

Exercício Banco de dados SQL

1. Abra o IDE DBeaver (ou o que você tiver acesso) e crie a seguinte tabela:

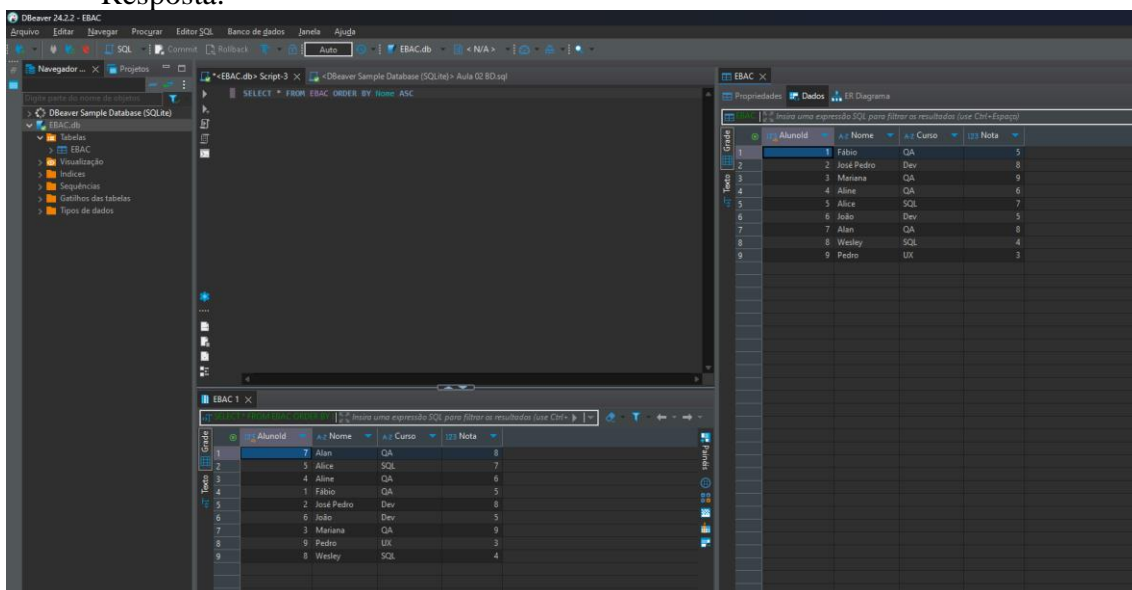
```
CREATE TABLE EBAC(  
  AlunoId INTEGER PRIMARY KEY AUTOINCREMENT,  
  Nome VARCHAR(30),  
  Curso VARCHAR (20),  
  Nota INTEGER(2)  
);
```

2. Na sequência insira os seguintes dados:

```
INSERT INTO EBAC (Nome, Curso, Nota)  
VALUES
```

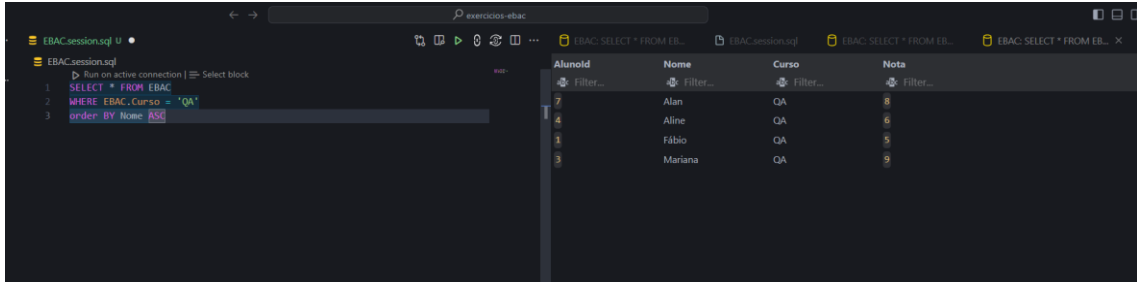
```
("Fábio", "QA", 5),  
("José Pedro", "Dev", 8),  
("Mariana", "QA", 9),  
("Aline", "QA", 6),  
("Alice", "SQL", 7),  
("João", "Dev", 5),  
("Alan", "QA", 8),  
("Wesley", "SQL", 4),  
("Pedro", "UX", 3);
```

3. Selecione todos os dados da tabela EBAC, ordenando o nome em ordem alfabética.
Resposta:



4. Selecione Todos os alunos do curso de QA.

Resposta:



The screenshot shows a SQL IDE with a query editor on the left and a results table on the right. The query editor contains the following SQL code:

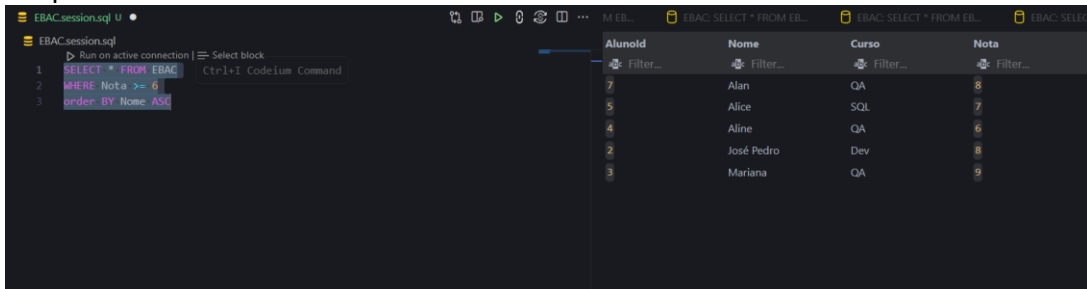
```
1 SELECT * FROM EBAC
2 WHERE EBAC.Curso = 'QA'
3 order BY Nome ASC
```

The results table displays the following data:

AlunoId	Nome	Curso	Nota
7	Alan	QA	8
4	Aline	QA	6
1	Fábio	QA	5
3	Mariana	QA	9

5. Selecione todos os alunos com nota maior e igual a 6.

Resposta:



The screenshot shows a SQL IDE with a query editor on the left and a results table on the right. The query editor contains the following SQL code:

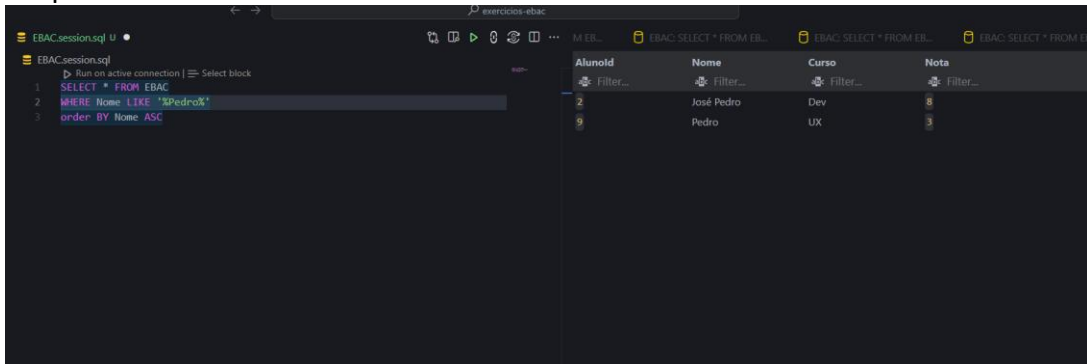
```
1 SELECT * FROM EBAC
2 WHERE Nota >= 6
3 order BY Nome ASC
```

The results table displays the following data:

AlunoId	Nome	Curso	Nota
7	Alan	QA	8
5	Alice	SQL	7
4	Aline	QA	6
2	José Pedro	Dev	8
3	Mariana	QA	9

6. Selecione todos os alunos que tem a palavra “Pedro” no nome.

Resposta:



The screenshot shows a SQL IDE with a query editor on the left and a results table on the right. The query editor contains the following SQL code:

```
1 SELECT * FROM EBAC
2 WHERE Nome LIKE '%Pedro%'
3 order BY Nome ASC
```

The results table displays the following data:

AlunoId	Nome	Curso	Nota
2	José Pedro	Dev	8
9	Pedro	LUX	3

Exercício Banco de dados MongoDB

1. Execute o docker e abra o MongoDB Compass e crie o seguinte banco:

use EBAC

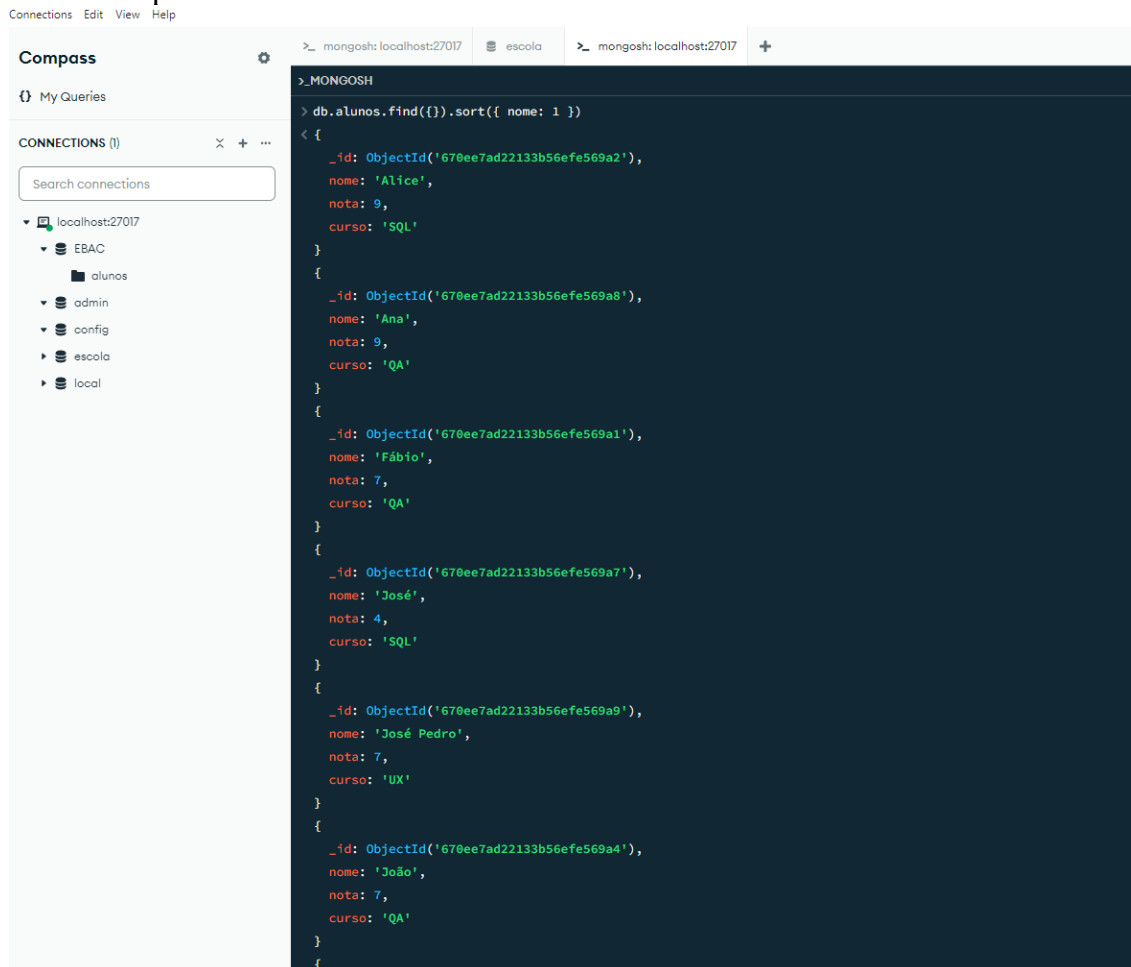
2. Crie a seguinte coleção:

```
db.alunos.insertMany([
  {
    "nome": "Fábio",
    "nota": 7,
    "curso": "QA"
  },
  {
    "nome": "Alice",
    "nota": 9,
    "curso": "SQL"
  },
  {
    "nome": "Mariana",
    "cargo": "Professora",
    "curso": ["QA", "FrontEnd", "MongoDB"]
  },
  {
    "nome": "João",
    "nota": 7,
    "curso": "QA"
  },
  {
    "nome": "Paulo",
    "nota": 5,
    "curso": "Dev"
  },
  {
    "nome": "Maria",
    "nota": 8,
    "curso": "QA"
  },
  {
    "nome": "José",
    "nota": 4,
    "curso": "SQL"
  },
  {
    "nome": "Ana",
    "nota": 9,
    "curso": "QA"
  },
  {
    "nome": "José Pedro",
    "nota": 7,
    "curso": "UX"
  }
])
```

```
}  
])
```

1. Selecione todos os dados da Collection Alunos, ordenando o nome em ordem alfabética.

Resposta:



The screenshot shows the MongoDB Compass interface. On the left, the 'CONNECTIONS' panel lists the 'alunos' collection under the 'escola' database. The main editor shows a query: `db.alunos.find().sort({ nome: 1 })`. The results are displayed as a JSON array of documents, sorted by name.

```
> db.alunos.find().sort({ nome: 1 })  
< {  
  "_id": ObjectId('670ee7ad22133b56efe569a2'),  
  "nome": 'Alice',  
  "nota": 9,  
  "curso": 'SQL'  
}  
{  
  "_id": ObjectId('670ee7ad22133b56efe569a8'),  
  "nome": 'Ana',  
  "nota": 9,  
  "curso": 'QA'  
}  
{  
  "_id": ObjectId('670ee7ad22133b56efe569a1'),  
  "nome": 'Fábio',  
  "nota": 7,  
  "curso": 'QA'  
}  
{  
  "_id": ObjectId('670ee7ad22133b56efe569a7'),  
  "nome": 'José',  
  "nota": 4,  
  "curso": 'SQL'  
}  
{  
  "_id": ObjectId('670ee7ad22133b56efe569a9'),  
  "nome": 'José Pedro',  
  "nota": 7,  
  "curso": 'UX'  
}  
{  
  "_id": ObjectId('670ee7ad22133b56efe569a4'),  
  "nome": 'João',  
  "nota": 7,  
  "curso": 'QA'  
}  
}
```

2. Selecione todos os alunos do curso de SQL.

Resposta:

```
>_MONGOSH
> db.alunos.find({curso: "SQL"})
< {
  _id: ObjectId('670ee7ad22133b56efe569a2'),
  nome: 'Alice',
  nota: 9,
  curso: 'SQL'
}
{
  _id: ObjectId('670ee7ad22133b56efe569a7'),
  nome: 'José',
  nota: 4,
  curso: 'SQL'
}
EBAC>
```

3. Selecione todos os alunos com “nota maior e igual a 6” e “do curso de QA”.

Resposta:

```
>_MONGOSH
> db.alunos.find({
  $and: [
    { nota: { $gte: 6 } },
    { curso: "QA" }
  ]
})
< {
  _id: ObjectId('670ee7ad22133b56efe569a1'),
  nome: 'Fábio',
  nota: 7,
  curso: 'QA'
}
{
  _id: ObjectId('670ee7ad22133b56efe569a4'),
  nome: 'João',
  nota: 7,
  curso: 'QA'
}
{
  _id: ObjectId('670ee7ad22133b56efe569a6'),
  nome: 'Maria',
  nota: 8,
  curso: 'QA'
}
{
  _id: ObjectId('670ee7ad22133b56efe569a8'),
  nome: 'Ana',
  nota: 9,
  curso: 'QA'
}
EBAC>
```

4. Selecione todos os alunos que tem a palavra “Pedro” no nome.
Resposta:

```
>_MONGOSH  
  
> db.alunos.find({nome: /Pedro/})  
< {  
  _id: ObjectId('670ee7ad22133b56efe569a9'),  
  nome: 'José Pedro',  
  nota: 7,  
  curso: 'UX'  
}  
EBAC>
```