g2jfvtudk

March 3, 2025

1 Variables Compartidas en Apache Spark

1.1 Introducción

Las variables compartidas son tipos de variables especiales que se pueden usar dentro de las tareas de Spark. Aunque se crean en el Driver, se actualizan y se les da uso en los workers, dentro de cada tarea de Spark.

Las variables compartidas permiten: - **Broadcast**: Distribuir datos de solo lectura a todos los nodos - **Acumuladores**: Agregar valores desde los ejecutores al driver

Son esenciales para optimizar operaciones y recolectar métricas.

1.2 1. Broadcast Variables

Propiedades: - Datos de solo lectura - Almacenados en caché en cada nodo - Útiles para tablas de lookup grandes

Consideraciones: - Usar para datos que caben en memoria y > 100KB - No modificar después de crear - Limpiar variables broadcast después de usar

[0]:
```
### Ejemplo: Mapeo de Códigos de País

# Datos de muestra
data = [
    ("user1", "US"),
    ("user2", "UK"),
    ("user3", "IN"),
    ("user4", "AU")
]
rdd = sc.parallelize(data)

# Diccionario de mapeo de códigos
country_map = {"US": "United States", "UK": "United Kingdom", "IN": "India"}

# Crear variable broadcast
broadcast_var = sc.broadcast(country_map)

# Función para mapear códigos usando broadcast
def map_country(user):
```

```
    code = user[1]
    return (user[0], broadcast_var.value.get(code, "Desconocido"))

resultado = rdd.map(map_country).collect()
print(resultado)
```

```
[('user1', 'United States'), ('user2', 'United Kingdom'), ('user3', 'India'),
('user4', 'Desconocido')]
```

1.3 2. Acumuladores

Propiedades: - Variables de solo adición - Suma agregada disponible en driver - Útiles para contadores o métricas

Importante: - Solo el driver puede leer el valor final - Se actualizan una vez por tarea con éxito - Usar acciones para garantizar ejecución

[0]:
```
### Ejemplo: Contar Registros Inválidos

# Inicializar acumulador
acumulador_errores = sc.accumulator(0)

# Datos con posibles errores
datos = ["123", "456", "abc", "789", "def"]
rdd = sc.parallelize(datos)

def validar_numero(texto):
    if texto.isdigit():
        return int(texto)
    else:
        acumulador_errores.add(1)  # Incrementar acumulador
        return None

numeros = rdd.map(validar_numero)

# Forzar ejecución para actualizar acumulador
numeros.count()

print("Errores detectados:", acumulador_errores.value)
```

```
Errores detectados: 2
```