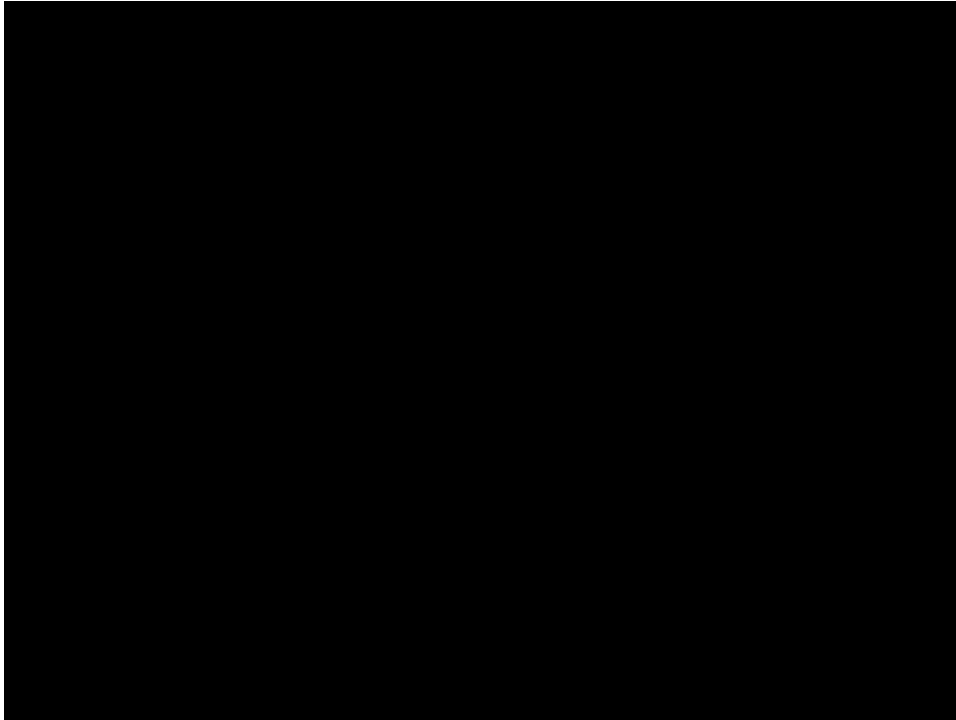

Aplicação com BD

— —

Flávio Souza

Introdução a Banco de Dados - Prelúdio



Link do Vídeo:

https://drive.google.com/file/d/1VzVajP_4lY4AkYIkNx62VH1Tf39IsFo7/view?usp=sharing

Introdução a Banco de Dados - Crie um usuário do BD

The screenshot shows the MySQL Workbench interface. The 'Administration - Users and Privileges' window is open, displaying the 'Details for account teste@localhost' tab. The 'User Accounts' table lists the following users:

User	From Host
mysql.infoschema	localhost
mysql.session	localhost
mysql.sys	localhost
root	localhost
teste	localhost

The 'Details for account teste@localhost' tab shows the following configuration:

- Login Name: teste
- Authentication Type: Standard
- Limit to Hosts Matching: localhost
- Password: [masked]
- Confirm Password: [masked]
- Buttons: Add Account, Delete, Refresh, Revert, Apply, Expire Password

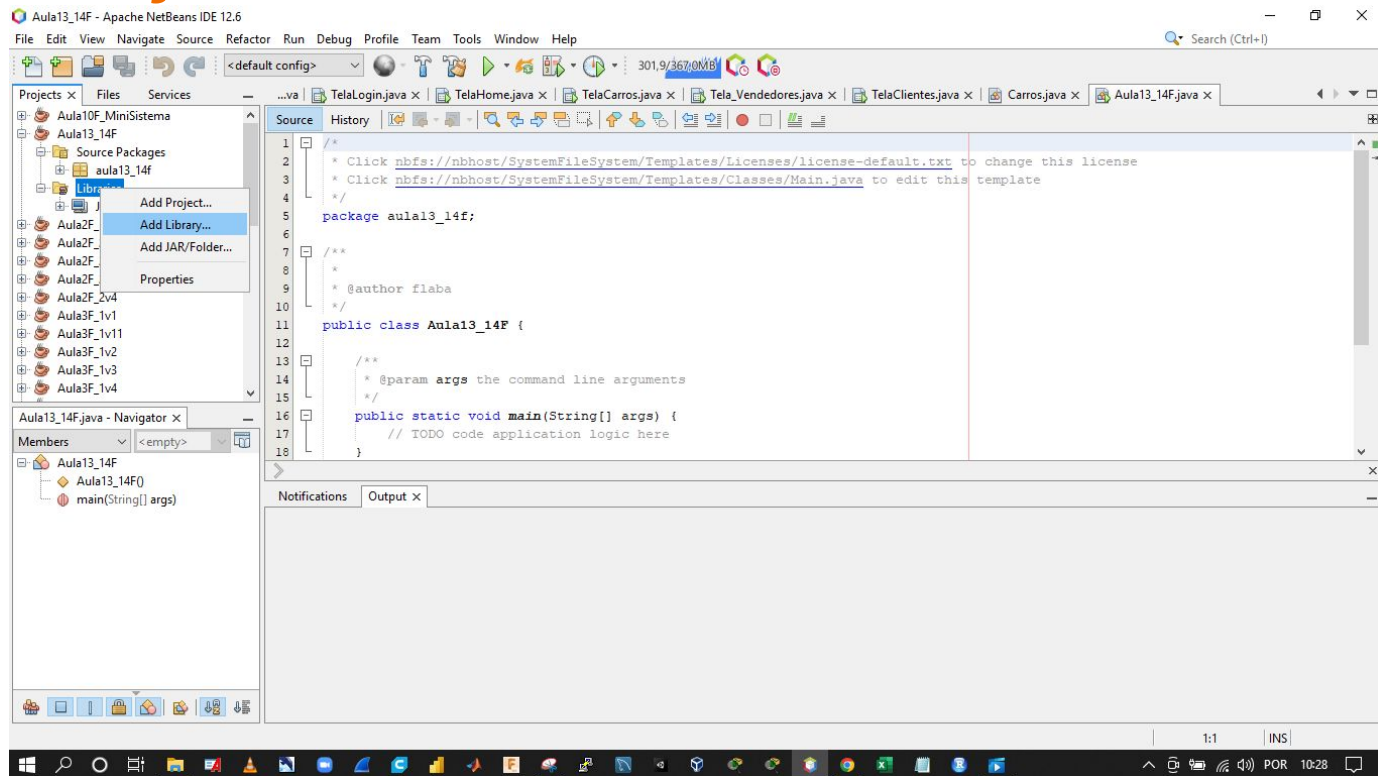
The 'Output' window at the bottom shows the following message:

#	Time	Action
1	10:45:17	use batman

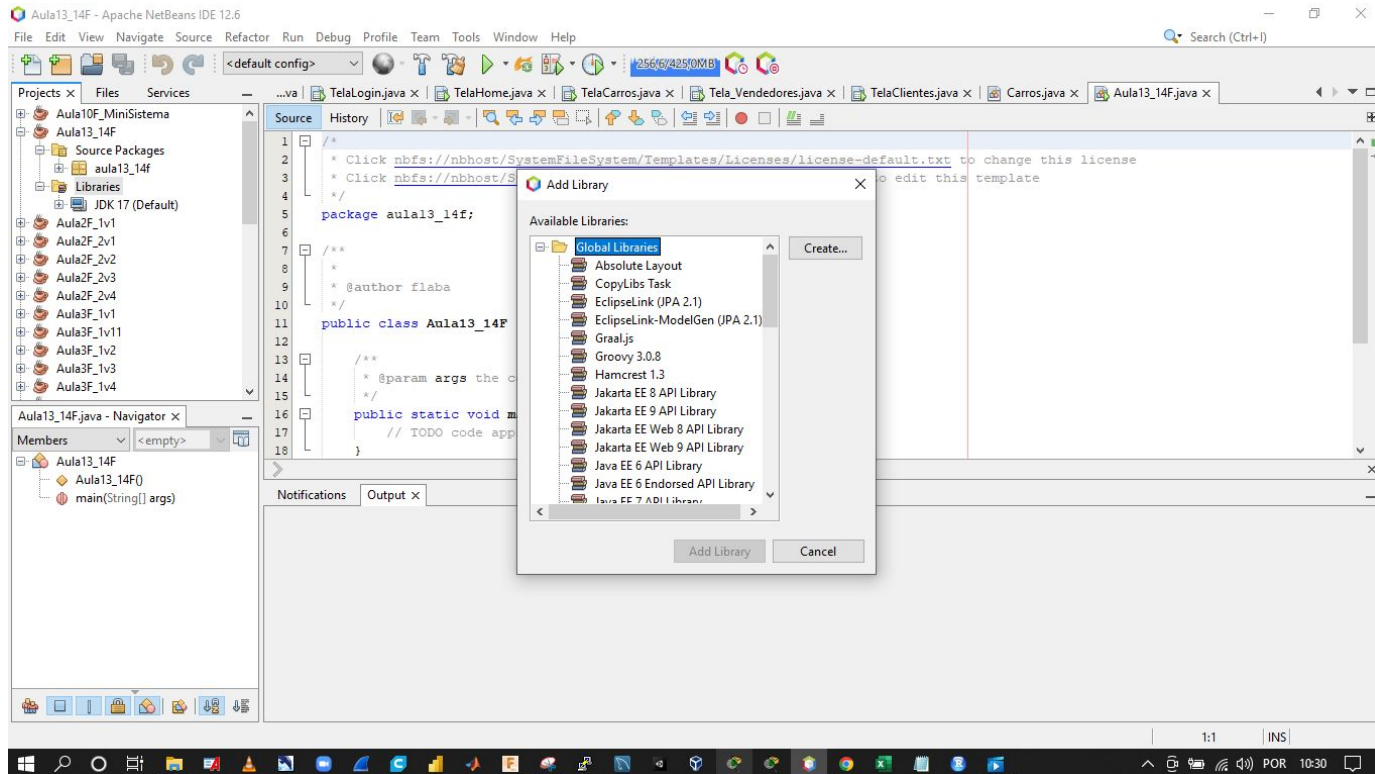
0 row(s) affected
Duration / Fetch: 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

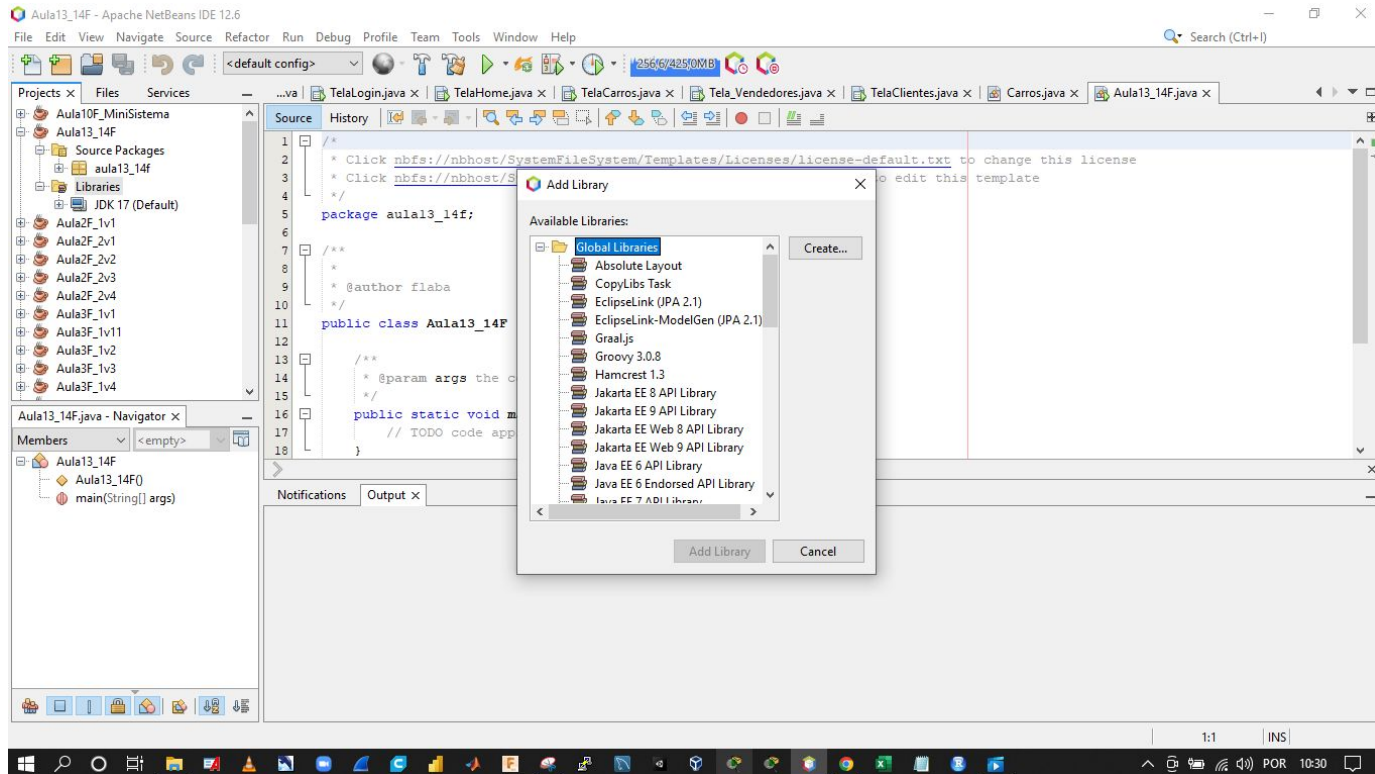
Introdução a Banco de Dados



Introdução a Banco de Dados



Introdução a Banco de Dados



Pode ser que
não tenha a
MYSQL

Introdução a Banco de Dados - PODE SER NECESSÁRIO

<https://downloads.mysql.com/archives/c-j/>

MySQL Product Archives

MySQL Connector/J (Archived Versions)

 Please note that these are old versions. New releases will have recent bug fixes and features!

To download the latest release of MySQL Connector/J, please visit [MySQL Downloads](#).

Product Version:

Operating System:

Platform Independent (Architecture Independent), Compressed TAR Archive

(mysql-connector-java-8.0.28.tar.gz)

Dec 15, 2021

4.0M

[Download](#)

MD5: 5f21fdde306a6d643e4c6e8a2ec4b5bd | [Signature](#)

Platform Independent (Architecture Independent), ZIP Archive

(mysql-connector-java-8.0.28.zip)

Dec 15, 2021

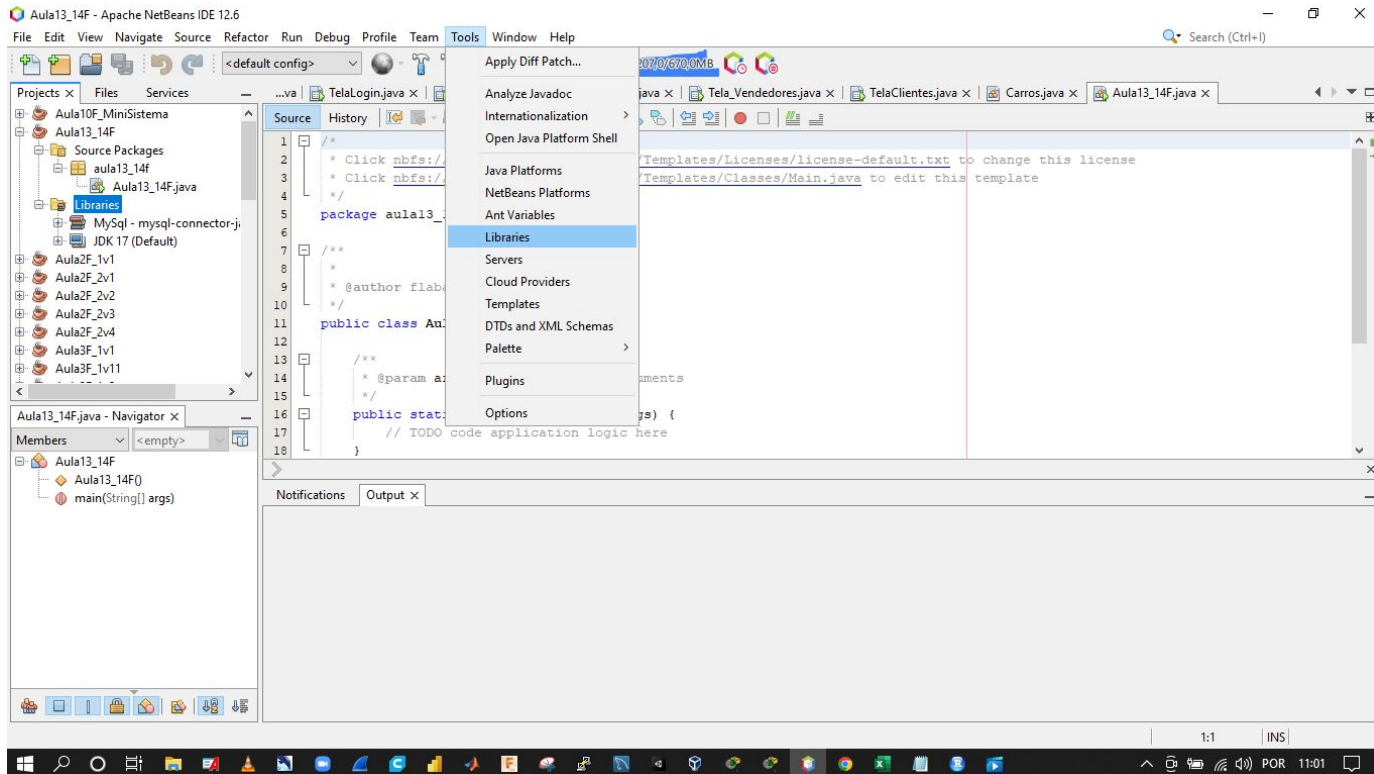
4.8M

[Download](#)

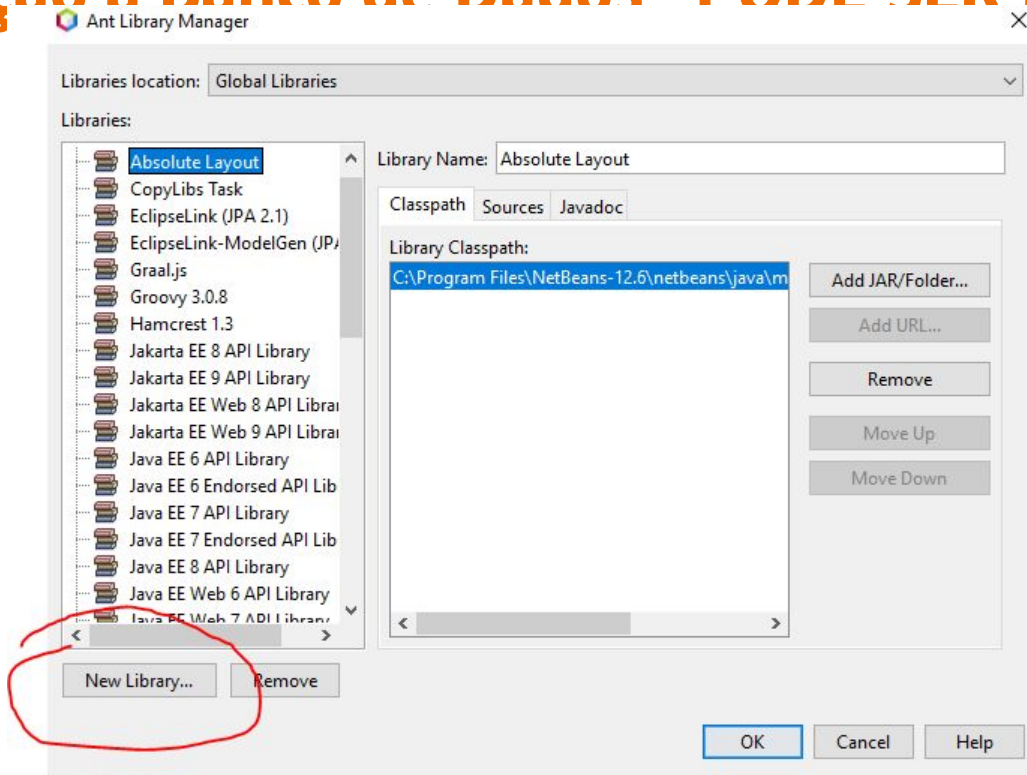
MD5: d19db89de75c46b71c4257badfa1e22b | [Signature](#)

 We suggest that you use the [MD5 checksums](#) and [GnuPG signatures](#) to verify the integrity of the packages you download.

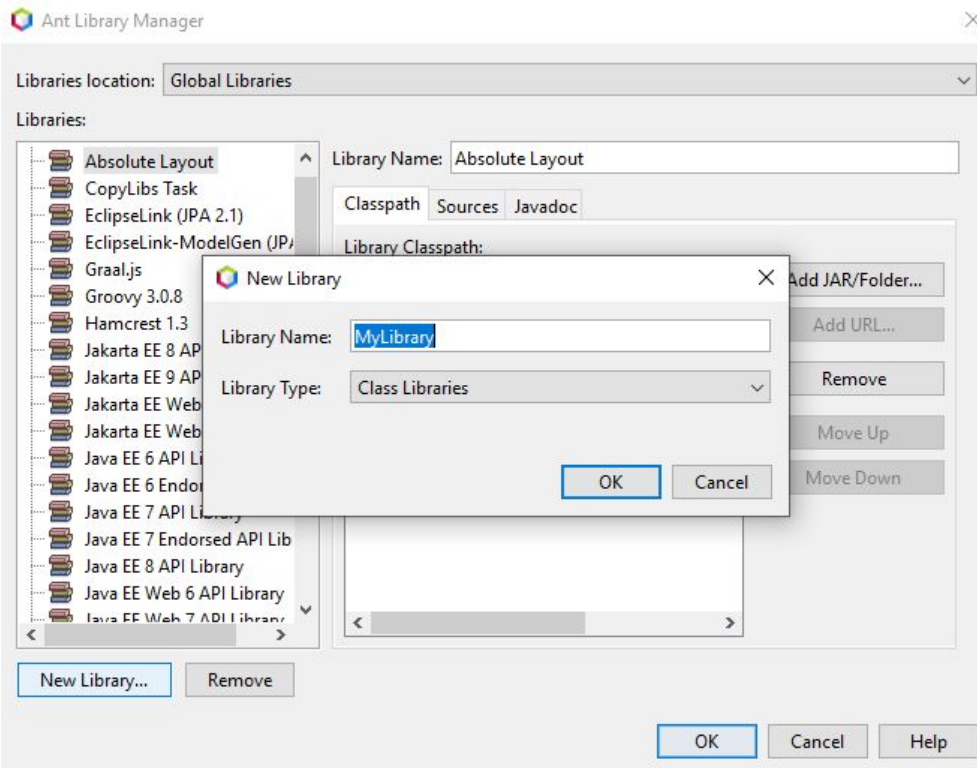
Introdução a Banco de Dados - PODE SER NECESSÁRIO



Introdução a Banco de Dados - PODE SER NECESSÁRIO

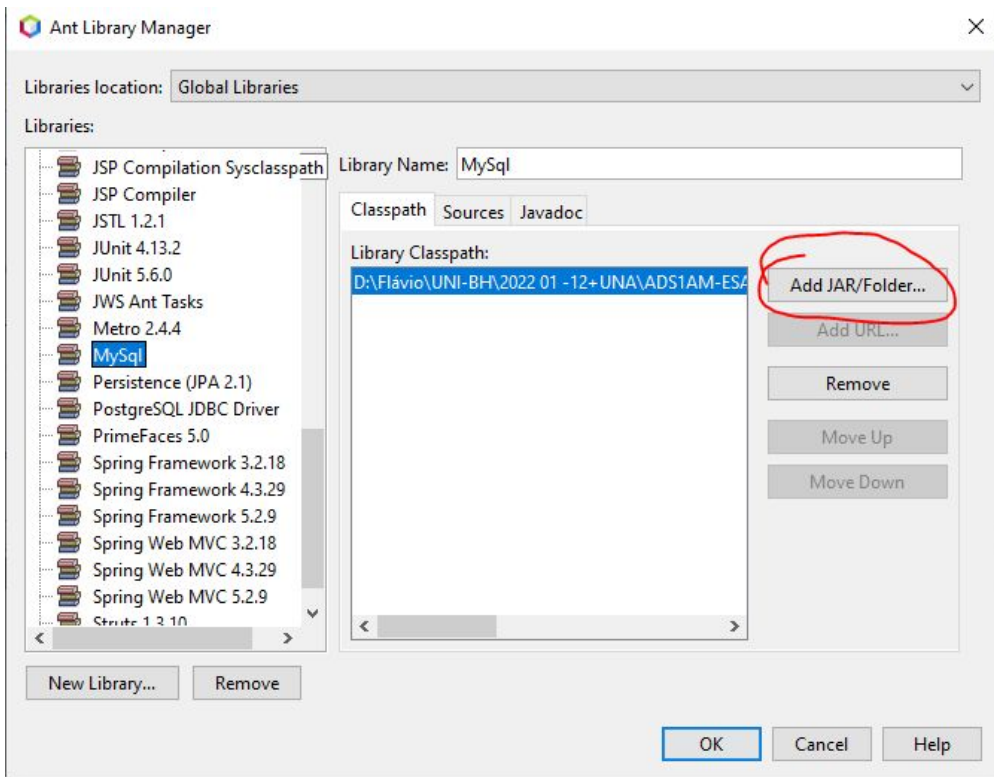


Introdução a Banco de Dados - PODE SER NECESSÁRIO



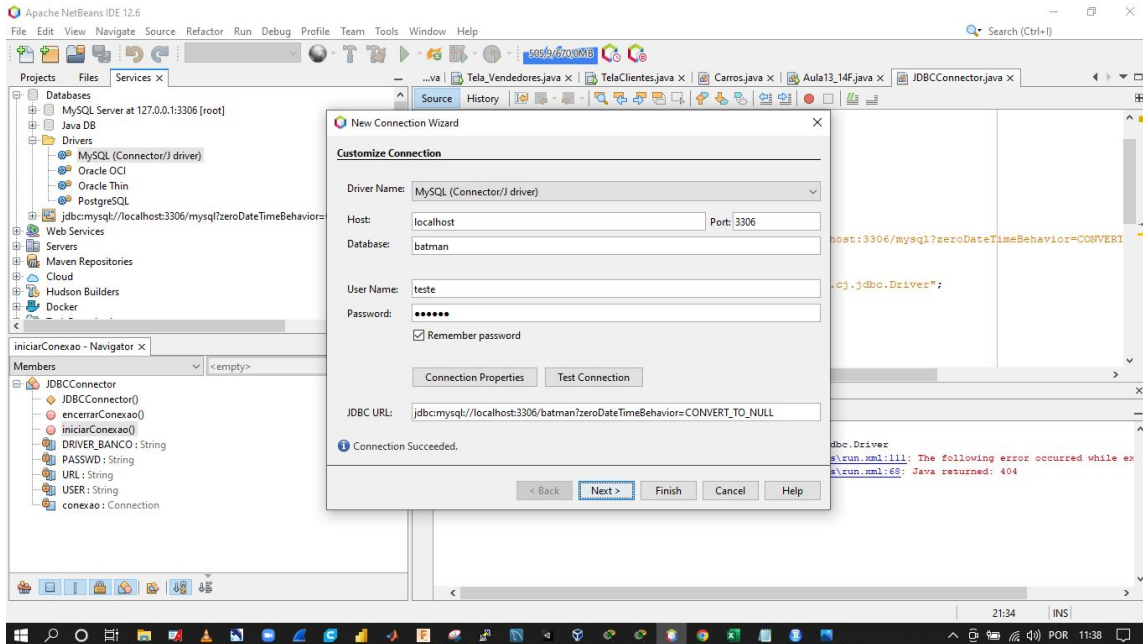
Coloque MySql

Introdução a Banco de Dados - PODE SER NECESSÁRIO



Adicione a pasta zip
(Windows) que você fez
o download

Introdução a Banco de Dados - PODE SER NECESSÁRIO

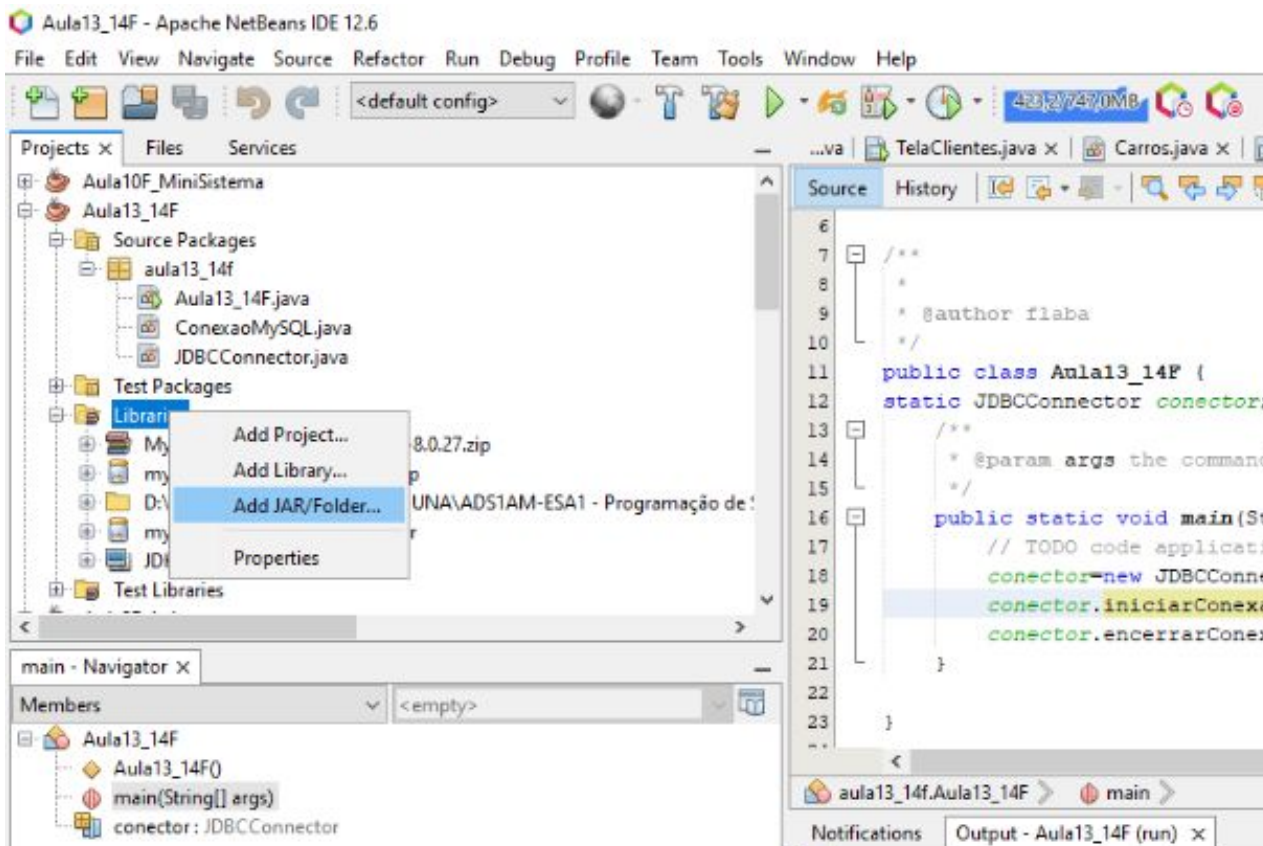


Adicione a pasta zip
(Windows) que você fez
o download

Introdução a Banco de Dados - PODE SER NECESSÁRIO

CRIE O PROJETO!

Introdução a Banco de Dados - PODE SER NECESSÁRIO



Lembre-se de adicionar a biblioteca .JAR

Introdução a Banco de Dados - PODE SER NECESSÁRIO

CODE TIME!!!!

Na classe MAIN:

```
7  /**
8   *
9   * @author flaba
10  */
11  public class Aula13_14F {
12      static JDBCConnector conector;
13      /**
14       * @param args the command line arguments
15       */
16      public static void main(String[] args) {
17          // TODO code application logic here
18          conector=new JDBCConnector();
19          conector.iniciarConexao();
20          conector.encerrarConexao();
21      }
22  }
23
24
```


Crie a Classe de conexão (é uma classe comum, só terá o papel de realizar as conexões com o Banco de dados)

Nome dela - JDBCConnector

```
5 package aula13_14f;
6 import java.sql.Connection;
7 import java.sql.DriverManager;
8 import java.sql.SQLException;
9 /**
10  *
11  * @author flaba
12  */
13 public class JDBCConnector {
14     private static final String URL="jdbc:mysql://localhost:3306/batman";
15     private static final String USER="teste";
16     private static final String PASSWD="123456";
17     private static final String DRIVER_BANCO="com.mysql.cj.jdbc.Driver";
18     private Connection conexao;
19     public void iniciarConexao() {
20         try{
21             Class.forName(DRIVER_BANCO);
22             conexao=DriverManager.getConnection(URL,USER,PASSWD);
23             System.out.println("[OK] CONEXÃO ESTABELECIDADA");
24         }catch(ClassNotFoundException | SQLException ex){
25             System.out.println("[ERRO] na CONEXÃO "+ex);
26             System.exit(404);
27         }
28     }
29     public void encerrarConexao() {
30         try{
31             conexao.close();
32             System.out.println("[ERRO] CONEXÃO ENCERRADA");
33         }catch(SQLException ex){
34             System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);
35             System.exit(404);
36         }
37     }
38 }
```

Manipulando Dados No Sql via Java

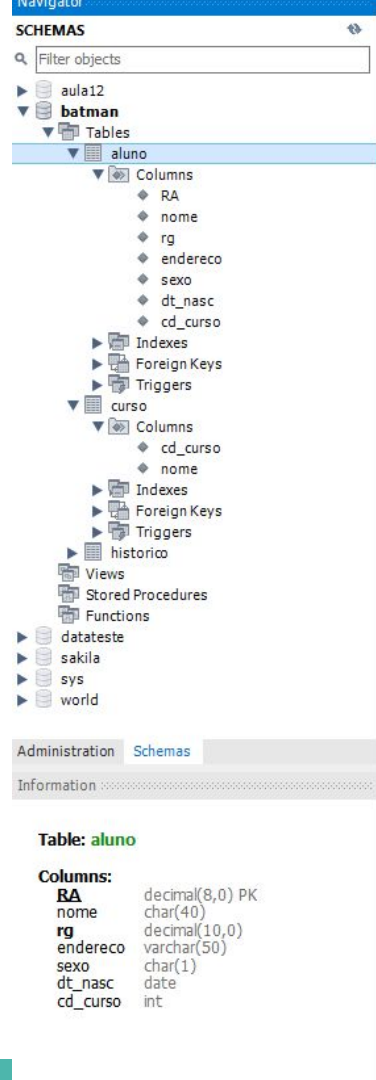
Crie o Banco de Dados no SQL e Depois você pode manipular seus dados

Aqui: foram criadas as tabelas aluno e curso da seguinte forma:

```
CREATE TABLE `curso` (  
  `cd_curso` int NOT NULL,  
  `nome` char(40) NOT NULL,  
  PRIMARY KEY (`cd_curso`)  
) ENGINE=InnoDB DEFAULT  
CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

```
CREATE TABLE `aluno` (  
  `RA` decimal(8,0) NOT NULL,  
  `nome` char(40) NOT NULL,  
  `rg` decimal(10,0) NOT NULL,  
  `endereco` varchar(50) DEFAULT NULL,  
  `sexo` char(1) DEFAULT NULL,  
  `dt_nasc` date DEFAULT NULL,  
  `cd_curso` int DEFAULT NULL,  
  PRIMARY KEY (`RA`),  
  UNIQUE KEY `rg` (`rg`),  
  CONSTRAINT `aluno_chk_1` CHECK (((`sexo` =  
_utf8mb4'M') or (`sexo` = _utf8mb4'F') or (`sexo`  
= _utf8mb4'O'))))  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci;
```

Crie as duas tabelas e insira dois registros para verificar



Antes de mais nada, na classe **MAIN**:

```
7  import java.sql.ResultSet;
8  import java.sql.SQLException;
9
```

Antes de mais nada, na classe de conexão que criamos (**JDBCConnector**):

```
6  import java.sql.Connection;
7  import java.sql.DriverManager;
8  import java.sql.PreparedStatement;
9  import java.sql.ResultSet;
10 import java.sql.SQLException;
```

INSERIR DADOS:

3 passos PRESTE ATENÇÃO!!!!

Para INSERIR DADOS:

1 - NA CLASSE JDBCConnector que foi criada:

A - Insira o código em Azul **DEPOIS** do método de encerrar conexão.

B - não esqueça de que tem uma **ÚLTIMA CHAVE** da Classe (**NÃO A PERCA**)

C - coloque no inicio da classe:
import java.sql.PreparedStatement;

```
28         System.exit(404);
29     }
30 }
31 public void encerrarConexao() {
32     try{
33         conexao.close();
34         System.out.println("[ERRO] CONEXÃO ENCERRADA");
35     }catch(SQLException ex){
36         System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);
37         System.exit(404);
38     }
39 }
40 public void inserir(String sql) throws SQLException {
41     try{
42         PreparedStatement statement = conexao.prepareStatement(sql);
43         int rowsInserted = statement.executeUpdate();
44         if (rowsInserted > 0) {
45             System.out.println("A new data was inserted successfully!");
46         }
47     }catch(SQLException ex){
48         System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);
49         System.exit(404);
50     }
51 }
52 }
53 }
```

```
5 package aula13_14f;
6
7 import java.sql.SQLException;
8
9 /**
10  *
11  * @author flaba
12  */
13 public class Aula13_14F {
14     static JDBCConnector conector;
15
16     /**
17      * @param args the command line arguments
18      */
19     public static void main(String[] args) throws SQLException {
20         // TODO code application logic here
21         conector=new JDBCConnector();
22         conector.iniciarConexao();
23         conector.inserir("insert into aluno values('13246','Dick Greyson','12365','Mansão Wayne','M','1995-05-14','2')");
24         conector.encerrarConexao();
25     }
26
27 }
28
```

2 - Main onde vc USA O CÓDIGO:

A - Insira o código em Azul **DEPOIS** do método de encerrar conexão.

B - não esqueça de que tem uma ÚLTIMA CHAVE da Classe (NÃO A PERCA)

3 passos PRESTE ATENÇÃO!!!!

Para INSERIR DADOS:

3 - Execute e confira se está inserida no seu BD (vá no MySQL para isso:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view with 'aula12' expanded, containing 'batman', 'curso', and 'historico'. The 'batman' schema is selected. In the center, the 'Query 1' editor shows the SQL query: `SELECT * FROM batman.aluno;`. This query is circled in red. Below the query editor, the 'Result Grid' shows the results of the query. The results are as follows:

RA	nome	rg	endereco	sexo	dt_nasc	cd_curso
13246	Dick Greyson	12365	Mansão W...	M	1995-05-14	2
1111111	Antonio Souza	12345	Rua X	M	2022-02-23	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL

At the bottom, the 'Action Output' pane shows a log of executed queries and their results:

#	Time	Action	Message
✓ 1	10:23:37	SELECT * FROM batman.aluno LIMIT 0, 1000	1 row(s) returned
✓ 2	10:24:01	SELECT * FROM batman.aluno LIMIT 0, 1000	1 row(s) returned
✗ 3	10:24:37	Apply changes to new_procedure	
! 4	10:28:32	Apply changes to batman	No changes detected
✓ 5	11:05:10	SELECT * FROM batman.aluno LIMIT 0, 1000	1 row(s) returned
✓ 6	11:10:07	select * from aluno LIMIT 0, 1000	2 row(s) returned
✓ 7	11:21:56	SELECT * FROM batman.aluno LIMIT 0, 1000	2 row(s) returned

Alterar DADOS:

3 passos PRESTE ATENÇÃO!!!!

Para ATUALIZAR DADOS:

1 - NA CLASSE JDBCConnector
que foi criada:

A - Insira o código em vermelho
DEPOIS do método de inserir.
B - não esqueça de que tem uma
ÚLTIMA CHAVE da Classe (NÃO
A PERCA)

```
39 -  
40 public void inserir(String sql) throws SQLException {  
41     try{  
42         PreparedStatement statement = conexao.prepareStatement(sql);  
43         int rowsInserted = statement.executeUpdate();  
44         if (rowsInserted > 0) {  
45             System.out.println("A new data was inserted successfully!");  
46         }  
47     }catch(SQLException ex){  
48         System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);  
49         System.exit(404);  
50     }  
51 }  
52  
53 public void atualizar(String sql) throws SQLException {  
54     try{  
55         PreparedStatement statement = conexao.prepareStatement(sql);  
56         int rowsInserted = statement.executeUpdate();  
57         if (rowsInserted > 0) {  
58             System.out.println("A new data was updated successfully!");  
59         }  
60     }catch(SQLException ex){  
61         System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);  
62         System.exit(404);  
63     }  
64 }  
65  
66 }  
67
```

```
18 public static void main(String[] args) throws SQLException {
19     // TODO code application logic here
20     conector=new JDBCConnector();
21     conector.iniciarConexao();
22     //conector.inserir("insert into aluno values('13246','Dick Greyson','12365','Mansão Wayne','M','1995-05-14','2')");
23
24     conector.atualizar("update aluno set nome='Jason Todd' where RA='13246'");
25
26     conector.encerrarConexao();
27 }
28
29 }
30
```

2 - Main onde vc USA O CÓDIGO:

- A - Olhe a seta **Azul** - comente a linha com o comando de inserir (senão vai inserir duas vezes ao executar)
- B - Insira o código em **VERMELHO** e rode

Navigator

SCHEMAS

Filter objects

aula12

batman

Tables

aluno

Columns

Indexes

PRIMARY

rg

Foreign Keys

Triggers

curso

historico

Views

Stored Procedures

Functions

datatest

sakila

sys

world

Administration Schemas

Information

Query 1

aluno x batman.curso

Limit to 1000 rows

1 • SELECT * FROM batman.aluno;

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

RA	nome	rg	endereço	sexo	dt_nasc	cd_curso
13246	Jason Todd	12365	Mansão W...	M	1995-05-14	2
1111111	Antonio Souza	12345	Rua X	M	2022-02-23	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL

aluno 3 x

Output

Action Output

#	Time	Action	Message
✓ 1	10:23:37	SELECT * FROM batman.aluno LIMIT 0, 1000	1 row(s) returned
✓ 2	10:24:01	SELECT * FROM batman.aluno LIMIT 0, 1000	1 row(s) returned

3 - Confira no BD:

Seleccionar DADOS:

```
public ResultSet selecionar(String sql) throws SQLException {  
    try{  
        java.sql.Statement statement = conexao.createStatement();  
        ResultSet result = statement.executeQuery(sql);  
        return result;  
  
    }catch(SQLException ex){  
        System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);  
        return null;  
    }  
}
```

1 - NA CLASSE JDBCConnector que foi criada:

A - Insira o código.

B - não esqueça de que tem uma ÚLTIMA CHAVE da Classe (NÃO A PERCA)

The screenshot shows an IDE with a project named 'Aula13_14F'. The 'Source Packages' folder contains 'aula13_14f' with files 'Aula13_14F.java' and 'JDBCCConnector.java'. The 'Test Packages' folder contains 'Libraries' and 'Test Libraries'. The 'Test Libraries' folder contains several test files: 'Aula2F_1v1', 'Aula2F_2v1', 'Aula2F_2v2', 'Aula2F_2v3', 'Aula2F_2v4', 'Aula3F_1v1', and 'Aula3F_1v11'. The 'main - Navigator X' tab shows the 'main' method of 'Aula13_14F' with parameters 'String[] args'. The 'Members' tab shows the 'main' method of 'Aula13_14F' with parameters 'String[] args' and the 'conector' object of type 'JDBCCConnector'.

```
15 static JDBCCConnector conector;  
16 /**  
17  * @param args the command line arguments  
18  */  
19 public static void main(String[] args) throws SQLException {  
20     // TODO code application logic here  
21     conector=new JDBCCConnector();  
22     conector.iniciarConexao();  
23     //conector.inserir("insert into aluno values('13246','Dick Greyson','12365','Mansão Wayne');"  
24     //conector.atualizar("update aluno set nome='Jason Todd' where RA='13246'");  
25     int count = 0;  
26     ResultSet resultado = conector.selecionar("select * from aluno");  
27     while (resultado.next()) {  
28         String RA = resultado.getString(1);  
29         String nome = resultado.getString(2);  
30         String rg = resultado.getString(3);  
31         String endereco = resultado.getString(4);  
32         String output = "Aluno #id: %s - %s - %s - %s";  
33         System.out.println(String.format(output, ++count, RA, nome, rg, endereco));  
34     }  
35     conector.encerrarConexao();  
36 }  
37 }  
38 }  
39 }
```

Output - Aula13_14F (run)

```
run:  
[OK] CONEXÃO ESTABELECIDA  
Aluno #1: 13246 - Jason Todd - 12365 - Mansão Wayne  
Aluno #2: 1111111 - Antonio Souza - 12345 - Rua X  
[ERRO] CONEXÃO ENCERRADA  
BUILD SUCCESSFUL (total time: 0 seconds)
```

2 - Main onde vc USA O CÓDIGO:

A - Olhe a seta **Azul** - comente a linha com o comando de inserir (senão vai inserir duas vezes ao executar)

B - Insira o código em **VERMELHO** e rode

DELETAR DADOS:


```
80 public void deletar(String sql) throws SQLException {
81     try{
82         PreparedStatement statement = conexao.prepareStatement(sql);
83         int rowsDeleted= statement.executeUpdate();
84         if (rowsDeleted > 0) {
85             System.out.println("A data was excluded successfully!");
86         }
87     }catch(SQLException ex){
88         System.out.println("[ERRO] ao Encerrar na CONEXÃO "+ ex);
89         System.exit(404);
90     }
91 }
92 }
93 }
```

1 - NA CLASSE JDBCConnector que foi criada:

A - Insira o código.

B - não esqueça de que tem uma ÚLTIMA CHAVE da Classe (NÃO A PERCA)

```
20 public static void main(String[] args) throws SQLException {
21     // TODO code application logic here
22     conector=new JDBCConnector();
23     conector.iniciarConexao();
24     //conector.inserir("insert into aluno values('13246','Dick Greyson','12365','Mansão Wayne','M','1995-05-14','2')");
25     // conector.atualizar("update aluno set nome='Jason Todd' where RA='13246'");
26     //int count = 0;
27     //ResultSet resultado = conector.selecionar("select * from aluno");
28     // while (resultado.next()){
29     //     String RA = resultado.getString(1);
30     //     String nome = resultado.getString(2);
31     //     String rg = resultado.getString(3);
32     //     String endereco = resultado.getString(4);
33     //     String output = "Aluno #d: %s - %s - %s - %s";
34     //     System.out.println(String.format(output, ++count, RA, nome, rg, endereco));
35     conector.deletar("delete from aluno where nome='Jason Todd'");
36     conector.encerrarConexao();
37 }
38
39
```

2 - Main onde vc USA O CÓDIGO:

- A - Olhe a seta **Azul** - comente a linha com o comando de inserir (senão vai inserir duas vezes ao executar)
- B - Insira o código em **VERMELHO** e rode

27 • `select * from alund`

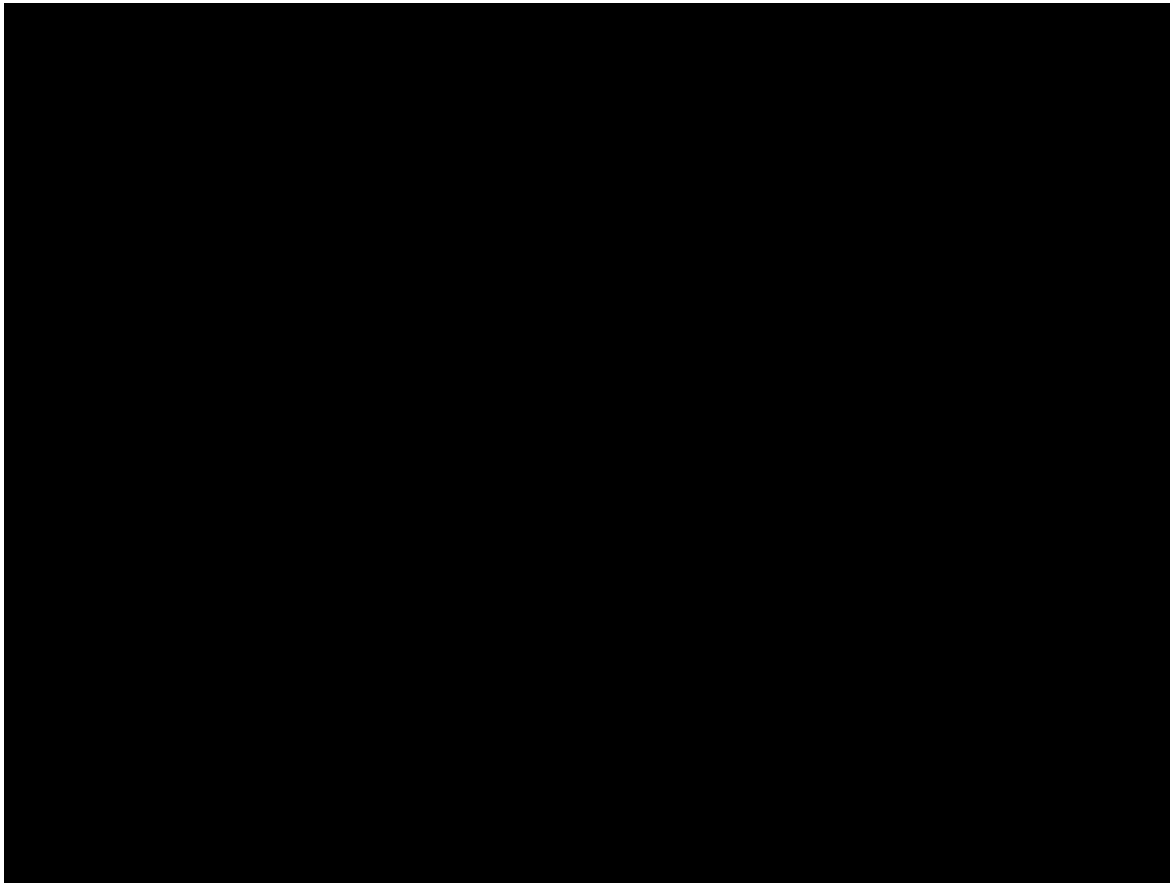
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: [IA](#)

	RA	nome	rg	endereco	sexo	dt_nasc	cd_curso
▶	1111111	Antonio Souza	12345	Rua X	M	2022-02-23	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Jason?

2 - Main onde vc USA O CÓDIGO:

- A - Olhe a seta **Azul** - comente a linha com o comando de inserir (senão vai inserir duas vezes ao executar)
- B - Insira o código em **VERMELHO** e rode



Caso não acesse, o vídeo está em:
https://drive.google.com/file/d/15Hf9Mas2I_j1CM-VyF6Y3V_H-oZ1sEzA/view?usp=sharing