1) Com usuário administrador criar o Database REVISAODB2

```
create database REVISAODB2
```

2) Criar o login "Estudlg" (senha 12345) no servidor (desconsidere a politica de senhas do servidor) e o associe ao database REVISAODB2

```
create login Estudlg with password = '12345', default_database = REVISAODB2,
check_policy=off
```

3) Criar o usuário "Estud1" no database "REVISAODB2", associando o mesmo ao login "Estudlg"

```
use REVISAODB2
create user Estud1 for login Estudlg
```

4) Criar esquema SREV no database REVISAODB2

```
create schema SREV;
```

5) Criar uma tabela XPTO (X int, Y int, Z int) no esquema SREV; Criar uma VIEW VXPTO, mostrando somente (X,Y),
-- no esquema SREV; Criar uma procedure P1 no esquema SREV com o comando "SELECT 'REVISAO DB2'

```
create table srev.XPTO (
x int,
y int,
z int,
);

create view srev.VXPTO as
select x,y from srev.xpto
select * from srev.vxpto

create procedure srev.p1 as
select 'REVISAO DE BD2'
exec srev.p1
```

6) Criar uma Role de servidor chamada RSREV; Tornar o login "Estudlg" membro da Role a RSREV

```
create server role RSREV
alter server role RSREV add member Estudlg
```

7) Apagar Role de servidor chamada RSREV

```
drop server role RSREV
```

8) Criar uma Role de DATABASE chamada RREV e tornar o usuário "Estud1" membro da Role a RREV

```
create role RREV
alter role RREV add member Estud1
```

9) Retirar o usuário "Estud1" como membro da Role a RREV
```
alter role RREV drop member Estud1
```

10)
-- Tornar o usuário "Estud1" apto a criar objetos no Database
```
alter server role db_DDLADMIN add member Estudlg
```

-- Tornar o usuário "Estud1" owner do Database
```
alter server role db_OWNER  add member Estudlg
```

-- Tornar o usuário "Estud1" apto tirar backup no Database
```
alter server role db_Backupoperator add member Estudlg
```

11) Conceder ao usuário "Estud1" acesso de leitura às tabelas do esquema SREV
```
grant select on schema::srev to Estud1
```

12) Dar permissão de leitura e execução no procedimento armazenado SREV.P1 a role RREV
```
grant execute on srev.p1 to RREV
```

13) Negar o acesso de Leitura a coluna X da tabela SREV.XPTO a role RREV
```
deny select on srev.xpto(x) to RREV
```

14) Fornecer permissão de inserção e atualização das tabelas de SREV à role RREV, e testar com usuário Estud1
```
grant insert, update on schema::srev to RREV
```

15) Liste os índices existentes na tabela CONSULTAS e explique de onde eles se originaram
Não tem indices, devido não ter chaves primárias

16) Crie um índice clusterizado no campo "codp" da tabela Consultas para a consulta SELECT * FROM Consultas where codp=1020
```
create clustered index CodPIdx on consultas(codp)
```

17) Crie um índice não clusterizado no campo "DATA" da tabela Consultas para melhorar o desempenho da consulta SELECT * FROM Consultas where codp=1020 and data = '30/09/2015'
```
create nonclustered index DataIdx on consultas(data)
```

18) Liste os índices existentes na tabela PACIENTES e explique de onde eles se originaram. Qual a origem de cada um?
codp, índice clusterizado
nome, índice non clusterizado

19)
/* Criar Transações para Consultas Médicas – marcação única
Inserir uma transação contendo uma consulta, passando como parâmetros código do médico, código do paciente, data e hora. A transação deve ser confirmada se não houver erros (marcação duplicada, marcação em horário já utilizado), caso contrário deve ser desfeita, informando via PRINT o motivo.

Nome da transação (procedimento) – T1
Parâmetros de teste –
T1 (49, 1070, "2022/08/01", "09:00")
T1 (49, 1030, "2022/08/01", "09:00")
T1 (50, 1030, "2022/08/01", "09:30")
T1 (49, 1070, "2022/08/01", "09:00")
T1 (51, 1030, "2022/08/01", "09:30") */

```
Create PROCEDURE T1 (@MEDICO INT, @PACIENTE INT,  @DATA date,  @HORA time(7)) AS
BEGIN
 BEGIN TRANSACTION
-- INSERE A CONSULTA
        INSERT INTO CONSULTLAS VALUES (@MEDICO, @PACIENTE, @DATA, @HORA)

-- TESTA DUPLICIDADE (COMO A TABELA NAO TEM CHAVE PRIMARIA< ESTE TESTE é
NECESS€RIO
        IF (SELECT COUNT(*) FROM CONSULTAS WHERE CODM=@MEDICO AND CODP=@PACIENTE
AND
 DATA=@DATA AND HORA=@HORA) > 1
        BEGIN
                PRINT ('DUPLICADO')
                ROLLBACK
        END

-- TESTA SE O MEDICO Jé TEM CONSULTA NESTA DATA/HORA
        ELSE
         IF (SELECT COUNT(*) FROM CONSULTAS WHERE CODM=@MEDICO AND DATA=@DATA AND
HORA=@HORA) > 1
         BEGIN
                PRINT ('HORARIO Já UTILIZADO!!')
                ROLLBACK
         END
         ELSE
-- TUDO OK - EFETIVA A MARCA€€O
         BEGIN
                 PRINT('EFETIVADO')
                 COMMIT
         END
 END

--- TESTANDO
EXEC T1 49, 1030, '2022/08/01', '09:00' -- EFETIVADA
EXEC T1 49, 1030, '2022/08/01', '09:00' -- DUPLICADA
EXEC T1 50, 1030, '2022/08/01', '09:30' -- EFETIVADA
EXEC T1 49, 1070, '2022/08/01', '09:00' -- HORARIO Jé UTILIZADO
EXEC T1 51, 1030, '2022/08/01', '09:30' -- EFETIVADA (Outro Medico)
```