### **Examen de Conocimientos Técnicos**

### **Versión 1: Examen para Candidatos**

#### **Parte 1: Python**

### **Ejercicio 1: Limpieza de Datos**

**Enunciado:**Tienes el siguiente conjunto de datos en formato tabular que representa transacciones bancarias:

| **transaction\_id** | **customer\_id** | **product\_type** | **transaction\_amount** | **status** |
| --- | --- | --- | --- | --- |
| 1 | 1001 | Loan | 1500 | Completed |
| 2 | 1002 | Deposit | 2000 | Completed |
| 3 | 1003 | CreditCard | 2500 | Pending |
| 4 | 1004 | Loan | None | Completed |
| 5 | 1005 | Deposit | 1000 | Failed |

**Pregunta:**

1. Elimina las transacciones con status "Failed".
2. Rellena los valores nulos de la columna transaction\_amount con el promedio de los valores existentes.
3. Genera un resumen que muestre el monto total de transacciones por product\_type.
4. Guarda los datos limpios y el resumen en dos archivos CSV separados.

### 

### **Ejercicio 2: Identificación y Solución de Problemas**

**Enunciado:**Tienes el siguiente conjunto de datos que contiene duplicados y valores nulos:

| **transaction\_id** | **customer\_id** | **product\_type** | **transaction\_amount** | **status** |
| --- | --- | --- | --- | --- |
| 1 | 1001 | Loan | 1500 | Completed |
| 2 | 1002 | Deposit | None | Pending |
| 2 | 1002 | Deposit | None | Pending |
| 3 | 1003 | CreditCard | 2500 | Completed |
| 4 | 1004 | Loan | None | Pending |

**Pregunta:**

Identifica y elimina los duplicados basados en transaction\_id.

Rellena los valores nulos en la columna transaction\_amount con el promedio de los valores existentes.

1. Actualiza las transacciones con estatus "Pending" para que tengan un nuevo estatus "Processing".

#### **Parte 2: PySpark**

### **Ejercicio 3: Optimización de Procesos de Analítica**

**Enunciado:**Tienes el siguiente conjunto de datos con transacciones bancarias:

| **transaction\_id** | **customer\_id** | **product\_type** | **transaction\_amount** | **status** |
| --- | --- | --- | --- | --- |
| 1 | 1001 | Loan | 1500 | Completed |
| 2 | 1002 | Deposit | 2000 | Completed |
| 3 | 1003 | CreditCard | 2500 | Pending |
| 4 | 1004 | Loan | 3000 | Completed |
| 5 | 1005 | Deposit | 1000 | Failed |

**Pregunta:**

1. Particiona los datos por product\_type para optimizar el procesamiento.
2. Guarda los datos particionados en formato Parquet.

#### **Pregunta práctica:**

#### Dado el siguiente DataFrame en PySpark:

#### data = [("A", 100), ("B", 200), ("A", 300), ("B", None)]

#### columns = ["product\_type", "amount"]

#### df = spark.createDataFrame(data, columns)

#### Escribe el código para reemplazar los valores nulos con 0, agrupar por product\_type y calcular la suma de la columna amount.

#### 

#### **Pregunta práctica avanzada:**

#### Optimiza el siguiente código para mejorar su rendimiento:

#### **df = df.withColumn("new\_col", col("existing\_col") \* 10).filter(col("existing\_col") > 1000)**

#### **Parte 3: SQL**

### **Ejercicio 4: Query Complejo**

**Enunciado:**Usa las siguientes tablas para resolver el problema:

1. **Tabla transactions**:

| **transaction\_id** | **customer\_id** | **transaction\_amount** | **date** |
| --- | --- | --- | --- |
| 101 | 1001 | 1500 | 2024-01-01 |
| 102 | 1002 | 2000 | 2024-01-02 |
| 103 | 1003 | 2500 | 2024-01-03 |
| 104 | 1004 | 3000 | 2024-01-04 |
| 105 | 1005 | 1000 | 2024-01-05 |

1. **Tabla customers**:

| customer\_id | customer\_name | region |
| --- | --- | --- |
| 1001 | Alice | North |
| 1002 | Bob | South |
| 1003 | Charlie | North |
| 1004 | David | East |
| 1005 | Eva | South |

**Pregunta:**

1. Genera un reporte combinando las tablas para mostrar:
   * Nombre del cliente (customer\_name).
   * Región (region).
   * Monto total de transacciones (total\_amount).
   * Número de transacciones realizadas (transaction\_count).
2. Filtra el reporte para incluir solo clientes de la región "North".

### 

### 

### **Ejercicio 5: Proceso ETL**

**Enunciado:**Tienes el siguiente conjunto de datos de clientes:

| **customer\_id** | **customer\_name** | **region** | **account\_balance** |
| --- | --- | --- | --- |
| 1001 | Alice | North | 1500 |
| 1002 | Bob | South | 2000 |
| 1003 | Charlie | North | None |
| 1004 | David | East | 1800 |
| 1005 | Eva | South | 500 |

**Pregunta:**

1. Rellena los valores nulos de la columna account\_balance con 0.
2. Añade una nueva columna balance\_category para clasificar a los clientes según su balance:
   * "Low" si el balance es menor a 1000.
   * "Medium" si el balance está entre 1000 y 3000.
   * "High" si el balance es mayor a 3000.
3. Guarda los datos transformados en una nueva tabla llamada customers\_transformed.

### 