.

Bibliotecas y frameworks clave

Pandas

Pandas es una biblioteca de código abierto para Python especializada en la manipulación y análisis de datos. Su funcionalidad principal consiste en ofrecer estructuras de datos (como DataFrames y Series) y herramientas que permiten cargar, transformar, limpiar, analizar y visualizar datos de forma eficiente y sencilla.

El caso de uso típico de Pandas es en el análisis y procesamiento de datos tabulares, como datos provenientes de archivos CSV, bases de datos SQL o Excel, siendo clave en la ciencia de datos para preparar y explorar los datos antes de aplicar técnicas de machine learning o estadística.

TensorFlow

TensorFlow es una plataforma de código abierto para el aprendizaje automático (machine learning) y el deep learning, que facilita la creación, entrenamiento e implementación de modelos de inteligencia artificial. Su funcionalidad principal es permitir a los desarrolladores construir modelos complejos de redes neuronales y otros algoritmos de aprendizaje automático de manera flexible y escalable, con soporte para ejecución en diferentes dispositivos, desde servidores hasta dispositivos móviles.

El caso de uso típico de TensorFlow incluye reconocimiento de imágenes, análisis de sentimientos, diagnósticos médicos, y más, donde se requieren modelos capaces de aprender patrones complejos a partir de grandes volúmenes de datos.

Mini-ejemplo “Hola datos”

```python
import pandas as pd

# Cargar el archivo CSV ficticio 'datos.csv' en un DataFrame
df = pd.read_csv('datos.csv')

# Mostrar las primeras 5 filas
print(df.head())
```

Este código usa la biblioteca pandas para leer el archivo 'datos.csv' y luego imprime las primeras filas del DataFrame, lo que permite una revisión rápida de los datos cargados.

R

Descripción general

El lenguaje R es un lenguaje de programación interpretado y de tipado dinámico, orientado principalmente a objetos. Esto permite manejar datos y resultados como objetos que pueden modificarse mediante métodos específicos. R es ampliamente usado en ciencia de datos debido a su capacidad para realizar cálculos estadísticos complejos, análisis de datos y visualización gráfica de alta calidad.

Su ámbito de uso principal se centra en la ciencia de datos, análisis estadístico, minería de datos y cálculo numérico. Aunque puede integrarse con otros lenguajes y bases de datos, no es comúnmente utilizado para desarrollo de aplicaciones o back-end de software tradicional.

R destaca por su extenso ecosistema de paquetes especializados para estadística y gráficos, y su fortaleza en generar informes reproducibles y visualizaciones interactivas usando herramientas como Shiny. También es multiplataforma y compatible con Windows, macOS y Linux.

Bibliotecas y frameworks clave

Dplyr

Dplyr es un paquete de R diseñado para facilitar la manipulación y transformación de datos. Su funcionalidad principal es proporcionar una gramática consistente y sencilla para realizar operaciones comunes sobre data frames, como seleccionar columnas, filtrar filas, ordenar datos, agregar columnas nuevas o resumir información estadística.

El caso de uso de dplyr es la exploración y manipulación limpia y eficiente de conjuntos de datos, siendo muy popular en la ciencia de datos para preparar datos antes de análisis o visualizaciones. dplyr funciona muy bien con otros paquetes del ecosistema tidyverse.

Tidyr

tidyr es un paquete de R cuyo propósito principal es simplificar la transformación de datos desordenados en datos ordenados (tidy data), esto es datos en los que cada variable es una

columna, cada observación es una fila y cada celda contiene un único valor. Esta estructura facilita el análisis y visualización posterior.

El caso de uso de tidyr es la limpieza y reestructuración de conjuntos de datos que no están en un formato estándar, como transformar datos anchos en datos largos o viceversa, para hacerlos más manejables y compatibles con otras herramientas de análisis en R.

Mini-ejemplo “Hola datos”

Este código usa la función read_csv() de la librería readr para importar datos desde un archivo CSV a un data frame y luego muestra las primeras filas con head() y un resumen estadístico con summary(), facilitando la exploración inicial de datos.

```r
# Cargar librería readr para importar CSV
library(readr)

# Leer archivo CSV ficticio 'datos.csv'
df <- read_csv("datos.csv")

# Mostrar las primeras 6 filas
print(head(df))

# Mostrar un resumen del data frame
summary(df)
```

SCALA

Descripción general

Scala es un lenguaje de programación compilado que se ejecuta sobre la Máquina Virtual de Java (JVM), y es estáticamente tipado, lo que significa que la verificación de tipos se realiza en tiempo de compilación. Es un lenguaje multiparadigma que combina programación orientada a objetos y funcional, permitiendo escribir código conciso, expresivo y robusto.

El ámbito principal de uso de Scala incluye el desarrollo de aplicaciones y back-end, especialmente en sistemas distribuidos y procesamiento de datos a gran escala. Scala es muy popular en ingeniería de datos y big data, puesto que se integra estrechamente con tecnologías como Apache Spark, facilitando la creación de pipelines de datos y el análisis de grandes volúmenes de información.

Scala destaca por su interoperabilidad completa con Java, su sistema avanzado de tipos y su sintaxis limpia y escalable, lo que la hace apta tanto para proyectos pequeños como para aplicaciones empresariales complejas.

Bibliotecas y frameworks clave

Apache Spark

Apache Spark es un motor de procesamiento de datos de código abierto desarrollado principalmente en Scala. Su funcionalidad principal es permitir el procesamiento rápido y distribuido de grandes volúmenes de datos en memoria y en disco, con soporte para tareas en lote, procesamiento de flujos en tiempo real y análisis avanzados, como machine learning.

El caso de uso típico de Apache Spark en Scala incluye análisis y procesamiento de big data, construcción de pipelines de datos, aprendizaje automático, y análisis tanto en tiempo real como por lotes. Spark destaca por su velocidad, escalabilidad y flexibilidad, procesando datos en clústeres distribuidos y soportando diversas fuentes de datos, como HDFS, bases NoSQL y almacenamiento en la nube.

Breeze

Breeze es una biblioteca de Scala especializada en el procesamiento numérico y álgebra lineal. Su funcionalidad principal es proporcionar estructuras de datos y operaciones eficientes para cálculos matemáticos, similares a las bibliotecas Matlab o NumPy en Python. Breeze permite realizar operaciones con vectores, matrices y funciones matemáticas, lo que es crucial para aplicaciones de ciencia de datos, aprendizaje automático y análisis numérico en Scala.

El caso de uso típico de Breeze es en cálculos numéricos complejos, modelos estadísticos y algoritmos de machine learning donde se requiere un alto rendimiento computacional y facilidad para manipular datos numéricos.

Mini-ejemplo “Hola datos”

```scala
import java.io.File
import com.github.tototoshi.csv._

object HolaDatos {
  def main(args: Array[String]): Unit = {
    val reader = CSVReader.open(new File("datos.csv"))
    val allRows = reader.allWithHeaders()  // Lee todas las filas con encabezados
    allRows.take(5).foreach(println)        // Muestra las primeras 5 filas
    reader.close()
  }
}
```

Este código abre un archivo CSV llamado "datos.csv", lee todas las filas con sus encabezados y muestra las primeras cinco filas por consola. Es una forma sencilla y eficiente de empezar a trabajar con datos CSV en Scala usando scala-csv.

SQL

Descripción general

SQL (Structured Query Language) es un lenguaje de programación declarativo e interpretado, utilizado para gestionar y manipular bases de datos relacionales. No es compilado, sino que las consultas se procesan y ejecutan directamente por el motor del sistema de gestión de bases de datos (SGBD). SQL es de tipado dinámico en tanto que no requiere una declaración estática previa de los tipos de datos al escribir consultas.

Su ámbito principal de uso está en el manejo de bases de datos para recuperación, inserción, actualización y eliminación de datos, siendo fundamental en ciencia de datos para acceder y preparar grandes conjuntos de datos relacionales. También es crucial en desarrollo de aplicaciones y back-end, donde sirve para manejar la persistencia y consulta eficiente de datos.

SQL destaca por su capacidad para trabajar con datos estructurados en tablas relacionadas mediante claves primarias y foráneas, y por ofrecer cumplimiento de propiedades ACID que garantizan la integridad y consistencia en transacciones. Su uso es omnipresente en sistemas empresariales, aplicaciones web, análisis de datos, inteligencia de negocios y más.

Bibliotecas y frameworks clave

SQLAlchemy

SQLAlchemy es una biblioteca en Python que facilita la interacción con bases de datos SQL mediante un mapeo objeto-relacional (ORM) poderoso y flexible. Su funcionalidad principal es convertir automáticamente las llamadas a clases y funciones de Python en sentencias SQL, permitiendo manipular bases de datos de manera natural y segura sin escribir directamente SQL. SQLAlchemy también ofrece una capa core para expresiones SQL y manejo de conexiones, brindando control total a desarrolladores.

Un caso de uso concreto es la gestión de bases de datos en aplicaciones web, donde ayuda a definir modelos de datos como clases Python y realizar consultas, inserciones o actualizaciones sin escribir código SQL explícito, por ejemplo, con Flask.

Sequelize

Sequelize es un ORM (Object-Relational Mapping) para Node.js que facilita la interacción con bases de datos SQL desde aplicaciones JavaScript. Su funcionalidad principal es mapear tablas de bases de datos en modelos de JavaScript, permitiendo crear, leer, actualizar y eliminar registros sin necesidad de escribir consultas SQL directamente. Sequelize simplifica la gestión de datos en aplicaciones web, especialmente en back-end con frameworks como Express.

Un caso de uso concreto es el desarrollo de APIs donde Sequelize gestiona la persistencia de datos en bases como MySQL, PostgreSQL o SQLite, bridando una interfaz consistente y orientada a objetos.

Mini-ejemplo “Hola datos”

En SQL, para cargar un archivo CSV y mostrar las primeras filas, usualmente se sigue un proceso de dos pasos: primero importar los datos CSV a una tabla y luego consultar esa tabla. Un ejemplo típico sería usar el comando BULK INSERT para cargar el CSV y luego un SELECT para mostrar las filas:

```
-- Cargar datos desde un archivo CSV a una tabla llamada 'mi_tabla'
BULK INSERT mi_tabla
FROM 'C:\\ruta\\a\\archivo.csv'
WITH (
  FORMAT = 'CSV',
  FIRSTROW = 2,  -- Ignora la fila del encabezado
  FIELDTERMINATOR = ',',
  ROWTERMINATOR = '\n'
);

-- Mostrar las primeras 5 filas de la tabla
SELECT TOP 5 * FROM mi_tabla;
```

Este fragmento carga un archivo CSV en una tabla SQL Server y luego muestra las primeras filas, permitiendo revisar rápidamente los datos importados.

JULIA

Descripción general

Julia es un lenguaje de programación compilado justo a tiempo (JIT) que utiliza LLVM para transformar el código en código máquina eficiente en tiempo de ejecución, proporcionando un rendimiento cercano al de lenguajes de bajo nivel como C o Fortran. Tiene un tipado dinámico con soporte para tipos opcionales, combinando así flexibilidad y robustez en la escritura de código. Julia es multiparadigma, permitiendo programación funcional, orientada a objetos y metaprogramación.

El ámbito principal de uso de Julia se encuentra en la ciencia de datos, el cálculo numérico y la computación científica, donde destaca por su capacidad para manejar grandes volúmenes de datos y realizar cálculos matemáticos complejos con alta velocidad. Aunque puede usarse para desarrollo de aplicaciones o back-end, su fuerte está en aplicaciones científicas, análisis avanzado, simulaciones y machine learning.

Además, Julia cuenta con un ecosistema creciente de bibliotecas para distintas áreas, incluyendo manipulación de datos (DataFrames.jl), aprendizaje automático (Flux.jl) y resolución de ecuaciones diferenciales (DifferentialEquations.jl), lo que la hace muy versátil para proyectos científicos e industriales.

Bibliotecas y frameworks clave

DataFrames.jl

DataFrames.jl es una biblioteca de Julia diseñada para la manipulación y análisis de datos tabulares. Su funcionalidad principal es proporcionar estructuras de datos tipo DataFrame, similares a las de pandas en Python o data.frames en R, que facilitan la carga, exploración, transformación y resumen de conjuntos de datos organizados en filas y columnas.

El caso de uso típico de DataFrames.jl es en ciencia de datos para trabajar con datos estructurados, desde cargar archivos CSV hasta realizar operaciones como filtrado, agrupación y agregación. Es una herramienta indispensable para análisis estadísticos y preparación de datos en Julia.

Flux.jl

Flux.jl es una biblioteca de aprendizaje automático (machine learning) para el lenguaje Julia. Su funcionalidad principal es proporcionar herramientas para construir, entrenar y desplegar modelos de deep learning y machine learning de manera flexible y eficiente, utilizando toda la potencia y velocidad del lenguaje Julia. Flux.jl permite definir modelos como funciones matemáticas, facilitando la personalización y extensión con código Julia puro.

Un caso de uso concreto es la creación de modelos neuronales para tareas como regresión, clasificación o reconocimiento de patrones en datos, aprovechando la facilidad para construir arquitecturas personalizadas y optimizar automáticamente los modelos.

Mini-ejemplo “Hola datos”

```julia
using CSV
using DataFrames

# Cargar el archivo CSV ficticio "datos.csv" a un DataFrame
df = CSV.read("datos.csv", DataFrame)

# Mostrar las primeras 5 filas del DataFrame
println(first(df, 5))
```

Este código usa los paquetes estándar CSV.jl para lectura y DataFrames.jl para manipulación tabular, cargando un archivo CSV en un objeto DataFrame y mostrando las primeras filas para una vista rápida de los datos.

CONCLUSIÓN

El análisis de los lenguajes Python, R, Scala, SQL y Julia permite observar que, si bien todos cumplen un papel importante en la ciencia de datos, cada uno se ha consolidado en nichos específicos gracias a sus características y ecosistema de bibliotecas. En cuanto a la facilidad de uso, Python y R destacan por sus sintaxis intuitivas y gran cantidad de recursos de aprendizaje, lo que los convierte en la primera elección para estudiantes y profesionales que buscan realizar análisis exploratorios o prototipado rápido. Julia, aunque más reciente, también apuesta por una sintaxis clara, combinando la flexibilidad de lenguajes de alto nivel con un rendimiento cercano al de C. Por otro lado, Scala presenta una curva de aprendizaje más pronunciada debido a su tipado estático y su integración con la JVM, aunque esta complejidad se compensa con la robustez y escalabilidad que ofrece en entornos empresariales.

Respecto al rendimiento, Julia y Scala suelen ser superiores cuando se trata de cómputo intensivo o procesamiento de grandes volúmenes de datos distribuidos, mientras que Python y R, al depender en gran medida de bibliotecas optimizadas en C o Fortran, mantienen un desempeño aceptable, aunque no siempre comparable. SQL, por su parte, no compite en términos de cálculo, pero es insustituible en la gestión y consulta eficiente de bases de datos estructuradas. En cuanto al ecosistema de bibliotecas, Python es el más amplio y versátil, con opciones para casi cualquier tarea (Pandas, TensorFlow, Scikit-learn). R sobresale en estadística y visualización, Scala en big data gracias a Spark, y Julia en cálculo científico avanzado.

En relación con los casos de aplicación, se puede afirmar que Python y R son ideales para proyectos de análisis ligero, exploración de datos y prototipos, debido a su facilidad de uso y rapidez para obtener resultados. En contraste, Scala y Julia se recomiendan para proyectos de producción a gran escala o entornos donde el rendimiento y la eficiencia son críticos, como sistemas distribuidos, big data o simulaciones científicas complejas. SQL, en todos los casos, permanece como un complemento indispensable para la gestión y preparación de datos antes de su análisis.

La elección del lenguaje no debería centrarse en cuál es “mejor”, sino en cuál es el más adecuado para el problema en cuestión: Python y R para exploración ágil y versatilidad, y Scala o Julia cuando el reto exige escalabilidad, alto rendimiento y procesamiento intensivo.

REFERENCIAS BIBLIOGRÁFICAS

1. *Welcome to Python.org*. (18 septiembre de 2025). Python.org. https://www.python.org/
2. *¿Qué es Python Pandas?* (6 junio de 2025). IONOS Digital Guide. https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/python-pandas/
3. Alberca, A. S. (14 junio de 2022). *La librería Pandas | Aprende con Alf*. Aprende Con Alf. https://aprendeconalf.es/docencia/python/manual/pandas/
4. *Por qué TensorFlow*. (s. f.). TensorFlow. https://www.tensorflow.org/about?hl=es-419
5. Ángel. (s. f.-b). *¿Qué es TensorFlow y para qué sirve?* Incentro. https://www.incentro.com/es-ES/blog/que-es-tensorflow
6. *R: The R Project for Statistical Computing*. (s. f.). https://www.r-project.org/
7. *Introducción al manejo de datos con {dplyr} | Blog*. (s. f.). Blog. https://bastianolea.rbind.io/blog/r_introduccion/dplyr_intro/
8. *Manipulación de datos rápida y efectiva con dplyr - Máxima Formación*. (18 marzo de 2025). Máxima Formación. https://www.maximaformacion.es/blog-ciencia-datos/manipulacion-de-datos-rapida-y-efectiva-con-dplyr/
9. Lgayhardt. (s. f.). *Uso de Tidyverse - Microsoft Fabric*. Microsoft Learn. https://learn.microsoft.com/es-es/fabric/data-science/r-use-tidyverse
10. GeeksforGeeks. (15 julio de 2025). *tidyr Package in R Programming*. GeeksforGeeks. https://www-geeksforgeeks-org.translate.goog/r-language/tidyr-package-in-r-programming/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc
11. *The Scala programming language*. (s. f.). https://www.scala-lang.org/
12. *¿Qué es Apache Spark?* (19 octubre de 2023). IONOS Digital Guide. https://www.ionos.mx/digitalguide/servidores/know-how/que-es-apache-spark/

13. Holdsworth, J., & Kosinski, M. (19 marzo de 2025). Conjunto de Datos Distribuido Resiliente. *¿Qué es un Conjunto de Datos Distribuido Resiliente (RDD)?* https://www.ibm.com/mx-es/think/topics/resilient-distributed-dataset
14. Krykowski, M. (18 abril de 2024). *Top 15 Scala Libraries for Data Science in 2023*. Scalac - Software Development Company - Akka, Kafka, Spark, ZIO. https://scalac-io.translate.goog/blog/top-15-scala-libraries-for-data-science-in-2021/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc
15. *¿Qué es SQL? El lenguaje de bases de datos*. (18 octubre de 2024). IONOS Digital Guide. https://www.ionos.mx/digitalguide/servidores/know-how/que-es-sql/
16. G, L. J. C. (s. f.). *Aprende Frameworks de Desarrollo Web en Python*. https://entrenamiento-frameworks-web-python.readthedocs.io/es/latest/leccion2/sqlalchemy.html
17. *Sequelize*. (s. f.). Feature-rich ORM For Modern TypeScript & JavaScript. https://sequelize.org/
18. *Introducción · Programación básica con Julia*. (s. f.). https://hedero.webs.upv.es/julia-basico/
19. *Welcome · Flux*. (s. f.). https://fluxml.ai/Flux.jl/stable/