

# AI & CHATBOT

Aula 11 – Armazenamento de Conhecimento

Prof. Henrique Ferreira

Prof. Miguel Bozer

Prof. Guilherme Aldeia

Prof. Michel Fornaciali

Prof. Daniel Petrini

FIAP  
GRADUAÇÃO

# Bases de Dados

Visão geral sobre armazenamento de informação

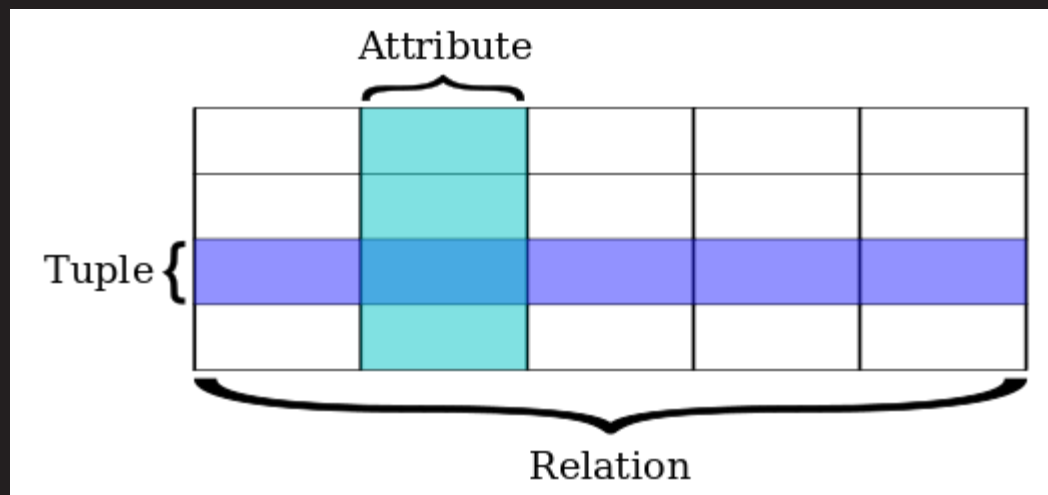
# Bases de dados



- A informação pode ser armazenada de diversas formas; Em geral temos dois tipos de bases de dados:
  - Relacional (SQL): dados são armazenados em tabelas;
  - Não relacional (NoSQL): dados podem ser armazenados em diferentes formatos;
- Além disso os dados podem estar armazenados de maneira:
  - Estruturada: tabelas
  - Semiestruturada: HTML, XML, JSON
  - Não estruturada: texto, imagens, áudio;

# Bases de dados Relacional

- Na base de dados relacional os dados estão em **tabelas** (chamadas de relação) onde as **colunas são atributos** e cada **linha é constituída de uma tupla de dados**. (obs: tupla é o nome geral para dupla, tripla, etc)



- É comum implementar usando SQL e algum programa de banco de dados;
- Diagramas UML são usados para descrever a relação e função das tabelas;

# Bases de dados Não Relacional

- Existem várias formas de criar uma base de dados não relacional, por exemplo usando grafos, objetos ou documentos (arquivos);
- Em alguns lugares o termo não relacional é colocado análogo ao termo semiestruturado;
- Aqui vamos focar no estudo de BD orientados a documentos. Exemplos são XML, JSON, YAML e BSON;
- A ideia por trás é armazenar os dados em arquivos com uma estrutura pré-determinada.

# Bases de dados Não Relacional

JSON	Raw Data	Headers
Save	Copy	Collapse All Expand All Filter JSON
▼ intents:		
▼ 0:		
intent:	"agradecimento"	
▼ examples:		
▼ 0:		
text:	"Agradeço o serviço"	
▼ 1:		
text:	"Esse bot tá show, obrigado"	
▶ 2:	{...}	
▶ 3:	{...}	
▶ 4:	{...}	
▶ 5:	{...}	
▶ 6:	{...}	
▶ description:	"Captura se o usuário est...adecer por alguma coisa"	
▶ 1:	{...}	
▶ 2:	{...}	
▶ 3:	{...}	
▶ 4:	{...}	
▶ entities:	{...}	
▼ metadata:		
▶ api_version:	{...}	
▼ dialog_nodes:		
▼ 0:		
type:	"standard"	
title:	"Bem-vindo"	

Exemplo de um arquivo JSON e de uma base de dados com vários arquivos JSON.

Nome	Tipo	Tamanho
bot_falandoll.json	Arquivo JSON	4 KB
bot_falandolll.json	Arquivo JSON	6 KB
bot_falandollV.json	Arquivo JSON	5 KB
exemplo_proc_imagem.json	Arquivo JSON	6 KB
exemplo_translator.json	Arquivo JSON	5 KB
fluxo_com_contexto.json	Arquivo JSON	4 KB
fluxo_http.json	Arquivo JSON	2 KB
fluxo_sem_contexto.json	Arquivo JSON	3 KB
skill-Ajuda_aula02.json	Arquivo JSON	5 KB
skill-Ajuda_aula03.json	Arquivo JSON	12 KB
skill-Ajuda_aula04.json	Arquivo JSON	23 KB
telegram_nlu_reddit.json	Arquivo JSON	8 KB
tradutor_audio_texto.json	Arquivo JSON	7 KB








# Criando uma base de dados

Usando o Cloudbant da IBM para criar uma base dados

# Cloudant

- No final da aula 5 nós instanciamos Serviços em Nuvem para roda rodar o Node-RED na nuvem da IBM.
- Um desses serviços era o Cloudant:

Resumo de recurso		Visualizar tudo
11		
Recursos		
Apps do Cloud Foundry	✓	1
Serviços do Cloud Foundry		1
Serviços	✓	7
Apps		1
Developer tools		1

^ Serviços (7)					
	Continuous Delivery	Default	Dallas	Continuous Delivery	✓ Ativo
	Language Translator-fiap	Default	Dallas	Language Translator	✓ Ativo
	Natural Language Understanding-fiap	Default	Dallas	Natural Language Understand...	✓ Ativo
	Speech to Text-56	Default	Dallas	Speech to Text	✓ Ativo
	Text to Speech-fiap	Default	Dallas	Text to Speech	✓ Ativo
	Watson Assistant-fiap	Default	Dallas	Watson Assistant	✓ Ativo
	nodered-fiap-1tds-cloudant-161749664...	Default	Dallas	Cloudant	✓ Ativo



# Cloudant

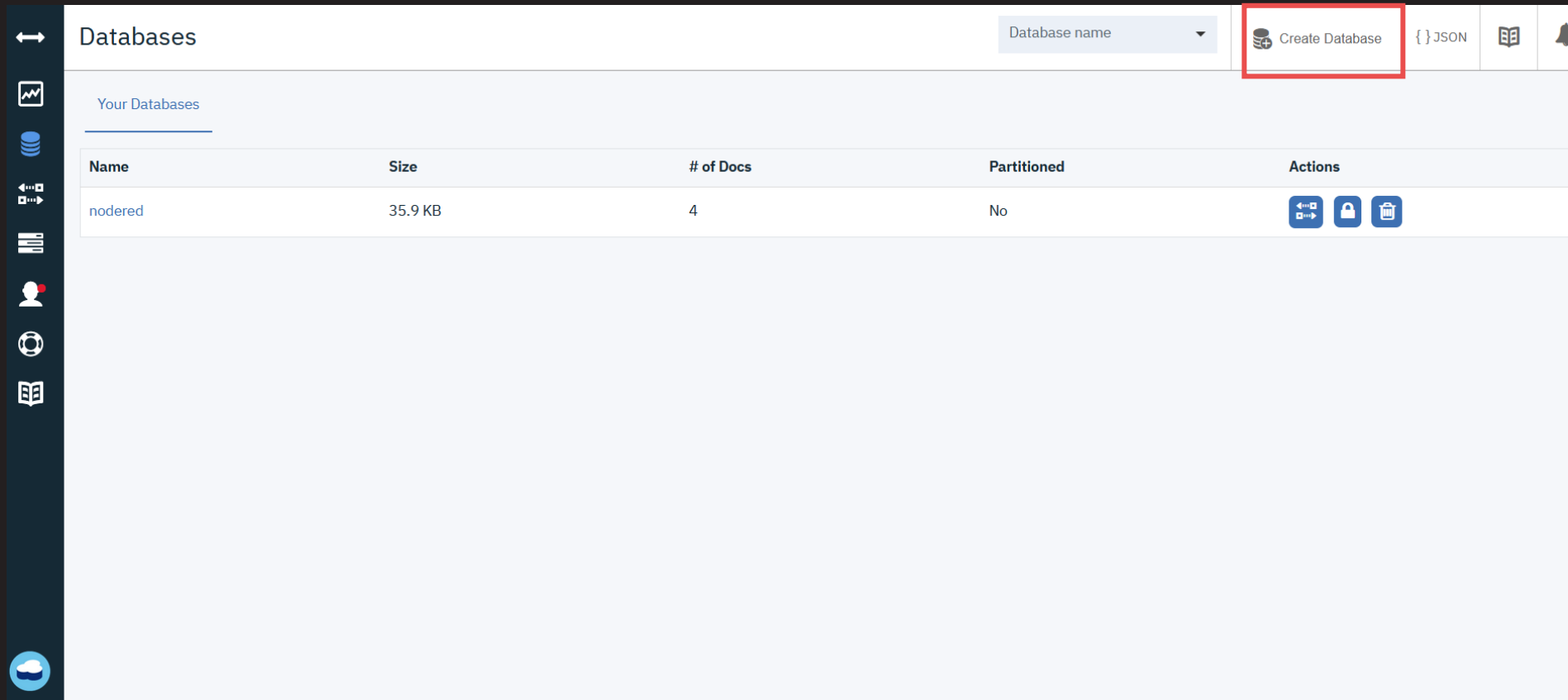
- Selecione o Cloudant e clique em Launch Dashboard;
- Se você tiver conseguido executar os passos da aula 5 corretamente, você deve ver uma tela como abaixo:

The screenshot shows the IBM Cloud console interface. At the top, there's a navigation bar with 'IBM Cloud', a search bar, and links for 'Catálogo', 'Documentos', 'Suporte', 'Gerenciar', and a user profile 'Henrique Ferreir...'. Below this, the breadcrumb 'Lista de recursos /' leads to the resource 'nodered-fiap-1tds-cloudant-1617496646088', which is marked as 'Ativo' (Active). A sidebar on the left lists 'Gerenciar', 'Credenciais de serviço', 'Plano', and 'Conexões'. The main content area has tabs for 'Overview', 'Dashboard', 'Capacity', and 'Docs'. The 'Overview' tab is selected, showing 'Deployment details' for the Cloudant instance. A red box highlights the 'Launch Dashboard' button in the top right corner of the details section.




Deployment details	
CRN	crn:v1:bluemix:public:cloudantnosqldb:us-south:a/d73947f2fe354fee92124a707fab34f0:63309a01-aea0-479e-8a55-d33e92626e93::
Location	Dallas
External Endpoint	<a href="https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudant.com">https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudant.com</a>
External Endpoint (preferred)	<a href="https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudantnosqldb.appdomain.cloud">https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudantnosqldb.appdomain.cloud</a>
Authentication methods	<a href="#">IBM Cloud IAM</a> and <a href="#">Cloudant credentials</a>
Activity Tracker event types ⓘ	Management <span>▼</span> <span>Save</span>
Disk encryption	Yes. Automatically generated disk encryption key.

# Cloudant

- O resultado será como abaixo. Perceba que temos apenas um Database relacionado ao nosso Serviço de Node-RED em nuvem. Vamos criar um para armazenar dados para o nosso bot:

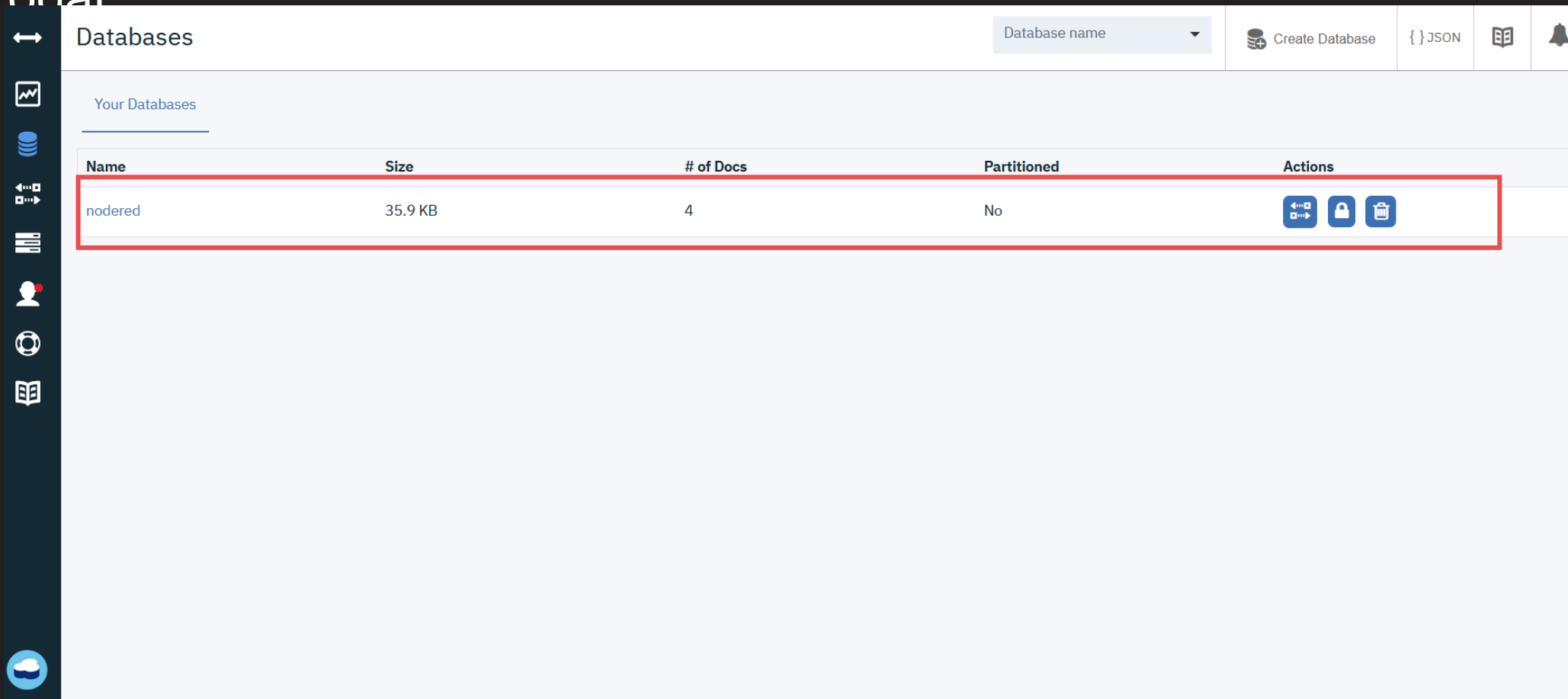


The screenshot shows the Cloudant Databases management interface. On the left is a dark sidebar with various icons. The main area has a header 'Databases' with a search bar and a 'Create Database' button highlighted with a red box. Below the header is a table titled 'Your Databases'.




Name	Size	# of Docs	Partitioned	Actions
nodered	35.9 KB	4	No	  

# Cloudant


- **ATENÇÃO:** não apague a base de dados do nodered (ou outro nome) que já estiver criada. Ela é que faz a sua aplicação do Node-RED na Nuvem rodar





The screenshot shows the Cloudant Databases management interface. At the top, there's a header with the title 'Databases', a 'Database name' dropdown, a 'Create Database' button, and links for JSON, documentation, and notifications. Below the header, a section titled 'Your Databases' contains a table listing existing databases. A red rectangle highlights the first row, which represents the 'nodered' database. The table has columns for Name, Size, # of Docs, Partitioned, and Actions. The 'nodered' database has a size of 35.9 KB, 4 documents, is not partitioned, and has three action icons: a refresh icon, a lock icon, and a delete icon.


Name	Size	# of Docs	Partitioned	Actions
nodered	35.9 KB	4	No	  

# Clouddant

 Create Database

 {} JSON





### Create Database

Database name

database\_bot


Partitioning

☐ Partitioned ☒ Non-partitioned

> What is a Partitioned Database?

Cancel Create

- Crie um databse particionado. Coloque um nome no seu database usando apenas letras minúsculas;
- O resultado deve ser:




database\_bot


No partition selected

Document ID

Options

{} JSON





Database created successfully

All Documents

Query

Permissions

Changes

Design Documents

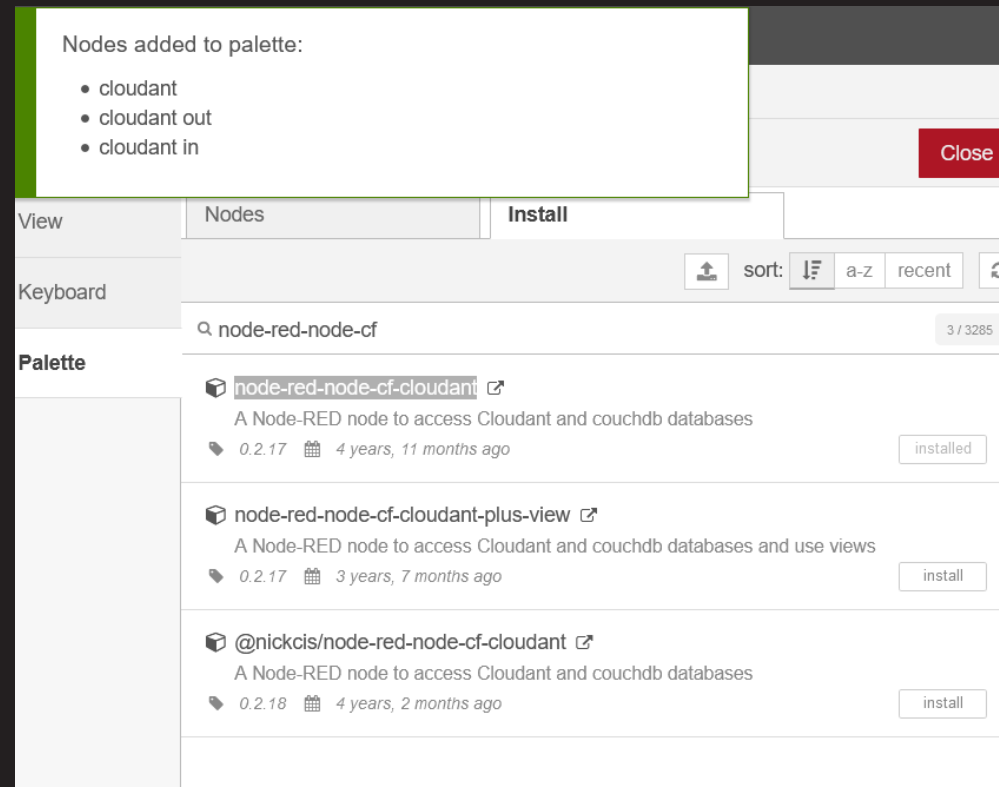
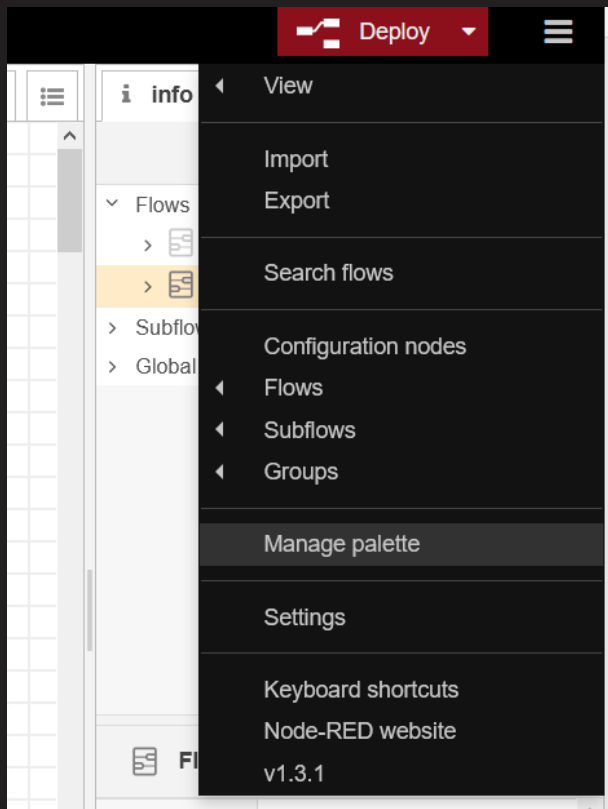
No Documents Found

# Integrando Base de Dados I

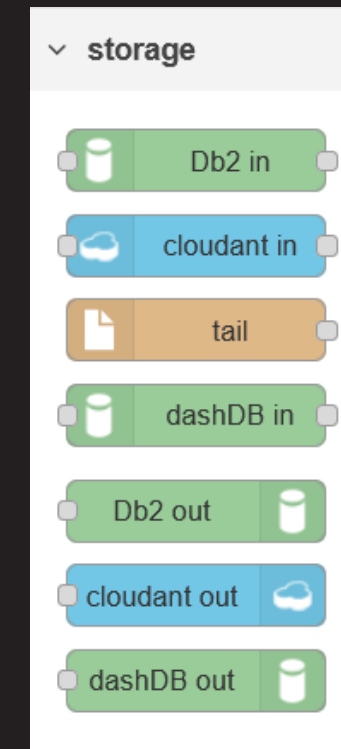
Exemplo de armazenamento de informação usando Node-RED

# Node-RED + Cloudant

- Vamos instalar os nós do Cloudant na nossa máquina (perceba que na Nuvem da IBM esses nós já estão instalados).
- Procure por **node-red-node-cf-cloudant**



Nuvem da IBM

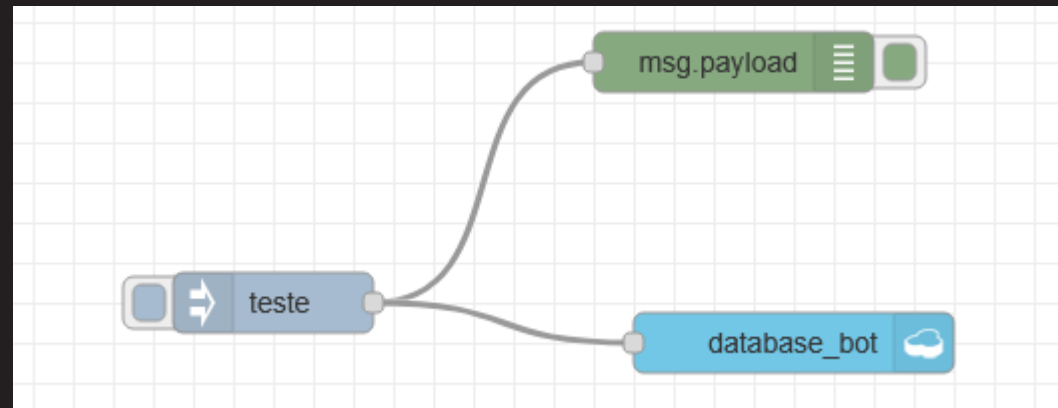


Máquina Local



# Node-RED + Cloudant: Armazenando dados

- Vamos criar um simples fluxo de armazenamento de informação;



- No nó de inject, configure para injetar uma string com um texto qualquer;
- No nó de **cloudant out**, é preciso colocar as credenciais. Atenção ao procedimento.

# Node-RED + Cloudant: Armazenando dados

- Se você estiver usando o Node-RED na nuvem da IBM, as credenciais do cloudant já estão embutidas. Basta colocar o nome do database (no caso, escolhemos database\_bot);
- Se você estiver usando o Node-RED local, então precisará pegar as informações.

Edit cloudant out node

Delete **Nuvem da IBM** Cancel Done

⚙ Properties

Service nodered-fiap-1tds-cloudant-1617496646088-6

Database database\_bot

Operation insert

☐ Only store msg.payload object?

Name Name

Edit cloudant out node > Edit cloudant node

Delete **Máquina Local** Cancel Update

⚙ Properties

Host https://76839436-580d-4faf-8add-0f8ccf639a43-l

Username apikey-v2-c1d486d9xbbousghjdn6ozi3hurooyim8

Password .....

Name nodered-fiap-1tds-cloudant-1617496646088



# Node-RED + Cloudant: Armazenando dados

- Na aba **Gerenciar** do serviço do Cloudant copie o link de acesso externo. Este será o valor do **host**.
- **Username** e **password** estão na aba **Credenciais de Serviço**. Clique na setinha para expandir. Não confunda use o host!

Location	Dallas
External Endpoint	<a href="https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudant.com">https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudant.com</a>
External Endpoint (preferred)	<a href="https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudantnosqldb.appdomain.cloud">https://76839436-580d-4faf-8add-0f8ccf639a43-bluemix.cloudantnosqldb.appdomain.cloud</a>
Authentication methods	IBM Cloud IAM and Cloudant credentials
Activity Tracker event types	Management <span>Save</span>
Disk encryption	Yes. Automatically generated disk encryption key.

Nome da chave	
c727a831-9084-4073-9e88	<pre>{   "apikey": "fCI9KhFXyq07   "host": "76839436-580d-   "iam_apikey_description   "iam_apikey_name": "c72   "iam_role_crn": "crn:v1   "iam_serviceid_crn": "c   f-4506-9f6c-05a07c66ac4a   "password": "66e24eadc   "port": 443,   "url": "https://apikey-   -0f8ccf639a43-bluemix_cl   "username": "apikey-v2 }</pre>

# Node-RED + Cloudant: Armazenando dados

- Ao clicar no bot de inject, você irá popular a a base de dados com um novo documento JSON.

database\_bot

Document ID

Options JSON

All Documents

Query

Permissions

Changes

Design Documents

Table Metadata JSON

Create Document

id	key	value
<input type="checkbox"/> d0cd547e20878c9e47060104311c9746	d0cd547e20878c9e47060104311c9746	{ "rev": "1-2d88c77c1d5287729e3eaac72720..." }

Lista de arquivos

database\_bot > d0cd547e20878c9e47060104311c9746

Save Changes Cancel

Dentro de um arquivo JSON

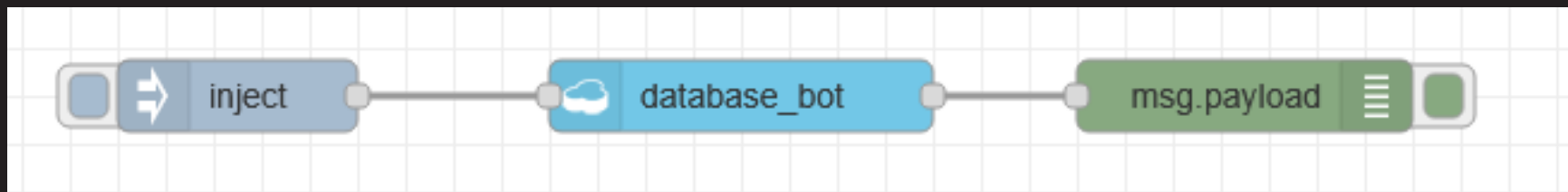
```
1 {  
2   "_id": "d0cd547e20878c9e47060104311c9746",  
3   "_rev": "1-2d88c77c1d5287729e3eaac727209541",  
4   "payload": "teste"  
5 }
```

# Integrando Base de Dados II

Exemplo de recuperação de informação usando Node-RED

# Node-RED + Cloudant: recuperando dados

- Agora vamos fazer um fluxo para recuperar dados salvos usando o nó **cloudant in**:



Vamos selecionar a opção **all documents**

**Properties**

Service: External cloudant or couchdb service

Server: nodered-fiap-1tds-cloudant-1617496646

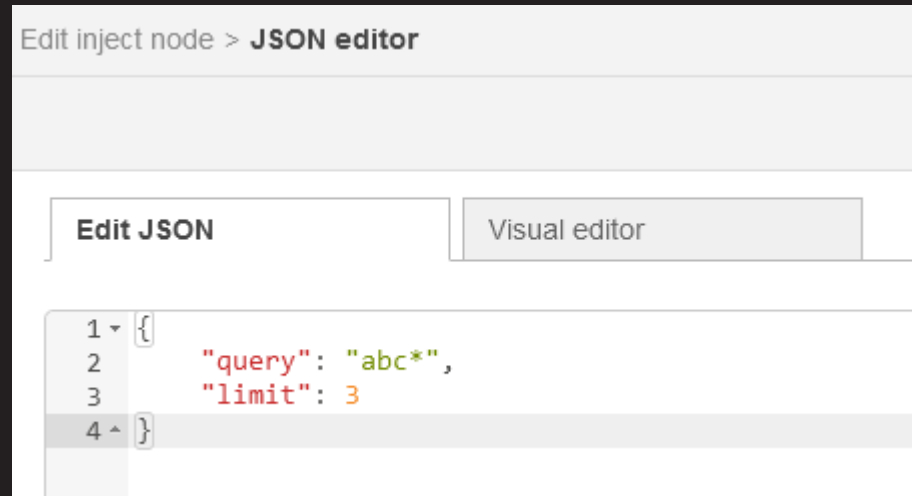
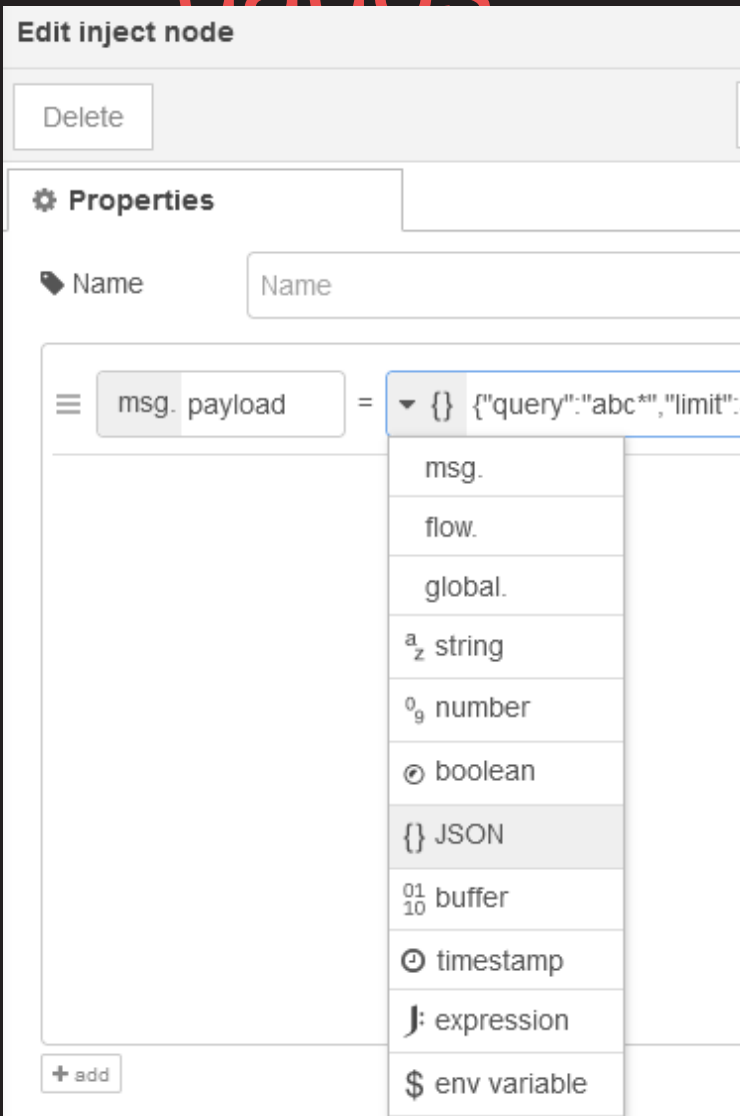
Database: database\_bot

Search by: all documents

Name:   
\_id  
search index  
**all documents**

# Node-RED + Cloudant: recuperando dados

- No nó de inject vamos configurar uma requisição. Escolha a opção JSON.
- Em seguida clique nos três pontinhos na frente do campo de preenchimento.
- No editor de JSON que abrirá, coloque: `{ "query": "abc*", "limit": 3 }`



# Node-RED + Cloudant: recuperando dados

- Dê um deploy e injete a mensagem. O resultado deverá ser algo como:

```
msg.payload : array[1]
  ▼ array[1]
    ▼ 0: object
      _id:
        "d0cd547e20878c9e47060104311c9746"
      _rev:
        "1-2d88c77c1d5287729e3eaac727209541"
      payload: "teste"
```

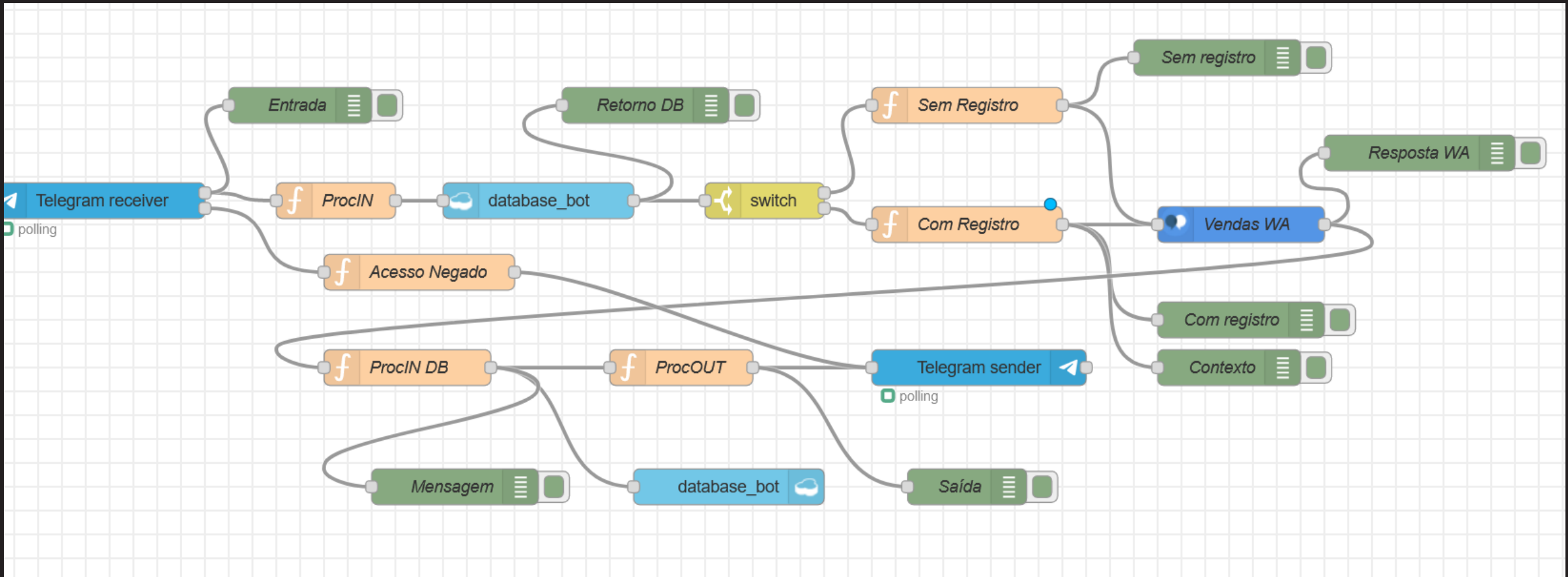
- Perceba que a busca por todos os documentos é útil, mas pode não ser muito prática. Entretanto se você já conhecesse o `_id` do documento de antemão, ficaria mais fácil recuperação a informação só dele.

# Integrando Base de Dados III

Salvando e recuperando informação em um bot

# Aumentando a memória do Bot Vendas: Bot + DB

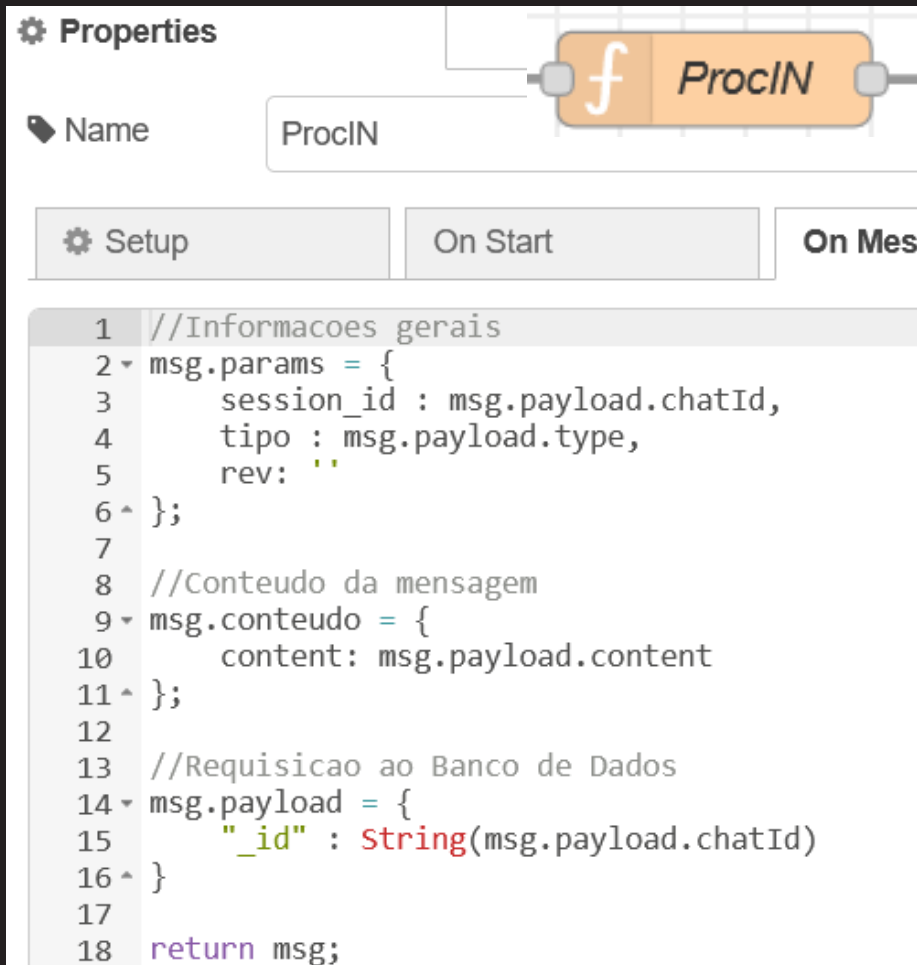
- Vamos fazer o seguinte fluxo:






# Bot + DB

- Dentro dos nós de function:



```
1 //Informacoes gerais
2 msg.params = {
3   session_id : msg.payload.chatId,
4   tipo : msg.payload.type,
5   rev: ''
6 };
7
8 //Conteudo da mensagem
9 msg.conteudo = {
10   content: msg.payload.content
11 };
12
13 //Requisicao ao Banco de Dados
14 msg.payload = {
15   "_id" : String(msg.payload.chatId)
16 }
17
18 return msg;
```



```
1 msg.params.rev = msg.payload._rev;
2 msg.additional_context={};
3
4 if(msg.payload.context){
5   if ("cep" in msg.payload.context){
6     msg.additional_context["cep"] = msg.payload.context.cep
7   }
8
9   if ("telefone" in msg.payload.context){
10    msg.additional_context['telefone'] = msg.payload.context.telefone
11  }
12
13   if ("email" in msg.payload.context){
14    msg.additional_context['email'] = msg.payload.context.email
15  }
16 }
17
18 msg.payload = msg.conteudo.content;
19 return msg;
```

# Bot + DB

- Dentro dos nós de function:

**Properties**

Name: Sem Registro

Setup On Start

```
1 msg.payload = msg.conteudo.content;
2 return msg;
```

**Properties**

Name: ProcOUT

Setup On Start

```
1 msg.payload={
2   chatId: msg.params.session_id,
3   type: msg.params.tipo,
4   content : msg.payload2
5 }
6 return msg;
```

**Properties**

Name: ProcIN DB

Setup On Start On Message On Stop

```
1 let imax = msg.payload.output.generic.length;
2 let resposta='';
3 for (let i = 0; i < imax; i++){
4   resposta = resposta+'\n'+msg.payload.output.generic[i].text;
5 }
6
7 msg.payload2 = resposta;
8
9 let u_cont = msg.payload.context.skills['main skill'].user_defined
10
11 //Primeira entrada no DB
12 if (msg.params.rev == ''){
13   msg.payload = {
14     "_id": String(msg.params.session_id),
15     "context": u_cont
16   }
17 //Demais entradas no DB
18 }else{
19   msg.payload = {
20     "_id": String(msg.params.session_id),
21     "context": u_cont,
22     "_rev":String(msg.params.rev )
23   }
24 }
25 return msg;
```

# Bot + DB

- Dentro dos outros nós relevantes:

**Edit cloudant out node**

Delete

**Properties**

Service: External cloudant or couchdb service

Server: nodered-fiap-1tds-cloudant-1617496646

Database: database\_bot

**Operation**: insert

☒ Only store msg.payload object?

Name: Name

**Edit cloudant in node**

Delete

**Properties**

Service: External cloudant or couchdb service

Server: nodered-fiap-1tds-cloudant-1617496646

Database: database\_bot

**Search by**: \_id

Name: Name

**Properties**

Name: Name

Property: msg.payload

== null → 1

!= null → 2

# Bot + DB

- Este fluxo é capaz de salvar as informações do **cadastro** (realizado na aula 4, Watson Assistant e Variáveis de Contexto. Estamos usando um nó de SLOT para gerar o cadastro que captura email, telefone e cep do usuário);
- **Atenção:** estamos usando as mesmas variáveis de contexto setadas no **Assistente de Vendas** que fizemos. Caso você esteja usando outro bot, deve se atentar para mudar as variáveis para ter o mesmo nome que as suas.

# Bot + DB

- A primeira vez que o fluxo é executado ele passa pelo nó “**Sem Registro**”. Assim que o cadastro é realizado, as variáveis de contexto setadas pelo WA são passada para o Cloudant; Na próxima interação com o bot, o fluxo irá passar pelo nó “**Com Registro**” que irá inserir as variáveis de contexto no WA através da variável `msg.additional_context`;
- Para dar nome ao arquivo JSON da nossa Base de Dados estamos usando o ID do Telegram. Lembre-se a **chave `_id` deve ser única** (em outras palavras, você deve ter um nome único para cada arquivo);
- O parâmetro **`_rev`** passado nas chamadas de edição da BD são flags para impedir que dois processos distintos reescrevam o mesmo arquivo. Idealmente você deve passar o `_rev` atual, mostrando que você sabe como o arquivo está antes de realizar uma alteração nele.

# Teste o fluxo e observe o resultado no BD

```
database_bot > 1297157419
```

☒ Save Changes Cancel

```
1 {
2   "_id": "1297157419",
3   "_rev": "32-95979c8d75bd0b37374ed723c1726e3c",
4   "context": {
5     "cep": "04041-004",
6     "telefone": "11 99874561",
7     "email": "professor@teste.com.br"
8   }
9 }
```

- Agora o seu bot é capaz de ter memória de longo prazo, podendo identificar dados e padrões de preferência de diversos clientes!

# Agora é com você!

Teste seu conhecimento do assunto realizando exercícios extras

# Exercício

1. Crie um fluxo no Watson Assistant com variáveis de contexto que armazenem informações sobre os gostos do usuário e que sejam usada para gerar uma oferta direcionada para ele sempre que o usuário escrever “bot me diga uma oferta que eu irei gostar”. Lembre-se de treinar as intenções e entidades necessárias para fazer isso e de armazenar as variáveis na Base de Dados do Cloudant.



# Próximos Passos

O que veremos na próxima aula

# Na próxima aula...

- Provas e finalização do semestre...



## **Copyright © 2022**

### **Slides do Prof. Henrique Ferreira - FIAP**

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).