

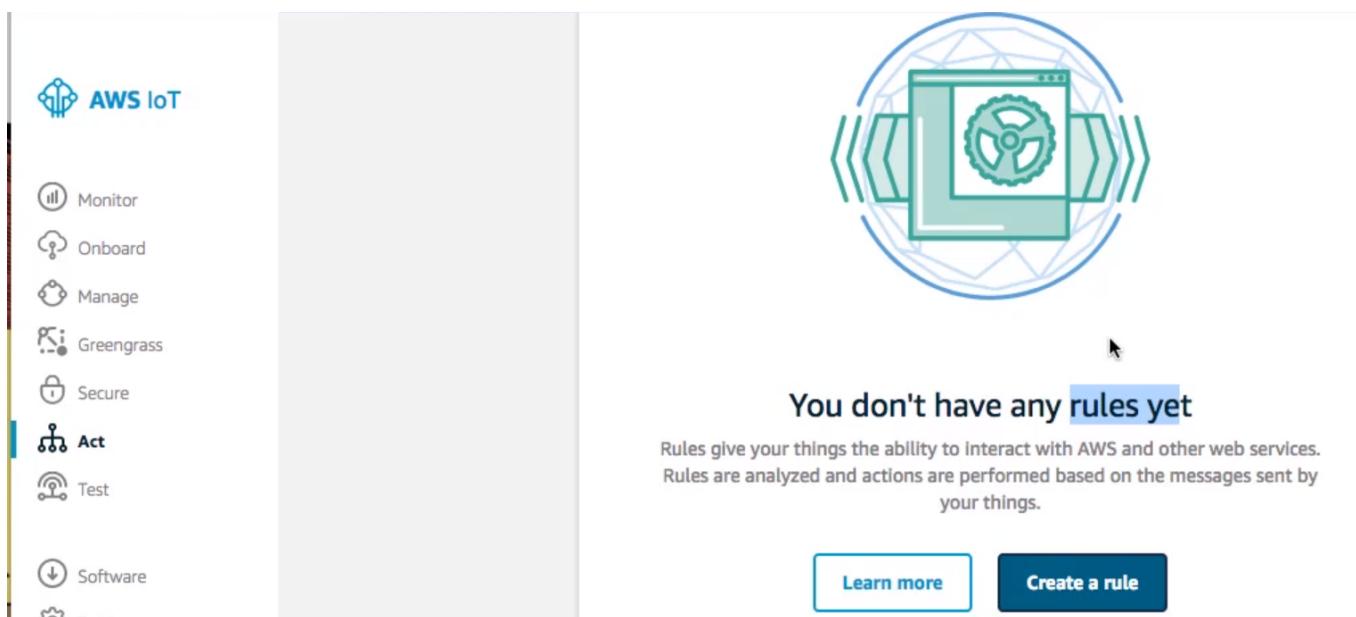
01

## Integração com DynamoDB

### Transcrição

Já progredimos bastante em nosso projeto onde agora temos o dispositivo conectado a plataforma da Amazon e o envio dos dados de temperatura funcionando. Faremos agora com que estes dados sejam armazenados no banco de dados. Neste caso, no **DynamoDB**.

Para gravar as informações no banco de dados, precisamos criar o que na plataforma está descrito como **rules** (*regras em português*) ou **act**, que é como se chama essa opção atualmente. Clicamos na opção **Act** do menu principal do AWS IoT e logo temos uma mensagem informando que não temos nenhuma regra criada ainda.



Clicamos no botão **Create a rule** e logo temos um formulário onde primeiramente informamos um nome para a regra e uma descrição. Ainda com a ideia de que podemos ter diversos dispositivos no mesmo projeto, agruparemos as regras de temperatura ou telemetria em uma regra só. Como nome usamos **TelemetryDB** e como descrição informamos **Gravação dos dados de telemetria no banco DynamoDB**.

Create a rule to evaluate messages sent by your things and specify what to do when a message is received (for example, write data to a DynamoDB table or invoke a Lambda function).

<b>Name</b>	TelemetryDB
<b>Description</b>	Gravação dos dados de telemetria no banco DynamoDB

Cada regra pode acessar todas as mensagens enviadas para a plataforma. Por isso temos que informar alguns critérios na regra para que sejam selecionadas apenas o que queremos de fato capturar. Os próximos campos são exatamente para isso.

Primeiro teremos um seletor de versão do SQL, que deixamos como está, usando a versão mais recente. Segundo temos um campo que ilustra como a seleção dos dados está sendo feita, ele é alterado dinamicamente de acordo com os outros campos e não podemos alterá-lo. É o campo `Rule query statement`.

O terceiro campo é chamado de `attribute`. Este é usado para informar qual campo da mensagem queremos selecionar, neste caso, usaremos o asterisco para selecionar todos os campos, ou seja, os campos de `timestamp`, `temperatura` e `sensor`. Por último temos o campo `Topic filter` onde informamos o tópico que será feita a seleção dos campos, neste caso será `telemetria/temperatura`. Como não teremos nenhuma condição específica para poder armazenar os dados, deixaremos o campo `condition` em branco. Queremos todos os dados.

## Message source

Indicate the source of the messages you want to process with this rule.

### Using SQL version

2016-03-23



### Rule query statement

```
SELECT * FROM 'telemetria/temperatura'
```

### Attribute

\*

### Topic filter



telemetria/temperatura

### Condition

e.g. `temperature > 75`

Até o momento já temos o nome, descrição e regras de captura dos dados, mas falta a ação que será feita sempre que o dado for capturado. Este é o último passo que vamos configurar neste formulário. Logo após o campo `condition` temos um botão `add action`, ao clicar no botão teremos uma lista de possíveis ações a serem realizadas. Selecionaremos a que diz `Split message into multiple columns of a database table (DynamoDBv2)`.

## Select an action

Select an action.



Insert a message into a DynamoDB table  
DYNAMODB



Split message into multiple columns of a database table (DynamoDBv2)  
DYNAMODBV2

Dessa forma faremos com que cada campo da mensagem se transforme em uma coluna no banco de dados. Por isso aquela formatação da mensagem em JSON era importante.

Se usássemos a primeira opção que é a `Insert a message into a DynamoDB table` teríamos que trabalhar um pouco mais no tratamento dos dados para organizar e analisar os mesmos.

Com a opção selecionada, clicamos em `Configure action`. A próxima página pede que selezionemos uma tabela, mas como não temos nenhuma tabela criada ainda, clicaremos no botão `Create a new resource`.

The DynamoDBv2 action allows you to write all or part of an MQTT message to a DynamoDB table. Each attribute in the payload is written to a separate column in the DynamoDB database. Messages processed by this action must be in the JSON format.

**\*Table name**

Choose a resource

Seremos redirecionados para o serviço do Amazon DynamoDB onde clicamos em `create table` para irmos diretamente ao ponto. A próxima página pede o nome da tabela e qual campo será utilizado como chave primária da tabela. Chamaremos essa tabela de `telemetria` e como chave primária configuraremos o campo `timestamp`. O restante das opção ficarão como está e clicaremos em `create`.

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

Table name\* `telemetry`

Primary key\* Partition key

`timestamp` String

Add sort key

## Table settings

Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.

- Use default settings
- No secondary indexes.
  - Provisioned capacity set to 5 reads and 5 writes.
  - Basic alarms with 80% upper threshold using SNS topic "dynamodb".

Alguns segundos depois, temos nosso tabela criada dentro do serviço do DynamoDB. É importante deixar claro que dentro da AWS, temos vários serviços como o de IoT e o DynamoDB, por que com o tempo, ficará ainda mais claro por que sempre precisamos integrar os serviços. Nós sempre precisaremos configurar qual serviço pode se conectar com qual e com que permissão. Tudo isso por questões de segurança.

The screenshot shows the AWS DynamoDB console. On the left, there's a sidebar with 'DynamoDB' selected, followed by 'Dashboard', 'Tables' (which is highlighted), 'Reserved capacity', 'DAX', 'Dashboard', 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main area shows a table named 'telemetry'. A search bar at the top says 'Filter by table name' with 'telemetry' typed in. Below it, a dropdown menu shows 'Name' selected. Under the table, there's a blue dot next to 'telemetry'. To the right, the 'telemetry' table details are shown. The 'Capacity' tab is active. Under 'Stream details', it says 'Stream enabled: No', 'View type: -', and 'Latest stream ARN: -'. There's a 'Manage Stream' button.

Note que há várias abas no lado direito da página da tabela. As mais importantes no momento são `Items` e a `Capacity`. a primeira por que será nela que veremos os dados posteriormente e a segunda por causa de uma configuração básica simples que podemos fazer para não termos uma maior do que esperada caso o projeto cresça.

Na aba `Capacity` vamos ajustar os campos `read capacity units` e `write capacity units` na sessão `Provisioned Capacity` para o valor 1, em vez do padrão 5. Estes campos provisionam o limite máximo de unidades de leitura e escrita na tabela. Como estamos apenas testando, deixaremos com 1 para que não gere custos altos. Note que há um custo estimado logo abaixo desses campos.

The screenshot shows the 'Capacity' tab of the DynamoDB table configuration. It displays 'Read capacity units' and 'Write capacity units' both set to 1. Below this, an 'Estimated cost' of '\$0.59 / month' is shown with a link to a 'Capacity calculator'. Under the 'Auto Scaling' section, there's a checkbox for 'Read capacity' which is checked. At the bottom, there are 'Save' and 'Cancel' buttons.

Clicamos em `save` e depois voltamos para a conexão dos serviços. Página que estávamos antes de clicar no botão `create resource` para criar a tabela. Agora precisamos criar uma `role`, que é a forma de como o serviço de IoT se conectará com o serviço do DynamoDB. Este é um processo muito mais simples do que o primeiro que fizemos.

Para criar a `role` basta clicar no botão `create a new role`. Será pedido um nome, usamos o nome `iot-dinamodb`, depois clicamos em `create a new role` novamente. Como último passo precisamos deixar selecionada a tabela e a `role` que acabamos de criar nos dois selecionadores ainda na mesma página.

The DynamoDBv2 action allows you to write all or part of an MQTT message to a DynamoDB table. Each attribute in the payload is written to a separate column in the DynamoDB database. Messages processed by this action must be in the JSON format.

\*Table name


Create a new resource


---

Choose or create a role to grant AWS IoT access to the DynamoDB resource to perform this action.

\*IAM role name


Update role
Create a new role

Caso o nome da tabela ou da role não apareça, clique no ícone de atualizar do lado do campo de seleção. E quando selecionar a role lembre-se de clicar no botão update role para confirmar a seleção da mesma. Uma mensagem de sucesso deverá ser exibida e com isso finalizamos essa parte, para encerrar de fato, basta clicar no botão create rule no final da página.

Seremos direcionados para a listagem de rules dentro do menu Act . Já podemos testar se tudo está funcionando. Lembre-se primeiro de ir até a opção Test do serviço AWS IoT e assinar o tópico telemetria/temperatura e depois disso, executar o script start.sh .

Com o script sendo executado, verificamos se na página Test os dados estão sendo impressos corretamente e depois no serviço DynamoDB, na tabela telemetria verificamos na aba Items os dados estão sendo armazenados de fato. E estão!

	timestamp	sensor	temperatura
<input type="checkbox"/>	2017-10-17 09:58:07.678655	rpi-sensor01	42.774
<input type="checkbox"/>	2017-10-17 09:57:47.588630	rpi-sensor01	43.312