

# Workshop Autonomous Database for Developers (JavaScript/NodeJS)

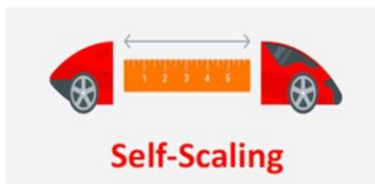
## Contents

|  |    |
|--|----|
| Introdução ao Autonomous Database .....                | 1  |
| Provisionamento do ambiente .....                      | 2  |
| Provisionamento de Banco de Dados .....                | 2  |
| Provisionamento de Rede .....                          | 5  |
| Provisionamento de Instância .....                     | 7  |
| Deploy da aplicação .....                              | 8  |
| Configuração da Instância .....                        | 8  |
| Configuração de Banco de Dados .....                   | 10 |
| Configuração de Backend .....                          | 12 |
| Configuração de Frontend .....                         | 12 |
| Testes .....   | 13 |
| Executando teste de conexão e revisando o código ..... | 13 |
| Executando teste de JSON e revisando o código .....    | 14 |
| Executando teste de SQL e revisando o código .....     | 16 |

## Introdução ao Autonomous Database

Oracle redefine o gerenciamento de dados com o primeiro banco de dados autônomo do mundo. O Autonomous Database Cloud elimina a complexidade, a probabilidade de erro humano e o gerenciamento manual, ajudando a garantir uma maior confiabilidade, segurança, além de uma maior eficiência operacional pelo menor custo.

## Autonomous Vision: Effortless, Limitless, Unbreakable Data Cloud



Para o desenvolvedor, o Autonomous Database simplifica o acesso ao banco e traduz, em a uma linguagem mais acessível, todas as vantagens de um banco de dados Oracle.

Neste Workshop veremos como é provisionar uma infraestrutura completa, onde serão executadas tarefas de criação de infraestrutura, de banco de dados, e o deploy de uma aplicação com duas formas de acesso ao Banco de Dados (JSON, e SQL), em uma instância na Oracle Cloud Infrastructure(OCI).


### Provisionamento do ambiente

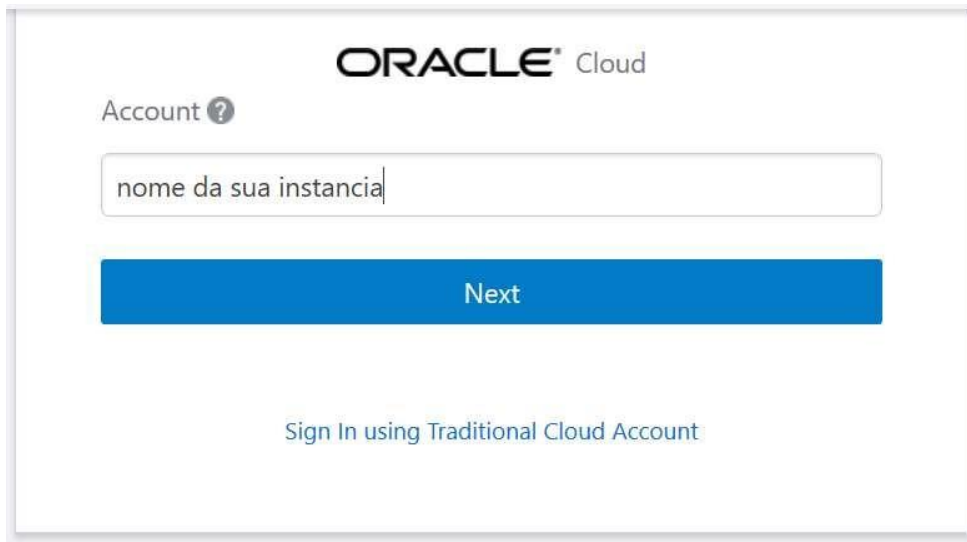
Neste passo vamos provisionar o ambiente necessário para o workshop.

Ferramentas como WINSCP e Putty serão utilizadas neste workshop, suas instalações não estão cobertas neste documento e suas funções podem ser substituídas por outro método a sua escolha.

### Provisionamento de Banco de Dados

O provisionamento do Autonomous Database pode ser feito de diversas formas, para esse workshop utilizaremos a UI da Oracle Cloud Infrastructure (OCI).

1. Acesse: <https://cloud.oracle.com>
2. No canto superior direito clique em  Sign In
3. Digite o nome da sua instância e clique em next



ORACLE® Cloud

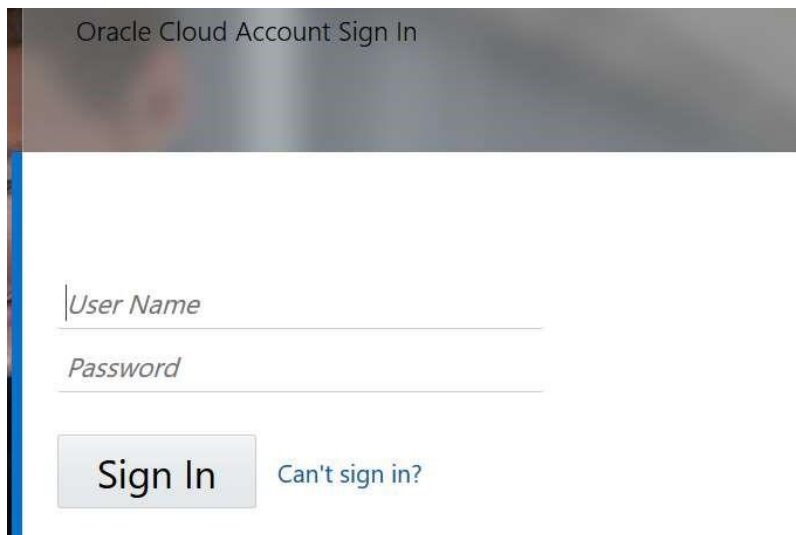
Account ?

nome da sua instancia

Next

[Sign In using Traditional Cloud Account](#)

4. Digite seu email, e sua senha e clique em sign in



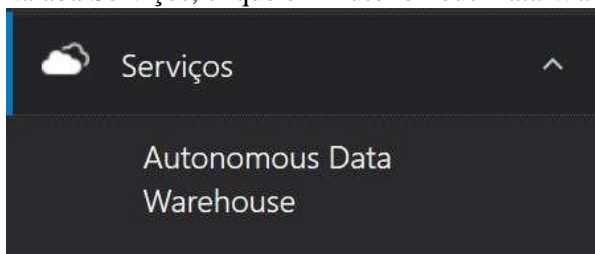
Oracle Cloud Account Sign In

User Name

Password

Sign In [Can't sign in?](#)

5. Clique no menu localizado no canto superior esquerdo
6. Na aba Serviços, clique em Autonomous Data Warehouse



7. No canto esquerdo inferior selecione, na sessão Compartment selecione o compartimento com final (root)
8. No canto esquerdo inferior, na sessão Filters selecione Transaction Processing

Filters

WORKLOAD TYPE

Transaction Processing

9. Clique em [Create Autonomous Database](#)
10. Preencha o formulário com as seguintes informações:

|                 |                          |
|-----------------|--------------------------|
| Compartment     | Mantenha o valor default |
| Display Name    | ATPFORDEV                |
| Database Name   | ATPFORDEV                |
| Workload Type   | Transaction Processing   |
| Deployment Type | Serverless               |
| CPU Core Count  | 1                        |
| Storage         | 1                        |
| Password        | Oracle123456             |
| License         | License Included         |

11. Clique em [Create Autonomous Database](#)
12. Aguarde o provisionamento (3min aprox.)



DB Connection

13. Clique em
14. Faça o download das credenciais de acesso (Wallet)

Download Client Credentials (Wallet)

To download your client credentials, click Download, and supply a password for the wallet.

Download

15. Insira a senha Oracle123456

Download Wallet [help](#) [close](#)


Database connections to your Autonomous Database use a secure connection. The wallet file will be required to configure your database clients and tools to access Autonomous Database.

Please create a password for this wallet. Some database clients will require that you provide both the wallet and password to connect to your database (other clients will auto-login using the wallet without a password).

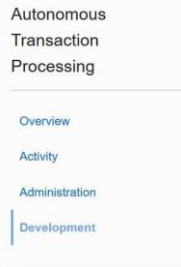
PASSWORD

CONFIRM PASSWORD

16. Salve o arquivo em uma pasta. (Não é necessário descompacta-lo).

17. Acesse a console do serviço clicando em  Service Console

18. No canto esquerdo, clique em Development



19. Clique em SQL Developer Web



20. Acesse utilizando o Usuario: admin a Senha: Oracle123456

21. Mantenha a aba do SQL Developer Web aberta, voltaremos a ela nos próximos passo.


Mais informações sobre o provisionamento e configuração da Base de dados podem ser encontradas em:

<https://docs.oracle.com/en/database/autonomous-database-cloud-services.html>

## Provisionamento de Rede

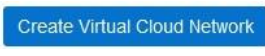
1. Retorne a console da Oracle Cloud Infrastructure (OCI). (Repita os passos de 1 a 6 do Provisionamento de Banco de Dados se caso necessário)

2. No canto superior esquerdo, acesse o menu 

3. Clique em Networking 

4. No canto esquerdo, na sessão Networking, selecione Virtual Cloud Network (VCN)



5. Clique em 

6. Preencha o formulário de criação da seguinte forma:

|             |                 |
|-------------|-----------------|
| Name        | <Qualquer nome> |
| Compartment | (root)          |

|                    |   |
|--------------------|---|
| Selecione a Opção: | Create virtual cloud network plus related resources |
|--------------------|---|

7. Clique em [Create Virtual Cloud Network](#)

8. Aguarde o provisionamento (< 1 min aprox.)



9. No canto inferior esquerdo, na sessão Resources, clique em Security Lists



10. No canto inferior central, clique em Default Security List <nome da sua vcn>

| Name  | State       |
|---|-------------|
| <a href="#">Default Security List for vcnWorkshop</a> | ● Available |

11. Clique em [Add Ingress Rules](#)

12. Preencha os campos da seguinte forma para abrir as portas necessárias para esse workshop.

Add Ingress Rules
cancel

Ingress Rule 1

Allows TCP traffic 80,1522,443,3000,8080

☐ STATELESS ⓘ

SOURCE TYPE

CIDR

SOURCE CIDR

0.0.0.0/0

Specified IP addresses: 0.0.0.0-255.255.255.255 (4,294,967,296 IP addresses)

IP PROTOCOL ⓘ

TCP

SOURCE PORT RANGE OPTIONAL ⓘ

All

Examples: 80, 20-22

DESTINATION PORT RANGE OPTIONAL ⓘ

80,1522,443,3000,8080

Examples: 80, 20-22

+ Additional Ingress Rule



Add Ingress Rules
Cancel

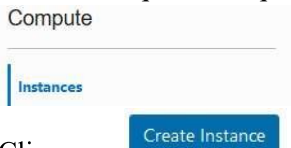
13. Clique em [Add Ingress Rules](#)

Mais informações sobre o provisionamento e configuração da Virtual Cloud Network podem ser encontradas em: <https://docs.cloud.oracle.com/iaas/Content/GSG/Tasks/creatingnetwork.htm>

## Provisionamento de Instância

1. Retorne a console da Oracle Cloud Infrastructure (OCI). (Repita os passos de 1 a 6 do Provisionamento de Banco de Dados se caso necessário)

2. No canto superior esquerdo, acesse o menu 
3. Clique em Compute 
4. No canto esquerdo clique em Instances



5. Clique em 
6. Preencha o formulário com as seguintes informações:

|                     |                      |
|---------------------|----------------------|
| Instance Name       | <nome a sua escolha> |
| Availability Zone   | AD 1, AD 2, ou AD 3  |
| Sistema Operacional | Oracle Linux 7.6     |
| Instance Type       | Virtual Machine      |
| Instance Shape      | VM.Standard.2.1      |

7. SSH Key:

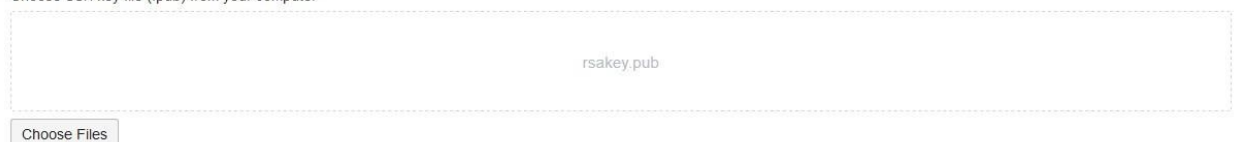
Para este exercício será necessário gerar um par de chave pública/privada para acessar a instância. Siga este passo-a-passo para gerar utilizando o Putty:

<https://docs.cloud.oracle.com/iaas/Content/GSG/Tasks/creatingkeys.htm#>

8. Realize o upload de sua chave pública .pub


☒ Choose SSH key file ☐ Paste SSH keys


Choose SSH key file (.pub) from your computer



9. Na sessão de Configure networking selecione a Virtual Cloud Network criada no exercício

## Provisionamento de Rede

10. Clique em Show Advanced Options 


11. Clique em 

12. Aguarde o provisionamento da instância. (<3min aprox.)



13. No canto inferior esquerdo, na sessão Resources clique em Console Connections

14. Clique em 

15. Copie/Faça o Upload da sua chave pública (.pub) e clique em  Create

Mais informações sobre o provisionamento e configuração de Instâncias podem ser encontradas em: <https://docs.cloud.oracle.com/iaas/Content/GSG/Tasks/creatingnetwork.htm>

## Deploy da aplicação

Neste passo vamos configurar o ambiente e fazer o deploy da aplicação.

### Configuração da Instância

1. Na console de sua instância copie a informação de IP Publico **Public IP Address:**
2. Acesse o PuTTY em seu desktop



3. Preencha o campo de Host Name com a seguinte informação:



Basic options for your PuTTY session

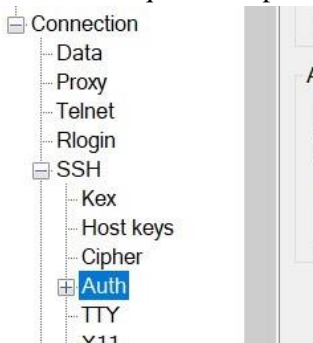
Specify the destination you want to connect to

| Host Name (or IP address)        | Port |
|----------------------------------|------|
| opc@<ip publico de sua instacia> | 22   |

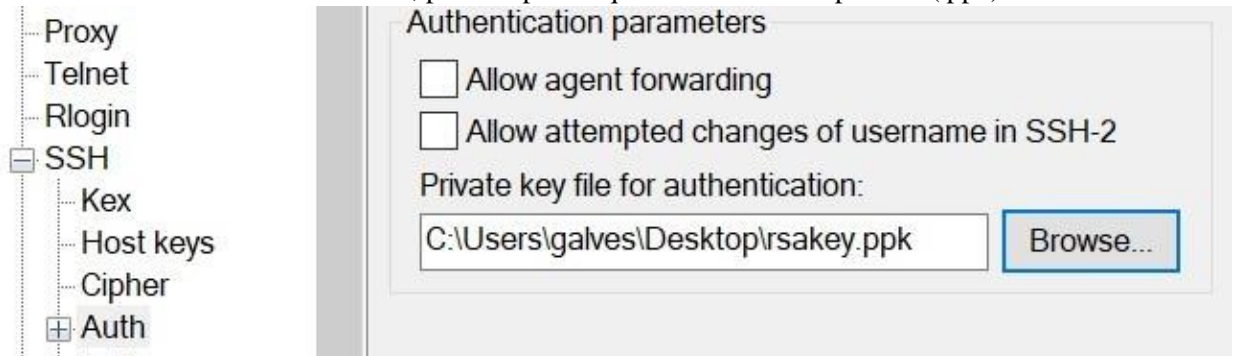
Connection type:

☐ Raw ☐ Telnet ☐ Rlogin ☒ SSH ☐ Serial

4. No canto esquerdo, expanda a sessão de Connection>SSH, e clique em Auth



5. Na sessão Authentication Parameters, procure pelo arquivo de sua chave privada (.ppk)

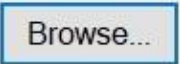


Authentication parameters

☐ Allow agent forwarding

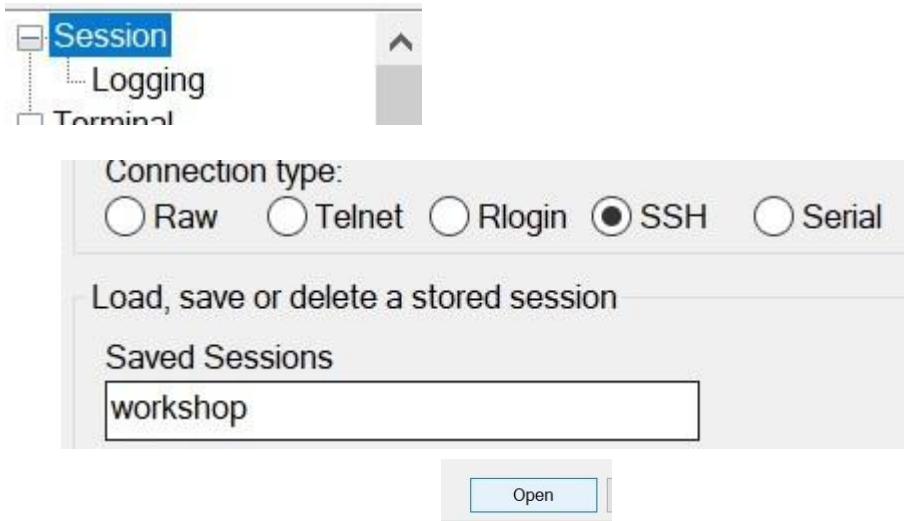
☐ Allow attempted changes of username in SSH-2

Private key file for authentication:

C:\Users\galves\Desktop\rsakey.ppk 



6. No canto esquerdo, clique em Session



7. PASSO OPCIONAL -  
Salve sua sessão, colocando um nome em Saved Sessions, e clicando em

Save

8. Clique em Open para iniciar a sessão  
9. Clique em Yes para adicionar a chave em suas chaves de segurança



10. Entre em modo root usando o comando  
`# sudo su`
11. Atualize os pacotes com o comando:  
`# yum update -y`  
Este passo pode levar alguns minutos
12. Instale o git:  
`# yum install -y git`
13. Crie um diretorio para desenvolvimento, neste caso usarei o diretorio /home/dev criado com o comando:  
`# mkdir /home/dev`

```
# cd /home/dev
```

14. Clone o repositório com a aplicação que utilizaremos neste workshop usando o comando:

```
# git clone https://github.com/gustavogaspar/workshops.git
```

15. Altere as permissões do arquivo /workshops/instancePrep.sh

```
# chmod 775 workshops/instancePrep.sh
```

16. Execute o script de configuração da instância:

```
# source workshops/instancePrep.sh
```

17. Acesse a pasta /workshop/lib

```
cd /workshop/lib
```

18. Instale o arquivo do Oracle Database Instant Client 19.x com o comando:

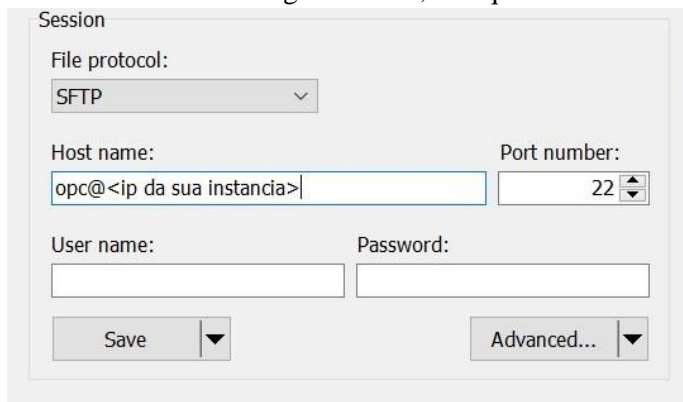
```
yum install -y oracle-instantclient19.3-basic-19.3.0.0.0-1.x86_64.rpm
```

## Configuração de Banco de Dados

1. Abra a ferramenta WINSCP em seu Desktop



2. Preencha conforme imagem abaixo, e clique em Advanced



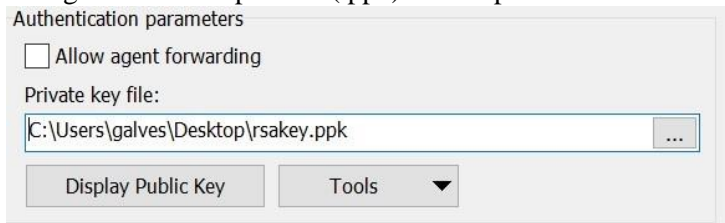
The image shows the 'Session' dialog box in WinSCP. It has the following fields and controls:

- File protocol:** A dropdown menu set to 'SFTP'.
- Host name:** A text box containing 'opc@<ip da sua instancia>'.
- Port number:** A spinner box set to '22'.
- User name:** An empty text box.
- Password:** An empty text box.
- Buttons:** 'Save' and 'Advanced...' (with a dropdown arrow).

3. Clique em SSH>Authentication



4. Carregue sua chave privada (.ppk) no campo Authentication Parameters



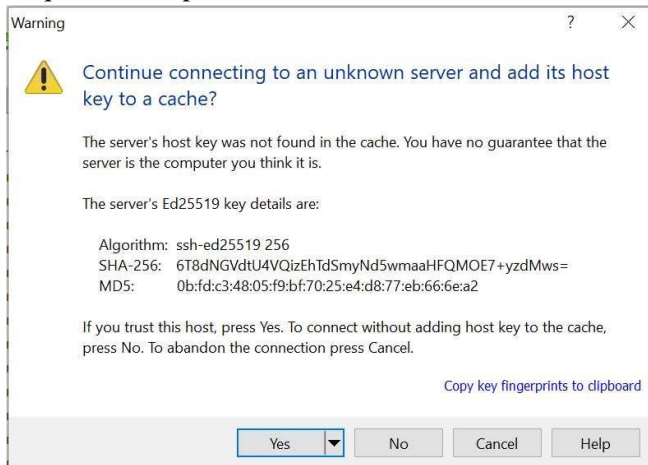
The image shows the 'Authentication parameters' dialog box. It has the following fields and controls:

- Allow agent forwarding:** An unchecked checkbox.
- Private key file:** A text box containing 'C:\Users\galves\Desktop\rsakey.ppk' and a browse button (...).
- Buttons:** 'Display Public Key' and 'Tools' (with a dropdown arrow).

5. Clique em OK

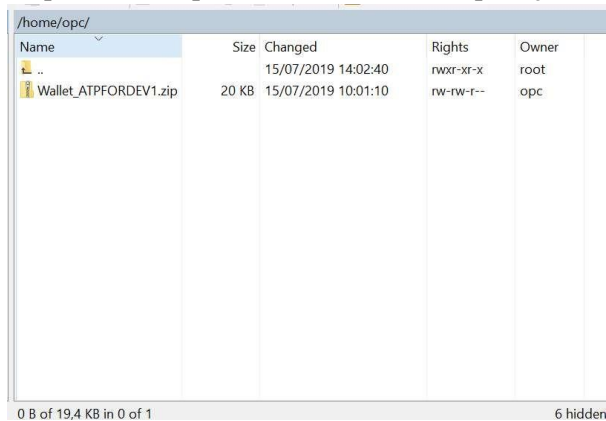
6. Clique em  Login

7. Clique em Yes para aceitar o certificado



8. Selecione o arquivo (.zip) de Wallet coletado nos passos 14 e 15 da sessão de Provisionamento de Banco de Dados.

9. Clique e arraste para o canto direito da aplicação



10. Feche a ferramenta WINSCP

11. Acesse a instância novamente utilizando a ferramenta PuTTY. Passos 1 a 7 da sessão de Configuração da Instância

12. Copie o arquivo de Wallet para a pasta de administrador criada no passo 13:

```
# cp /home/opc/<nome da Wallet> /usr/lib/oracle/19.3/client64/lib/network/admin/
```

13. Extraia os arquivos com os comando abaixo:

```
# cd /usr/lib/oracle/19.3/client64/lib/network/admin/
```

```
# unzip <nome da Wallet>
```

14. Inicie o serviço com os comandos:

```
# sh -c "echo /usr/lib/oracle/19.3/client64/lib > /etc/ld.so.conf.d/oracle-  
instantclient.conf"
```

```
# ldconfig
```

15. Retorne ao diretório de desenvolvimento criado no passo 14 da sessão de Configuração de Instância, neste caso:

```
# cd /home/dev
```

## Configuração de Backend

1. Acesse a instância novamente utilizando a ferramenta PuTTY. Passos 1 a 7 da sessão de Configuração da Instância
2. Acesse o modo root com o comando:  

```
# sudo su
```
3. Acesse o diretório de desenvolvimento criado no passo 14 da sessão de Configuração de Instância, neste caso:  

```
# cd /home/dev
```
4. Acesse a pasta backend dentro do projeto workshop e instale os pacotes necessários usando os comandos abaixo:  

```
# cd workshops/backend/  
# npm install
```
5. Altere o arquivo de dbconnect.js conforme a configuração do seu banco:  

```
# vim src/dbconnect.js
```
6. Ao abrir o arquivo pressione a tecla “i” do teclado, para entrar no modo de inserção, altere o código com as informações do seu Banco de dados criado na sessão Provisionamento do Banco de Dados. Exemplo abaixo: 

```
module.exports = {  user : "admin",  password :  
"<SENHA_DO_DB>",  connectionString : "<NOME DO SEU BANCO DE DADOS>_TP"  
}
```
7. Pressione a tecla “ESC” do teclado, e digite “:wq” para salvar e sair.
8. Inicie o servidor utilizando o comando  

```
# npm start &
```
9. Pressione ENTER para voltar ao terminal sem encerrar o processo.
10. Execute o comando abaixo para verificar se o processo ainda está ativo:  

```
# ps -a
```

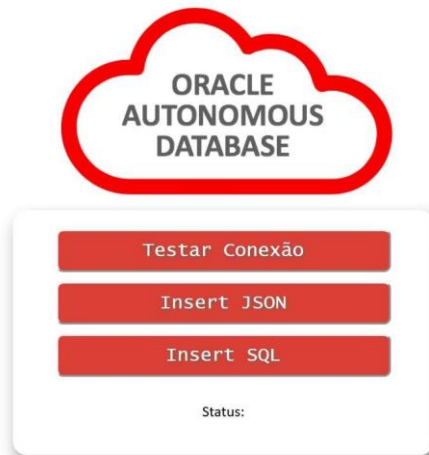
## Configuração de Frontend

1. Acesse a instância novamente utilizando a ferramenta PuTTY. Passos 1 a 7 da sessão de Configuração da Instância
2. Acesse o diretório de desenvolvimento criado no passo 14 da sessão de Configuração de Instância, neste caso:  

```
# cd /home/dev
```
3. Configure o arquivo /workshops/frontend/app.js conforme instruções abaixo:
  - a. Acesse o arquivo  

```
# vim /home/dev/workshops/frontend/app.js
```

- b. Configure o trecho abaixo o IP referente a sua Instância function btnCall(chamada) {  
document.getElementById('status').innerText = "PROCESSANDO...";  
axios.get('http://<IP\_PUBLICO>:3000/' + chamada)  
.then(response => { document.getElementById('status').innerText =  
response.data return console.log(response.data)  
})  
.catch(error => console.log(error)) }
4. Remova o conteúdo da pasta /usr/share/nginx/html/ e copie o conteúdo da pasta frontend para o mesmo destino:  
`# rm -rf /usr/share/nginx/html/*`  
`# cp /home/dev/workshops/frontend/* /usr/share/nginx/html/`
5. Abra o browser de sua preferência, coloque o IP de sua instância na barra de endereço, e verifique se a página abaixo é exibida:



## Testes

### Executando teste de conexão e revisando o código

O teste de conexão realiza um fluxo simplificado de abertura e encerramento de conexão com o banco quando a chamada GET é feita na raiz do endpoint.

Exemplo 1:

Utilizando curl para realizar a chamada: `curl`

`http://<ip da sua instancia>:3000/` Retorno:

```
[opc@atpdev2 ~]$ curl http://localhost:3000/  
A conexão foi iniciada e encerrada com sucesso
```

Exemplo 2:

Utilizando o frontend Web:



Observando o código:

```
1  const oracledb = require('oracledb')
2  const dbconnect = require('./dbconnect.js')
3
4  async function testConnection() {
5    let connection;
6    try {
7      connection = await oracledb.getConnection({
8        user: dbconnect.user,
9        password: dbconnect.password,
10       connectionString: dbconnect.connectionString
11     })
12   }
13   catch (err) {
14     console.log('Error in processing:\n', err);
15   }
16   finally {
17     const connectionStatus = "A conexão foi iniciada e" + await closeConnection(connection)
18     return connectionStatus
19   }
20 }
```

1. O código importa a biblioteca 'oracledb', esta biblioteca é responsável pelas funções relacionadas ao banco (ref. <https://oracle.github.io/node-oracledb/doc/api.html#getstarted>)
2. Os valores definidos no passo 6 da sessão Configuração do backend são utilizados para iniciar a conexão

### Executando teste de JSON e revisando o código

O teste de JSON realiza a criação de uma tabela, e a inserção de um registro simples no formato JSON quando a uma chamada GET é feita no endpoint /json.

Exemplo 1:

Utilizando curl para realizar a chamada: `curl`

`http://<ip da sua instancia/json`

```
[opc@atpdev2 ~]$ curl http://localhost:3000/json
A tabela carros já foi criada:

Infos do registro:
Nome: UNO

Marca: FIAT

Ano: 2003[opc@atpdev2 ~]$
```

Exemplo 2:

Utilizando o frontend Web:



Testar Conexão

Insert JSON

Insert SQL

Status: A tabela carros já foi criada:

Infos do registro:  
Nome: UNO

Marca: FIAT

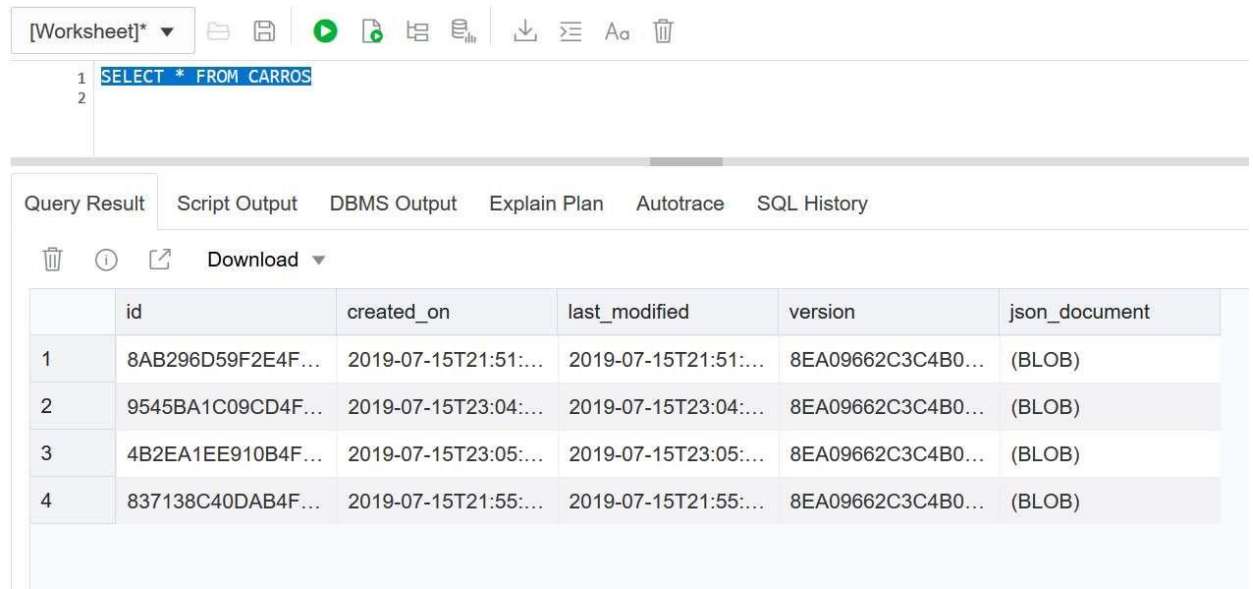
Ano: 2003

Observando o código:

1. Para o armazenamento em JSON utilizamos o Simple Oracle Document Access (SODA), um grupo de chamadas de API responsável por receber e transformar em query chamadas feitas no formato JSON. (ref. <https://docs.oracle.com/en/database/oracle/oracledatabase/18/adsdi/overview-soda.html>)

```
soda = connection.getSodaDatabase()
tabela = 'carros'
collection = await soda.createCollection(tabela)
```

2. No Exemplo acima, utilizamos o SODA para coletar as informações da Base de Dados, e criar uma Collection com o nome de carros. Para a base de dados a collection é interpretada como uma tabela, onde cada registro armazena um BLOB contendo o payload JSON. Exemplo abaixo:



The screenshot shows a database interface with a toolbar at the top containing icons for file operations, execution, and formatting. Below the toolbar, a SQL query is entered in a text area: `SELECT * FROM CARROS`. The query is numbered 1 and 2. Below the query area, there are tabs for 'Query Result', 'Script Output', 'DBMS Output', 'Explain Plan', 'Autotrace', and 'SQL History'. The 'Query Result' tab is active, displaying a table with 6 columns: 'id', 'id', 'created\_on', 'last\_modified', 'version', and 'json\_document'. The table contains 4 rows of data, each with a unique ID and a BLOB value for 'json\_document'.

|   | id                | created_on           | last_modified        | version           | json_document |
|---|-------------------|----------------------|----------------------|-------------------|---------------|
| 1 | 8AB296D59F2E4F... | 2019-07-15T21:51:... | 2019-07-15T21:51:... | 8EA09662C3C4B0... | (BLOB)        |
| 2 | 9545BA1C09CD4F... | 2019-07-15T23:04:... | 2019-07-15T23:04:... | 8EA09662C3C4B0... | (BLOB)        |
| 3 | 4B2EA1EE910B4F... | 2019-07-15T23:05:... | 2019-07-15T23:05:... | 8EA09662C3C4B0... | (BLOB)        |
| 4 | 837138C40DAB4F... | 2019-07-15T21:55:... | 2019-07-15T21:55:... | 8EA09662C3C4B0... | (BLOB)        |

3. O payload JSON é enviado por meio de uma função após a execução do metodo `.commit()`, conforme exemplo abaixo:

```
async function insertAndGetValuesJSON(coll, conn) {  
  let payload = '{"name": "UNO", "detalhe": { "marca": "FIAT", "ano": "2003" } }'  
  try {  
    let result = await coll.insertOneAndGet(JSON.parse(payload))  
    conn.commit()  
  }  
}
```

### Executando teste de SQL e revisando o código

O teste de SQL realiza a criação de uma tabela, e a inserção de um registro simples no formato SQL quando a uma chamada GET é feita no endpoint `/sql`.

Os comandos sql são realizados por meio do metodo `.execute` conforme exemplos abaixo:



```
let createTable = await connection.execute(  
  `CREATE TABLE series(  
    nome VARCHAR2(50),  
    ano VARCHAR2(4),  
    temporadas NUMBER  
  )`,  
);  
console.log("Create Table: ", createTable)
```

```
let select = await connection.execute(  
  "SELECT * FROM SERIES"  
)  
console.log("Select: ", select.rows)
```

Este é o fim do workshop!

Para mais exemplos de uso acesse: <https://github.com/oracle/node-oracledb/tree/master/examples>