Laboratório Prático: Chaves de Mensagem e Offset do Kafka

Tempo estimado necessário: **40** minutos

Introdução

Neste laboratório, você manterá os fluxos de mensagens adicionados às partições em tópicos em um broker Kafka ordenados em sua ordem e estado de publicação originais, usando offset e grupos.

Objetivos

Após concluir este laboratório, você será capaz de:

- Usar chaves de mensagem para manter os fluxos de mensagens ordenados em seu estado e ordem de publicação originais
- Usar offset do consumidor para controlar e rastrear as posições sequenciais das mensagens nas partições do tópico

Sobre o Skills Network Cloud IDE

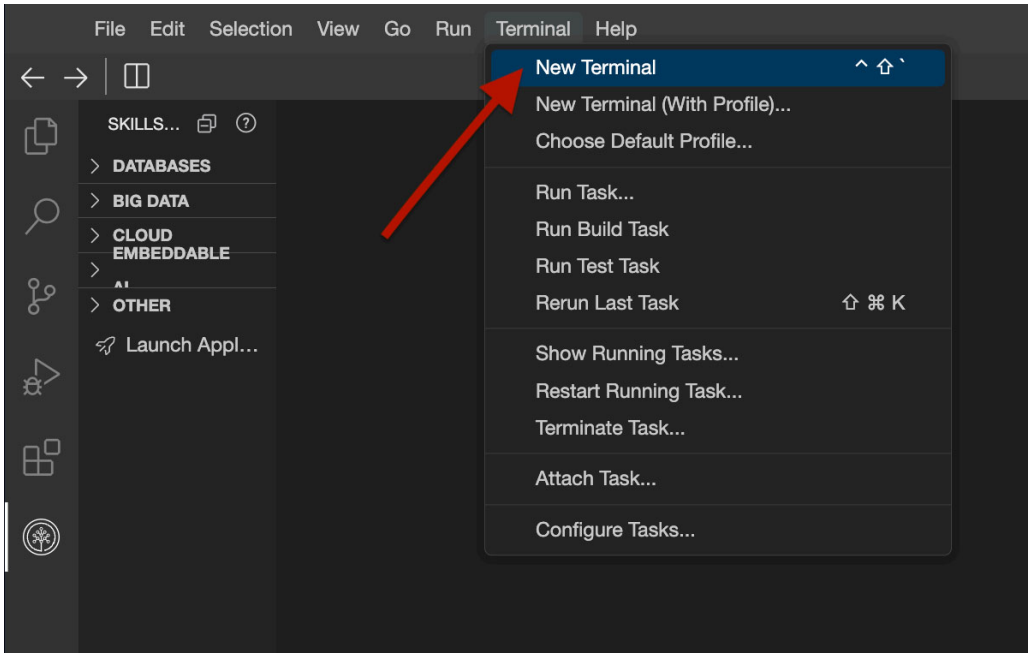
O Skills Network Cloud IDE (baseado no Theia e Docker) fornece um ambiente para laboratórios práticos relacionados a cursos e projetos. O Theia é um IDE (Ambiente de Desenvolvimento Integrado) de código aberto que pode ser executado em desktops ou na nuvem. Para completar este laboratório, usaremos o Cloud IDE baseado no Theia rodando em um contêiner Docker.

Aviso importante sobre este ambiente de laboratório

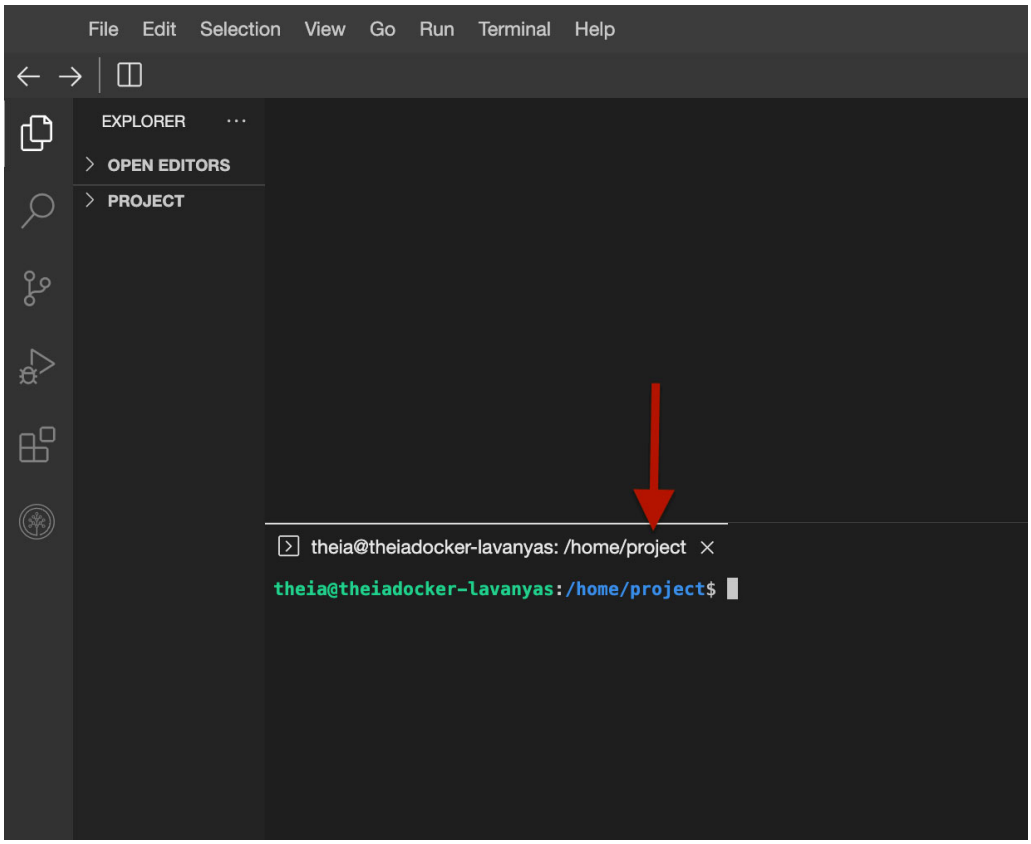
Por favor, esteja ciente de que as sessões para este ambiente de laboratório não são persistentes. Um novo ambiente é criado para você toda vez que você se conecta a este laboratório, e qualquer dado que você possa ter salvo em uma sessão anterior será perdido. Para evitar perder seus dados, planeje concluir estes laboratórios em uma única sessão.

Exercício 1: Baixar e extrair Kafka

1. Abra um novo terminal, clicando na barra de menu e selecionando **Terminal**->**Novo Terminal**.



Isso abrirá um novo terminal na parte inferior da tela.



Execute os comandos abaixo no terminal recém-aberto.

Nota: Você pode copiar o código clicando no pequeno botão de copiar no canto inferior direito do código abaixo e, em seguida, colá-lo onde desejar.

2. Baixe o Kafka executando o comando abaixo:

```
wget https://downloads.apache.org/kafka/3.8.0/kafka_2.13-3.8.0.tgz
```

3. Extraia o Kafka do arquivo zip executando o comando abaixo.

```
tar -xzf kafka_2.13-3.8.0.tgz
```

Nota: Este comando cria um diretório chamado `kafka_2.13-3.8.0` no diretório atual.

Exercício 2: Configurar KRaft e iniciar o servidor

1. Mude para o diretório `kafka_2.13-3.8.0`.

```
cd kafka_2.13-3.8.0
```

2. Gere um UUID de Cluster que identificará exclusivamente o cluster Kafka.

```
KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"
```

Este ID de cluster será usado pelo controlador KRaft.

3. O KRaft requer que os diretórios de log sejam configurados. Execute o seguinte comando para configurar os diretórios de log passando o ID do cluster.

```
bin/kafka-storage.sh format -t $KAFKA_CLUSTER_ID -c config/kraft/server.properties
```

4. Agora que o KRaft está configurado, você pode iniciar o servidor Kafka executando o seguinte comando.

```
bin/kafka-server-start.sh config/kraft/server.properties
```

Você pode ter certeza de que começou quando você vê uma saída como esta:

Exercício 3: Crie um tópico e produtor para processar transações de caixas eletrônicos bancários

Em seguida, você criará um tópico `bankbranch` para processar mensagens de máquinas de caixas eletrônicos de agências bancárias.

Suponha que as mensagens venham do caixa eletrônico na forma de um simples objeto JSON, incluindo um ID de caixa eletrônico e um ID de transação, como no seguinte exemplo:

```
{"atmid": 1, "transid": 100}
```

Para processar as mensagens do caixa eletrônico, você precisará primeiro criar um novo tópico chamado `bankbranch`.

1. Abra um novo terminal e mude para o diretório `kafka_2.13-3.8.0`.

   ```
   cd kafka_2.13-3.8.0
   ```

2. Crie um novo tópico usando o `--topic` argumento com o nome `bankbranch`. Para simplificar a configuração do tópico e explicar melhor como a chave da mensagem e o deslocamento do consumidor funcionam, você especifica o argumento `--partitions 2` para criar duas partições para este tópico. Para comparar as diferenças, você pode tentar outras configurações de `partitions` para este tópico.

   ```
   bin/kafka-topics.sh --create --topic bankbranch --partitions 2 --bootstrap-server localhost:9092
   ```

3. Liste todos os tópicos para verificar se `bankbranch` foi criado com sucesso.

   ```
   bin/kafka-topics.sh --bootstrap-server localhost:9092 --list
   ```

4. Você também pode usar o `--describe` comando para verificar os detalhes do tópico `bankbranch`.

   ```
   bin/kafka-topics.sh --bootstrap-server localhost:9092 --describe --topic bankbranch
   ```

Você pode visualizar o `bankbranch` como duas partições, `Partition 0` e `Partition 1`. Se nenhuma chave de mensagem for especificada, as mensagens serão publicadas nessas duas partições em uma sequência alternada assim:

`Partition 0` -> `Partition 1` -> `Partition 0` -> `Partition 1` ...

Em seguida, você pode criar um produtor para publicar mensagens de transação de caixa eletrônico.

5. Execute o seguinte comando na mesma janela do terminal com os detalhes do tópico para criar um produtor para o tópico `bankbranch`.

   ```
   bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic bankbranch
   ```

6. Para produzir as mensagens, procure o ícone > e adicione as seguintes mensagens do caixa eletrônico após o ícone:

   ```
   {"atmid": 1, "transid": 100}
   ```

```
{"atmid": 1, "transid": 101}
```

```
{"atmid": 2, "transid": 200}
```

```
{"atmid": 1, "transid": 102}
```

```
{"atmid": 2, "transid": 201}
```

Então, crie um consumidor em uma nova janela do terminal para consumir essas cinco novas mensagens.

7. Abra um novo terminal e mude para o diretório `kafka_2.13-3.8.0`.

```
cd kafka_2.13-3.8.0
```

8. Em seguida, inicie um novo consumidor para assinar o tópico `bankbranch`:

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --from-beginning
```

Você deve ser capaz de visualizar as cinco novas mensagens que publicou. No entanto, as mensagens podem não ser consumidas na mesma ordem em que foram publicadas. Normalmente, você precisará manter as mensagens consumidas ordenadas na sua ordem original de publicação, especialmente para casos de uso críticos,

como transações financeiras.

Exercício 4: Produzir e consumir com chaves de mensagem

Nesta etapa, você usará chaves de mensagem para garantir que mensagens com a mesma chave sejam consumidas na mesma ordem em que foram publicadas. No back-end, mensagens com a mesma chave são publicadas na mesma partição e sempre serão consumidas pelo mesmo consumidor. Assim, a ordem de publicação original é mantida do lado do consumidor.

Neste ponto, você deve ter os seguintes três terminais abertos no Cloud IDE:

- Terminal do Kafka Server
- Terminal do Produtor
- Terminal do Consumidor

Nos próximos passos, você alternará frequentemente entre esses terminais.

- Primeiro, vá para o terminal do consumidor e pare o consumidor usando `Ctrl` + `C` (Windows) ou `COMMAND` + `.` (Mac).

- Em seguida, mude para o terminal do Produtor e pare o produtor anterior.

Agora você começará um novo produtor e consumidor usando chaves de mensagem. Você pode iniciar um novo produtor com as seguintes opções de chave de mensagem:

- `--property parse.key=true` para fazer o produtor analisar chaves de mensagem
- `--property key.separator=:` define o separador de chave como o caractere `:`, então nossa mensagem com chave agora se parece com o seguinte par chave-valor:
    - `1:{"atmid": 1, "transid": 102}`
      Aqui, a chave da mensagem é `1`, que também corresponde ao ID do caixa eletrônico, e o valor é o objeto JSON da transação, `{"atmid": 1, "transid": 102}`.

1. Inicie um novo produtor com a chave de mensagem ativada.

   ```
   bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic bankbranch --property parse.key=true --property key.separator=:
   ```

2. Assim que você ver o símbolo >, pode começar a produzir as seguintes mensagens, onde você define cada chave para corresponder ao ID do caixa eletrônico para cada mensagem:

   ```
   1:{"atmid": 1, "transid": 103}
   ```

   ```
   1:{"atmid": 1, "transid": 104}
   ```

   ```
   2:{"atmid": 2, "transid": 202}
   ```

   ```
   2:{"atmid": 2, "transid": 203}
   ```

```
1:{"atmid": 1, "transid": 105}
```

3. Em seguida, volte para o terminal do consumidor e inicie um novo consumidor com os argumentos `--property print.key=true` e `--property key.separator=:` para imprimir as chaves.

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --from-beginning --property print.key=true --property key.sep
```

Agora, você deve ver que as mensagens com a mesma chave estão sendo consumidas na mesma ordem (por exemplo: `trans102 -> trans103 -> trans104`) em que foram publicadas.

Cada partição de tópico mantém sua fila de mensagens, e novas mensagens são enfileiradas (adicionadas ao final da fila) à medida que são publicadas na partição. Uma vez consumidas, as mensagens mais antigas são desenfileiradas e não estão mais disponíveis para consumo.

Lembre-se de que, com duas partições e sem chaves de mensagem especificadas, as mensagens de transação foram publicadas nas duas partições em rotação:

- Partição 0: `[{"atmid": 1, "transid": 100}, {"atmid": 2, "transid": 200}, {"atmid": 2, "transid": 201}]`
- Partição 1: `[{"atmid": 1, "transid": 101}, {"atmid": 1, "transid": 102}]`

Como você pode ver, as mensagens de transação de `atm1` e `atm2` foram espalhadas por ambas as partições. Seria difícil desenrolar isso e consumir mensagens de um único caixa eletrônico na mesma ordem em que foram publicadas.

No entanto, com a chave da mensagem especificada como o valor `atmid`, as mensagens dos dois caixas eletrônicos ficarão assim:

- Partição 0: `[{"atmid": 1, "transid": 103}, {"atmid": 1, "transid": 104}, {"atmid": 1, "transid": 105}]`
- Partição 1: `[{"atmid": 2, "transid": 202}, {"atmid": 2, "transid": 203}]`

Mensagens com a mesma chave sempre serão publicadas na mesma partição, de modo que sua ordem de publicação será preservada dentro da fila de mensagens de cada partição.

Assim, você pode manter os estados ou ordens das transações para cada caixa eletrônico.

Exercício 5: Deslocamento do consumidor

As partições de tópicos mantêm as mensagens publicadas em uma sequência, como uma lista. O deslocamento da mensagem indica a posição de uma mensagem na sequência. Por exemplo, o deslocamento de uma Partição 0 vazia de `bankbranch` é `0`, e se você publicar a primeira mensagem na partição, seu deslocamento será `1`.

Usando deslocamentos no consumidor, você pode especificar a posição inicial para o consumo de mensagens, como do início para recuperar todas as mensagens ou de algum ponto posterior para recuperar apenas as mensagens mais recentes.

Grupo de consumidores

Além disso, você normalmente agrupa consumidores relacionados juntos como um grupo de consumidores.

Por exemplo, você pode querer criar um consumidor para cada caixa eletrônico no banco e gerenciar todos os consumidores relacionados a caixas eletrônicos juntos em um grupo.

Então, vamos ver como criar um grupo de consumidores, o que é na verdade muito fácil com o argumento `--group`.

1. No terminal do consumidor, pare o consumidor anterior se ele ainda estiver em execução.

2. Execute o seguinte comando para criar um novo consumidor dentro de um grupo de consumidores chamado `atm-app`:

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --group atm-app
```

Após o consumidor dentro do grupo de consumidores `atm-app` ser iniciado, você não deve esperar que nenhuma mensagem seja consumida. Isso ocorre porque os offsets para ambas as partições já atingiram o final. Em outras palavras, consumidores anteriores já consumiram todas as mensagens e, portanto, as enfileiraram.

Você pode verificar isso conferindo os detalhes do grupo de consumidores.

3. Pare o consumidor.
4. Mostre os detalhes do grupo de consumidores `atm-app`:

```
bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group atm-app
```

Agora você deve ver as informações de offset para o tópico `bankbranch`:

Lembre-se de que você publicou `10` mensagens no total, e você pode ver que a coluna `CURRENT-OFFSET` da partição 1 e da partição 0 soma 10 mensagens.

A coluna `LOG-END-OFFSET` indica o último offset ou o final da sequência. Assim, ambas as partições atingiram o final de suas filas e não há mais mensagens disponíveis para consumo.

Enquanto isso, você pode verificar a coluna `LAG`, que representa o número de mensagens não consumidas para cada partição.

Atualmente, está `0` para todas as partições, como esperado.

A seguir, vamos produzir mais mensagens e ver como os offsets mudam.

5. Volte para o terminal do produtor anterior e publique mais duas mensagens:

```
1:{"atmid": 1, "transid": 106}
```

```
2:{"atmid": 2, "transid": 204}
```

6. Volte para o terminal do consumidor e verifique os detalhes do grupo de consumidores novamente.

```
bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group atm-app
```

Você deve notar que ambos os deslocamentos foram aumentados em 1, e as colunas `LAG` para ambas as partições se tornaram `1`. Isso significa que você tem uma nova mensagem para cada partição a ser consumida.

7. Inicie o consumidor novamente e veja se as duas novas mensagens serão consumidas.

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --group atm-app
```

Ambas as partições chegaram ao fim mais uma vez.

Redefinir deslocamento

A seguir, vamos ver como você pode definir as partições para consumir as mensagens novamente desde o início através da redefinição do deslocamento.

Você pode redefinir o índice com o argumento `--reset-offsets`.

Primeiro, vamos tentar redefinir o deslocamento para a posição mais antiga (início) usando `--reset-offsets --to-earliest`.

1. Pare o consumidor anterior se ele ainda estiver em execução e execute o seguinte comando para redefinir o deslocamento.

```
bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092  --topic bankbranch --group atm-app --reset-offsets --to-earliest --execute
```

Agora, os offsets foram definidos como 0 (o começo).

2. Inicie o consumidor novamente:

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --group atm-app
```

Você deve ver que todas as 12 mensagens foram consumidas e que todos os deslocamentos alcançaram novamente os finais da partição.

Você pode redefinir o deslocamento para qualquer posição. Por exemplo, vamos redefinir o deslocamento para que você consuma apenas as duas últimas mensagens.

3. Pare o consumidor anterior.
4. Desloque o deslocamento para a esquerda em duas usando `--reset-offsets --shift-by -2`:

```
bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092  --topic bankbranch --group atm-app --reset-offsets --shift-by -2 --execute
```

5. Se você executar o consumidor novamente, deverá ver que consumiu quatro mensagens, 2 para cada partição:

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic bankbranch --group atm-app
```

6. Pare seu produtor, consumidor e o servidor Kafka.

Autores

Lavanya T S

Outros Contribuintes

Yan Luo