# Proyecto: Ingeniería de Datos + Cloud

Desafíos 1 y 2 — Generación y Almacenamiento de Datos Simulados

## 1. Objetivo del Proyecto

El propósito fue construir una simulación de flujo de datos para un entorno de ingeniería de datos basado en la nube. El ejercicio se centró en los dos primeros desafíos: Desafío 1 (Generación automática de datos estructurados) y Desafío 2 (Persistencia de los datos en archivos CSV).

## 2. Tecnologías y Herramientas Utilizadas

|  |  |
| --- | --- |
| Componente | Descripción |
| Lenguaje | Python 3.13.5 |
| Librerías | pandas, numpy, Faker |
| Entorno | Jupyter Notebook (test\_test.ipynb) |
| Formato de salida | CSV (archivos planos) |
| Arquitectura Cloud Referencial | Azure Blob Storage + Data Factory + PostgreSQL + Power BI |

## 3. Desafío 1 – Generación Automática de Datos

Se desarrolló un script en Python que genera datasets sintéticos de tres entidades relacionadas: Departamentos, Puestos de trabajo y Empleados. El script crea 400 empleados simulados con relaciones coherentes entre las llaves foráneas.

### Archivos generados

|  |  |
| --- | --- |
| Archivo | Descripción |
| departamento.csv | Contiene los códigos y nombres de los departamentos. |
| puestos.csv | Define los puestos y sus rangos salariales. |
| empleados.csv | Lista de empleados con información básica y relaciones con las otras tablas. |

## 4. Desafío 2 – Almacenamiento y Formato de Datos

### Los datos generados se exportaron en formato CSV, con el propósito de garantizar su portabilidad, legibilidad y compatibilidad con diversas herramientas locales.

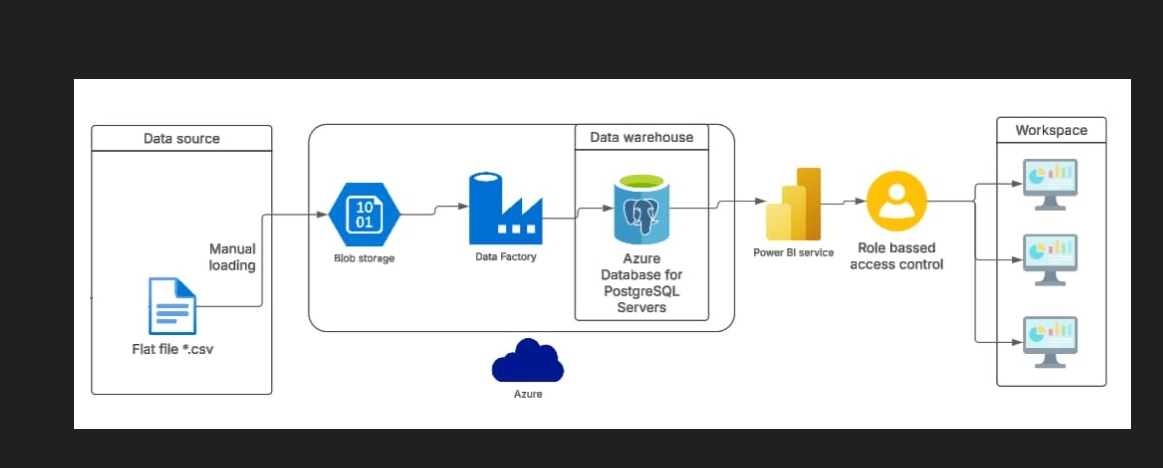
### No obstante, en escenarios de Big Data o procesamiento masivo de información, se recomienda utilizar el formato Parquet, ya que ofrece mayor eficiencia en almacenamiento, compresión y velocidad de lectura para volúmenes de datos de gran escala.

### Comparación de formatos

|  |  |  |
| --- | --- | --- |
| Criterio | CSV | Parquet |
| Facilidad de uso | Alta: compatible con Excel, Power BI, bases SQL | Requiere librerías adicionales |
| Peso y compresión | Mayor tamaño | Menor tamaño por compresión columnar |
| Uso recomendado | Procesos simples o cargas manuales | Procesos analíticos masivos o Data Lake |

## 5. Arquitectura Referencial (Diseño Cloud)

Aunque la implementación cloud no se realizó por motivos de costo, se propuso la siguiente arquitectura lógica.



Flujo de datos:  
1. Data Source (Flat files) → archivos CSV generados localmente.  
2. Azure Blob Storage → capa de almacenamiento en la nube.  
3. Azure Data Factory → orquestador para ingestión y movimiento de datos.  
4. Azure Database for PostgreSQL → base de datos relacional tipo Data Warehouse.  
5. Power BI Service → capa de consumo y visualización con control de acceso basado en roles.

## 6. Resultados

• Se generaron tres datasets coherentes y relacionales con 10 registros de empleados.  
• Se exportaron correctamente en formato CSV con estructura tabular.  
• Se documentó la arquitectura propuesta para su escalabilidad en Azure.

## 7. Conclusiones

• La generación automática de datos con Python permite simular entornos reales sin depender de fuentes productivas.  
• El formato CSV es adecuado para prototipos y flujos iniciales para este caso.  
• La arquitectura diseñada permitiría escalar el proceso hacia un pipeline completo de DataOps.