

Laboratório Prático: Construa um Pipeline ETL de Streaming usando Kafka



Tempo estimado: 45 minutos.

Cenário do projeto

Você é um engenheiro de dados em uma empresa de consultoria em análise de dados. Você foi designado para um projeto que visa descongestionar as rodovias nacionais analisando os dados de tráfego das diferentes praças de pedágio. À medida que um veículo passa por uma praça de pedágio, os dados do veículo, como `vehicle_id`, `vehicle_type`, `toll_plaza_id` e `timestamp`, são transmitidos para o Kafka. Seu trabalho é criar um pipeline de dados que colete os dados de streaming e os carregue em um banco de dados.

Objetivos

Nesta tarefa, você criará um pipeline de dados de streaming realizando os seguintes passos:

- Iniciar um servidor de banco de dados MySQL
- Criar uma tabela para armazenar os dados de pedágio
- Iniciar o servidor Kafka
- Instalar o driver Kafka para Python
- Instalar o driver MySQL para Python
- Criar um tópico chamado `toll` no Kafka
- Baixar o programa gerador de dados de streaming
- Personalizar o programa gerador para transmitir para o tópico `toll`
- Baixar e personalizar o consumidor de dados de streaming
- Personalizar o programa consumidor para gravar em uma tabela de banco de dados MySQL
- Verificar se os dados transmitidos estão sendo coletados na tabela do banco de dados

Nota sobre capturas de tela

Ao longo deste laboratório, você será solicitado a tirar capturas de tela e salvá-las em seu dispositivo. Você precisará enviar as capturas de tela para revisão por pares. Você pode usar várias ferramentas gratuitas de captura de tela ou as teclas de atalho do seu sistema operacional (Alt + PrintScreen no Windows, por exemplo) para capturar as capturas de tela necessárias. Você pode salvar as capturas de tela com a extensão `.jpg` ou `.png`.

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 um desktop ou na nuvem. Para completar este laboratório, você usará o Cloud IDE baseado no Theia, executando 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. Quaisquer dados que você possa ter salvo em uma sessão anterior serão perdidos. Para evitar perder seus dados, planeje concluir esses laboratórios em uma única sessão.

Exercício 1: Baixar e extrair Kafka

1. Baixe o Kafka executando o comando abaixo.

```
wget https://archive.apache.org/dist/kafka/3.7.0/kafka_2.12-3.7.0.tgz
```

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

```
tar -xzf kafka_2.12-3.7.0.tgz
```

Nota: Este comando cria um diretório chamado kafka_2.12-3.7.0 no diretório atual.

Exercício 2: Configurar KRaft e iniciar o servidor

1. Mude para o diretório kafka_2.12-3.7.0.

```
cd kafka_2.12-3.7.0
```

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

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

Nota: O novo id do cluster gerado 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
```

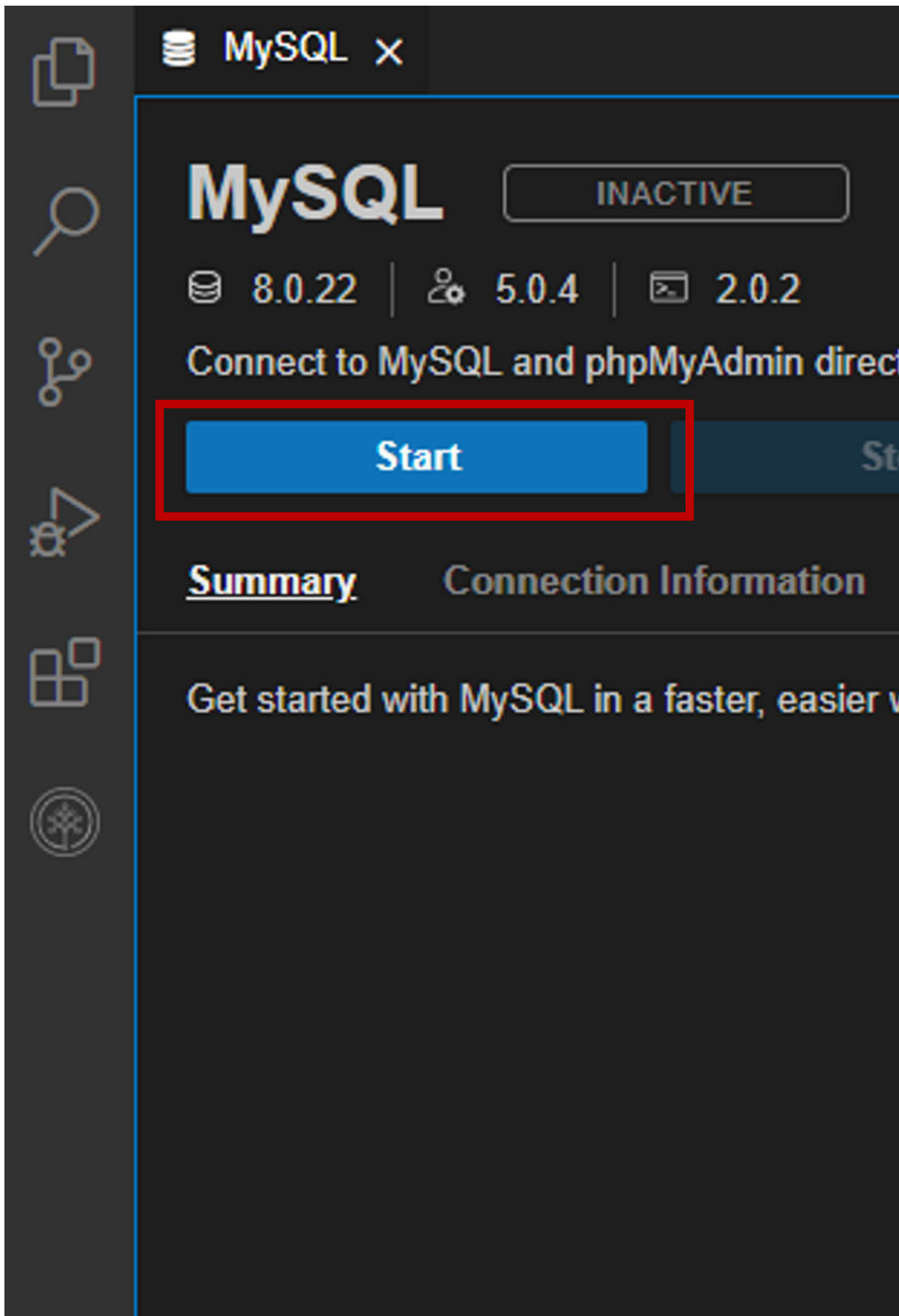
Nota: Você pode ter certeza de que o servidor Kafka iniciado gera informações de que o servidor foi iniciado com sucesso, juntamente com algumas mensagens adicionais, como log carregado.

```
[2024-06-12 02:19:51,129] INFO [BrokerServer id=1] Transition from S  
ver)  
[2024-06-12 02:19:51,130] INFO Kafka version: 3.7.0 (org.apache.kaf  
[2024-06-12 02:19:51,135] INFO Kafka commitId: 2ae524ed625438c5 (org  
[2024-06-12 02:19:51,135] INFO Kafka startTimeMs: 1718173191129 (org  
[2024-06-12 02:19:51,137] INFO [KafkaRaftServer nodeId=1] Kafka Serv  
[2024-06-12 02:20:25,678] INFO [ReplicaFetcherManager on broker 1] F  
ch-1, bankbranch-0) (kafka.server.ReplicaFetcherManager)  
[2024-06-12 02:20:25,718] INFO [LogLoader partition=bankbranch-1, d  
er state till offset 0 with message format version 2 (kafka.log.Uni  
[2024-06-12 02:20:25,722] INFO Created log for partition bankbranch-  
with properties {} (kafka.log.LogManager)  
[2024-06-12 02:20:25,725] INFO [Partition bankbranch-1 broker=1] No  
rtition bankbranch-1 (kafka.cluster.Partition)  
[2024-06-12 02:20:25,727] INFO [Partition bankbranch-1 broker=1] Log  
itial high watermark 0 (kafka.cluster.Partition)  
[2024-06-12 02:20:25,745] INFO [LogLoader partition=bankbranch-0, d  
er state till offset 0 with message format version 2 (kafka.log.Uni  
[2024-06-12 02:20:25,746] INFO Created log for partition bankbranch-  
with properties {} (kafka.log.LogManager)
```

Exercício 3: Iniciar o servidor MySQL e configurar o banco de dados

[Open MySQL Page in IDE](#)

1. Na página de lançamento, clique no botão **Iniciar**.



The image shows the MySQL Desktop Client interface. On the left is a dark sidebar with icons for file management, search, connections, a diagram tool, a table editor, and a help icon. The main panel has a dark theme. At the top, it says 'MySQL' with a close button. Below that, the status is 'INACTIVE'. The versions for MySQL (8.0.22), PHP (5.0.4), and Python (2.0.2) are listed. A red rectangle highlights a blue 'Start' button. Below the button are tabs for 'Summary' and 'Connection Information'. The 'Summary' tab is active, showing the text 'Get started with MySQL in a faster, easier v'.

MySQL x

MySQL

INACTIVE

8.0.22 | 5.0.4 | 2.0.2

Connect to MySQL and phpMyAdmin directly


Start

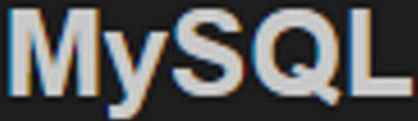
Summary Connection Information

Get started with MySQL in a faster, easier v



2. Uma vez que o servidor MySQL esteja iniciado, selecione a aba **Informações de Conexão**. A partir daí, copie a senha.

 MySQL x

 ACTIVE

8.0.22 | 5.0.4 | 2.0.2


Connect to MySQL and phpMyAdmin directly

Start

Summary Connection Information

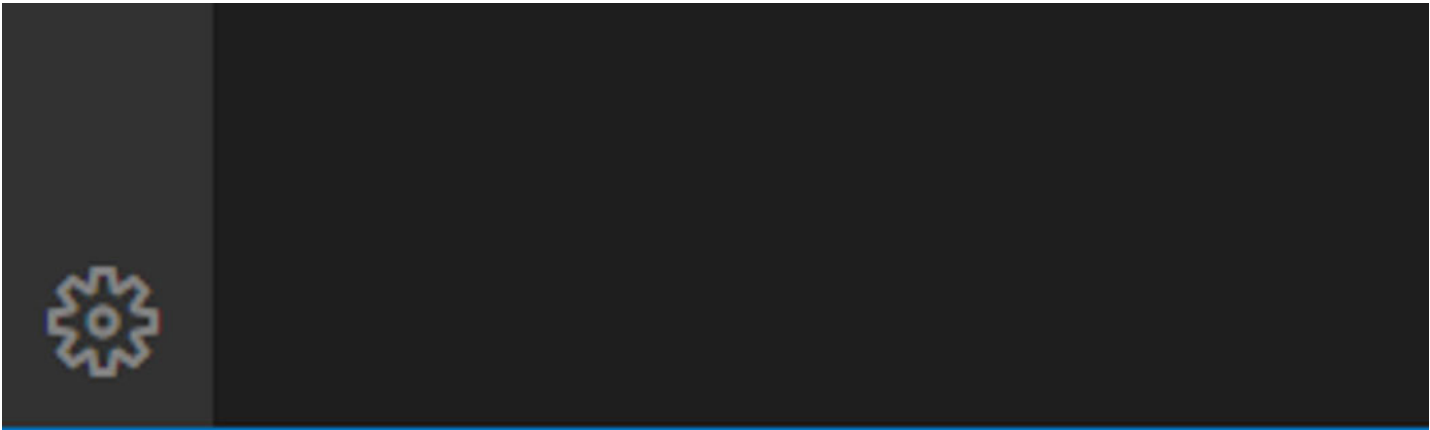
Your database and phpMyAdmin server are not running. Please check out the Details section.




You can manage MySQL via:



phpMyAdmin 



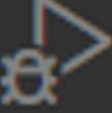
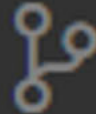


Or to interact with the database in the terminal:

MySQL CLI New Terminal





 MySQL 



MYSQL_URL:

`https://labs-mysc`

MySQL CLI Command:

`mysql --h`

MYSQL_COMMAND:

`mysql --hos`

MYSQL_PASSWORD:

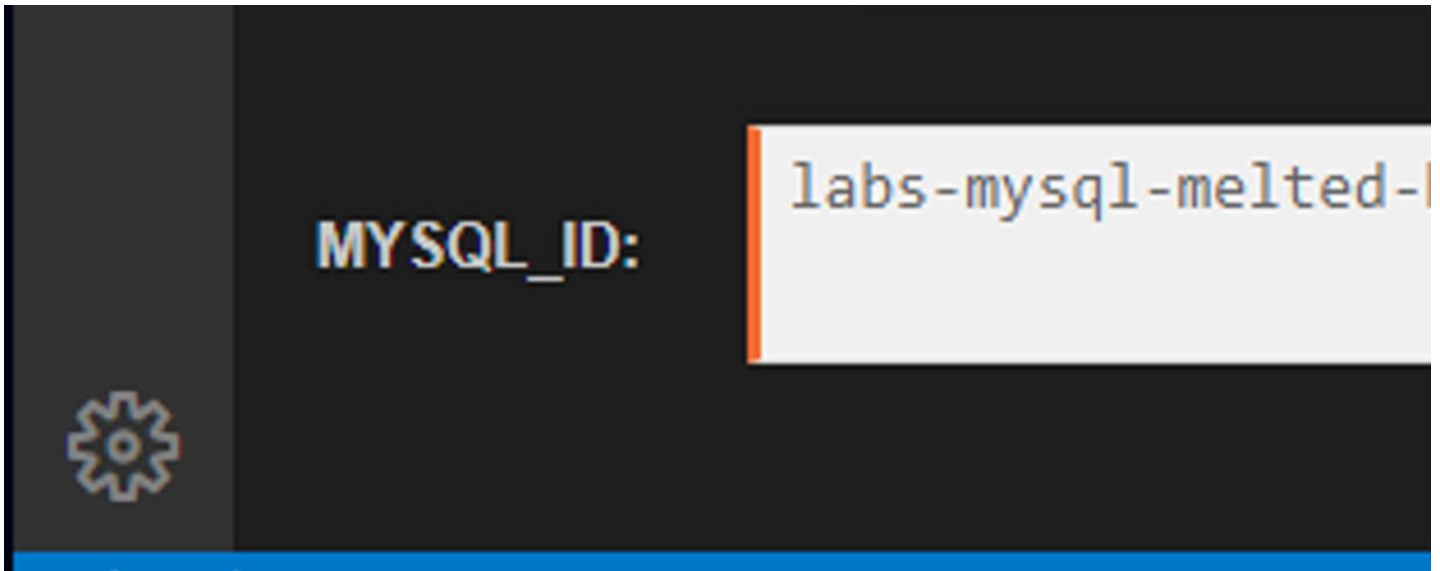
`vd6sFvnG62`

MYSQL_TITLE:

`MySQL Database`

about:blank

8/11



3. Conecte-se ao servidor MySQL usando o comando abaixo no terminal. Certifique-se de usar a senha fornecida quando o servidor MySQL iniciar. Por favor, anote a senha, pois você precisará dela mais tarde.

```
mysql --host=mysql --port=3306 --user=root --password=Replace your password
```

4. Crie um banco de dados chamado `tolldata`.

No prompt **mysql**>, execute o comando abaixo para criar o banco de dados.

```
create database tolldata;
```

5. Crie uma tabela chamada `livetolldata` com o esquema para armazenar os dados gerados pelo simulador de tráfego.

Execute o seguinte comando para criar a tabela:

```
use tolldata;  
create table livetolldata(timestamp datetime,vehicle_id int,vehicle_type char(15),toll_plaza_id smallint);
```

Nota: Esta é a tabela onde você armazenará todos os dados transmitidos que vêm do Kafka. Cada linha é um registro de quando um veículo passou por um determinado pedágio, junto com seu tipo e id anonimizado.

6. Desconecte-se do servidor MySQL.

```
exit
```

Exercício 4: Instalar os pacotes Python

1. Instale o módulo Python `kafka-python`. Este módulo Python ajudará você a se comunicar com o servidor Kafka. Ele pode ser usado para enviar e receber mensagens do Kafka.

```
pip3 install kafka-python
```

2. Instale o módulo Python `mysql-connector-python` usando o comando `pip`.

```
pip3 install mysql-connector-python==8.0.31
```

Este módulo Python ajudará você a interagir com o servidor MySQL.

Exercício 5: Criar pipeline de dados para dados de pedágio

1. Crie um tópico Kafka chamado `toll`.
2. Baixe o `toll_traffic_generator.py` do URL fornecido abaixo usando **wget**.

```
wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/toll_traffic_generato
```

3. Abra o código usando o editor pela opção “Menu → Arquivo → Abrir”.
4. Abra o `toll_traffic_generator.py` e defina o tópico como `toll`.
5. Execute o `toll_traffic_generator.py`.

```
python3 toll_traffic_generator.py
```

6. Baixe o `streaming-data-reader.py` do URL abaixo usando **wget**.

```
wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/vVxmU5uatDowvAIKRZrFjg/streaming-data-reader.py
```

7. Abra o `streaming-data-reader.py` e modifique os seguintes detalhes para que o programa possa se conectar ao seu servidor MySQL.

`TOPIC`

`DATABASE`

`USERNAME`

`PASSWORD`

8. Execute o `streaming-data-reader.py`.

```
python3 streaming-data-reader.py
```

9. Se você completou todas as etapas corretamente, os dados de pedágio de streaming serão armazenados na tabela `liveto11data`. Como último passo neste laboratório, abra o CLI do mysql e liste as 10 primeiras linhas na tabela `liveto11data`.

Authors

Ramesh Sannareddy

[Lavanya T S](#)

Other Contributors

Rav Ahuja

© IBM Corporation. Todos os direitos reservados.