



Centro Universitário das Faculdades Associadas de Ensino - FAE

## 04 – BANCO DE DADOS COMPARTILHADO (*Shared Database*)

INTEGRAÇÃO DE APLICAÇÕES

Prof. M.Sc. Tiago Rosa

# Banco de Dados Compartilhado

- Um banco de dados compartilhado garante uma estrutura comum, padronizada e obrigatório.
- Garante também atualização constante.
- Um banco de dados é projetado para lidar com acesso concorrente, níveis de isolamento, transações, etc.
  - sendo uma solução mais completa quando esses atributos forem necessários.
- Pode ser um banco de dados relacional, mas pode ser qualquer outra solução de banco de dados como XML ou NoSQL.

# Banco de Dados Compartilhado

- Um banco de dados compartilhado garante uma estrutura comum, padronizada e obrigatório.
- **Garante também atualização constante.**
- Um banco de dados é projetado para lidar com acesso concorrente, níveis de isolamento, transações, etc.
  - sendo uma solução mais completa quando esses atributos forem necessários.
- Pode ser um banco de dados relacional, mas pode ser qualquer outra solução de banco de dados como XML ou NoSQL.

# Banco de Dados Compartilhado

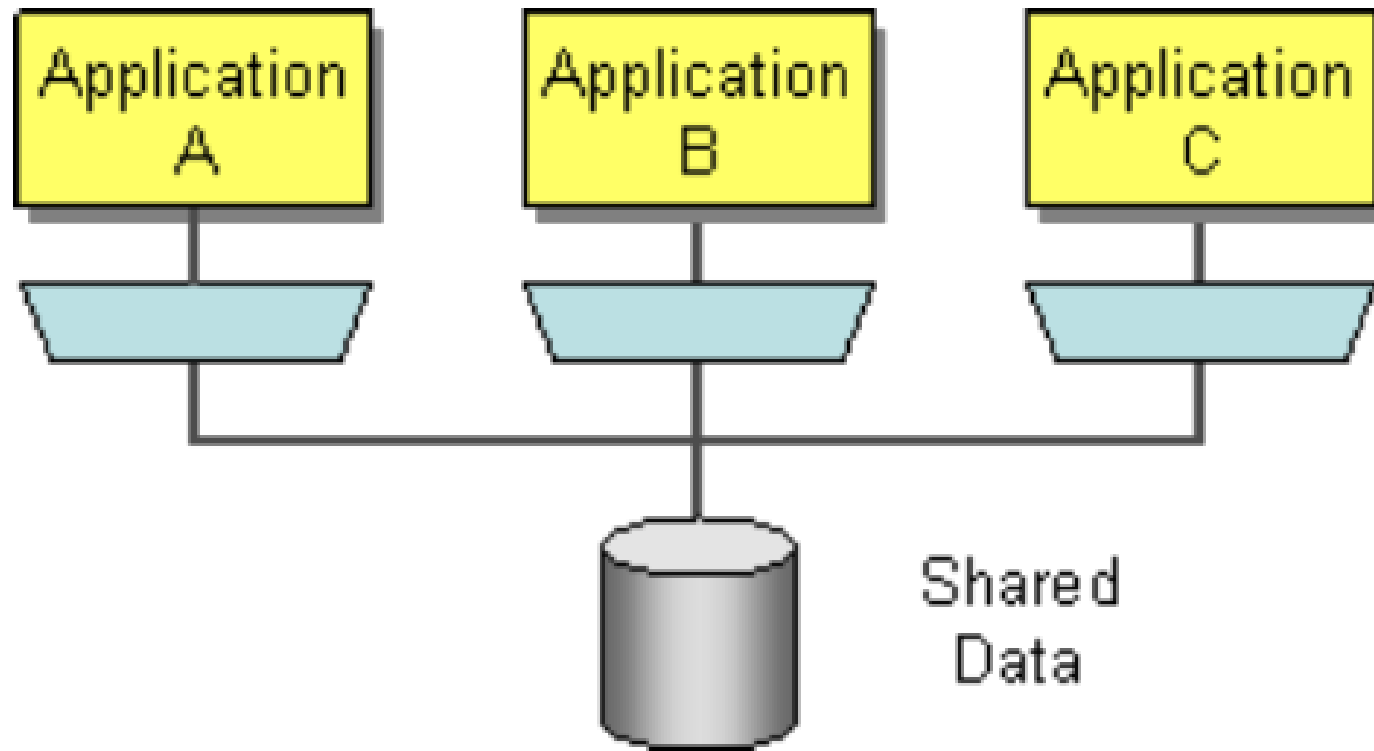
- Um banco de dados compartilhado garante uma estrutura comum, padronizada e obrigatório.
- Garante também atualização constante.
- Um banco de dados é projetado para lidar com acesso concorrente, níveis de isolamento, transações, etc.
  - sendo uma solução mais completa quando esses atributos forem necessários.
- Pode ser um banco de dados relacional, mas pode ser qualquer outra solução de banco de dados como XML ou NoSQL.

# Banco de Dados Compartilhado

- Um banco de dados compartilhado garante uma estrutura comum, padronizada e obrigatório.
- Garante também atualização constante.
- Um banco de dados é projetado para lidar com acesso concorrente, níveis de isolamento, transações, etc.
  - sendo uma solução mais completa quando esses atributos forem necessários.
- Pode ser um banco de dados relacional, mas pode ser qualquer outra solução de banco de dados como XML ou NoSQL.

# Banco de Dados Compartilhado

## Diagrama



# Banco de Dados Compartilhado

- A integração através de banco de dados compartilhado pode ser feita de três maneiras:
  - Integração através de rotinas
  - Integração através de base de interface
  - Integração online
- Será estudado três cenários que resumem algumas situações para que possamos exemplificar como realizar a integração em cada uma das maneiras.

# Integração Através de Rotinas



# Integração através de rotinas

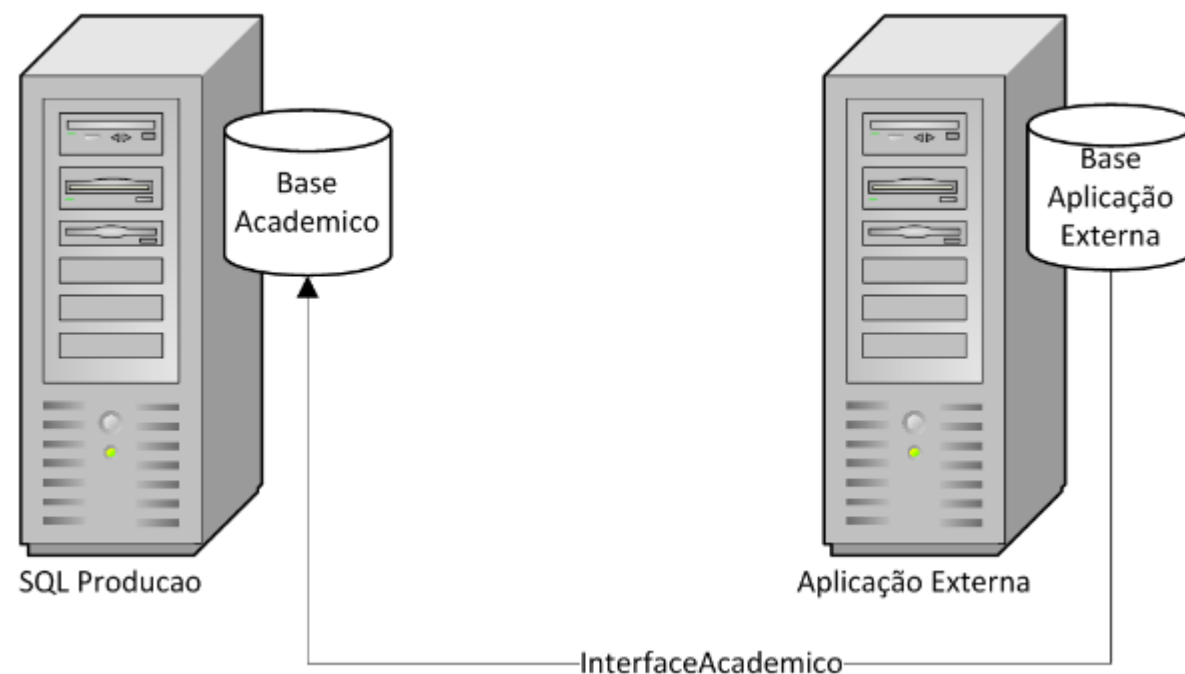
- Dentre os cenários que vamos criar, este é o que possui a solução mais simples.
- Este cenário descreve que temos dois sistemas que compartilham informação, mas apenas um deles tem a responsabilidade de mantê-las.
- Faz-se necessário expandir sua utilização, adicionando novas funcionalidades ou introduzindo um módulo web, por exemplo.

# Integração através de rotinas

- Imagine que temos um ERP para controle acadêmico,
  - responsável pelo cadastro de alunos, cursos e matrículas,
- e vamos introduzir um sistema de terceiros que precisa das informações de alunos e suas matrículas.
- Além disso, também faz-se necessário que o sistema externo insira informações sobre alunos e cursos.
- Para dificultar um pouco e tornar o cenário mais real, devemos considerar que o padrão de segurança da instituição não permite que terceiros acessem a base de produção.

# Integração através de rotinas

- Diagrama para ilustrar essa situação.



# Integração através de rotinas

- O que deve ser feito:
  - Criar algumas rotinas para que o sistema externo acesse a base de produção e faça suas modificações.
    - Isso deve ser feito para que a base de produção **não seja acessada diretamente**.
  - Em seguida, vamos criar **um usuário específico** e **dar acesso externo apenas a estas rotinas**.

# Integração através de rotinas

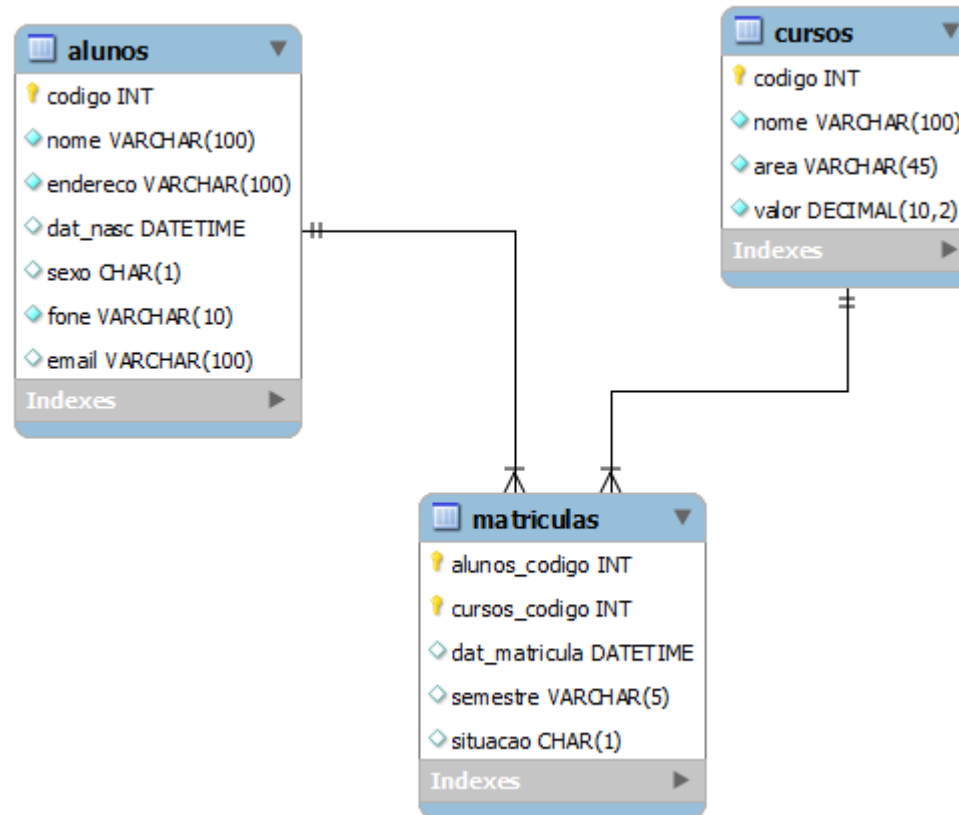
- Criar algumas rotinas para que o sistema externo acesse a base de produção e faça suas modificações.
  - As rotinas necessárias ao sistema externo são:
    - Listagem de alunos matriculados;
    - Listagem de cursos disponíveis;
    - Inserir alunos;
    - Inserir cursos.

# Integração através de rotinas

- Criar algumas rotinas para que o sistema externo acesse a base de produção e faça suas modificações.
  - As rotinas deverão ser criadas através de *Stored Procedures*!

# Integração através de rotinas

- Diagrama do banco e dados do Cenário 1.



Integração através de rotinas

O Script de criação do banco de dados do Cenário 1 esta disponível no portal!



# Integração através de rotinas

- Criando as rotinas para que o sistema externo acesse a base de produção

```
DELIMITER $$  
CREATE PROCEDURE retornaAlunosMatriculados()  
BEGIN  
    SELECT M.dat_matricula, M.semestre, A.nome, A.email, C.nome  
    FROM Matriculas M  
    INNER JOIN Alunos A ON (A.codigo = M.alunos_codigo)  
    INNER JOIN Cursos C ON (C.codigo = M.cursos_codigo)  
    WHERE M.situacao = 'A';  
END
```

Rotina que retorna os alunos matriculados

# Integração através de rotinas

- Criando as rotinas para que o sistema externo acesse a base de produção

```
DELIMITER $$  
CREATE PROCEDURE retornaAlunosMatriculados()  
BEGIN  
    SELECT M.dat_matricula, M.semestre, A.nome, A.email, C.nome  
    FROM Matriculas M  
    INNER JOIN Alunos A ON (A.codigo = M.alunos_codigo)  
    INNER JOIN Cursos C ON (C.codigo = M.cursos_codigo)  
    WHERE M.situacao = 'A';  
END
```

Rotina que retorna os alunos matriculados

# Integração através de rotinas

- Criando as rotinas para que o sistema externo acesse a base de produção

```
DELIMITER $$  
CREATE PROCEDURE retornaCursos()  
BEGIN  
    SELECT codigo, nome, valor  
    FROM Cursos C;  
END
```

Rotina que retorna os cursos cadastrados

# Integração através de rotinas

- Criando as rotinas para que o sistema externo acesse a base de produção

```
DELI M TER $$  
  
DROP PROCEDURE I F EXI STS retornaAl unos $$  
CREATE PROCEDURE retornaAl unos()  
BEGI N  
  
    SELECT *  
    FROM alunos;  
  
END $$  
  
DELI M TER ;
```

Rotina que retorna os cursos cadastrados

# Integração através de rotinas

- Além das rotinas de consulta, também é necessário criar rotinas para que o novo sistema **insira algumas informações em produção**,
- mas **nunca acessando diretamente a base de produção**,
  - isto é, sempre passando por rotinas desenvolvidas internamente.

# Integração através de rotinas

- Rotinas de inserção de dados:

```
DELIMITER $$
CREATE PROCEDURE insereAluno(
    IN codigo    int,
    IN nome      varchar(100),
    IN endereco  varchar(100),
    IN datanasc  datetime,
    IN sexo      char(1),
    IN fone      varchar(10),
    IN email     varchar(100)
)
BEGIN
    INSERT INTO Alunos (codigo, nome, endereco, datanasc, sexo, fone, email)
    VALUES (codigo, nome, endereco, datanasc, sexo, fone, email);
END
```

Rotina que permite inserir dados de alunos

# Integração através de rotinas

- Rotinas de inserção de dados:

```
DELIMITER $$  
CREATE PROCEDURE InsereCurso(  
    IN codigo    int,  
    IN nome      varchar(100) ,  
    IN area      varchar(50) ,  
    IN valor     decimal(10,2)  
)  
BEGIN  
  
    INSERT INTO Cursos (codigo, nome, area, valor)  
    VALUES (codigo, nome, area, valor);  
  
END
```

Rotina que permite inserir dados de cursos

# Integração através de rotinas

- Com todas as rotinas solicitadas prontas, vamos criar um usuário específico para utilizá-las.
- Quando desenvolvemos interfaces desta maneira é importante deixar um usuário para cada interface ou sistema externo que esteja acessando nossa base de produção.
- Dessa maneira conseguimos isolar e medir a utilização deste usuário na base, além de controlar as permissões de forma mais organizada.



# Integração através de rotinas

- Criando um novo usuário

```
USE academico;  
CREATE USER 'externo'@'localhost' IDENTIFIED BY '123456';
```

- Dando privilegio de execução as rotinas criadas

```
GRANT EXECUTE ON PROCEDURE insereAluno TO 'externo'@'localhost';  
GRANT EXECUTE ON PROCEDURE insereCurso TO 'externo'@'localhost';  
GRANT EXECUTE ON PROCEDURE retornaAlunosMatriculados TO 'externo'@'localhost';  
GRANT EXECUTE ON PROCEDURE retornaCursos TO 'externo'@'localhost';
```

# Integração através de rotinas

- Pontos positivos:
  - O sistema externo não acessa diretamente a base de dados de produção;
  - A complexidade da base de produção será encapsulada através das rotinas, facilitando a integração;
  - Criação de um usuário específico para este fim;
  - O acesso direto aos dados da produção faz com que o sistema externo tenha sempre os mesmos dados da produção.

# Integração através de rotinas

- Pontos negativos:
  - Do ponto de vista de segurança, não é interessante fornecer dados de acesso da produção
    - IP do servidor e nomes de tabelas
  - para usuários externos. Mesmo com permissão restrita;
  - Dependendo do fluxo de utilização do sistema externo, o acesso direto à produção pode prejudicar seu desempenho.

# Integração Através de Base de Interface

# Integração Através de Base de Interface

- Sistemas diferentes com duas bases de dados que compartilham informação e **cada uma é responsável por parte dos dados** que são inseridos em produção.

**Portanto, nenhuma das duas bases possui o domínio total do conteúdo.**

# Integração Através de Base de Interface

- Requisitos de segurança que deve ser considerado:

O encapsulamento das bases de produção.

- Com isso, as bases de produção não devem ser acessadas diretamente, sendo utilizadas, portanto, rotinas específicas para estes acessos.

# Integração Através de Base de Interface

- Cenário de exemplo:
  - Vamos acrescentar mais uma tabela à nossa base de dados do sistema Acadêmico.
  - Será a tabela de lançamentos financeiros, onde:
    - são registrados todos os lançamentos financeiros das mensalidades dos alunos e
    - os pagamentos que foram efetuados.
  - A necessidade agora é se comunicar com uma empresa externa que faz a cobrança dos lançamentos que não foram pagos.
  - Deste modo:
    - devemos enviar à empresa os lançamentos em aberto e
    - a empresa, por sua vez, retorna as cobranças que foram pagas através de acordos e devem ser liquidadas na base de produção.

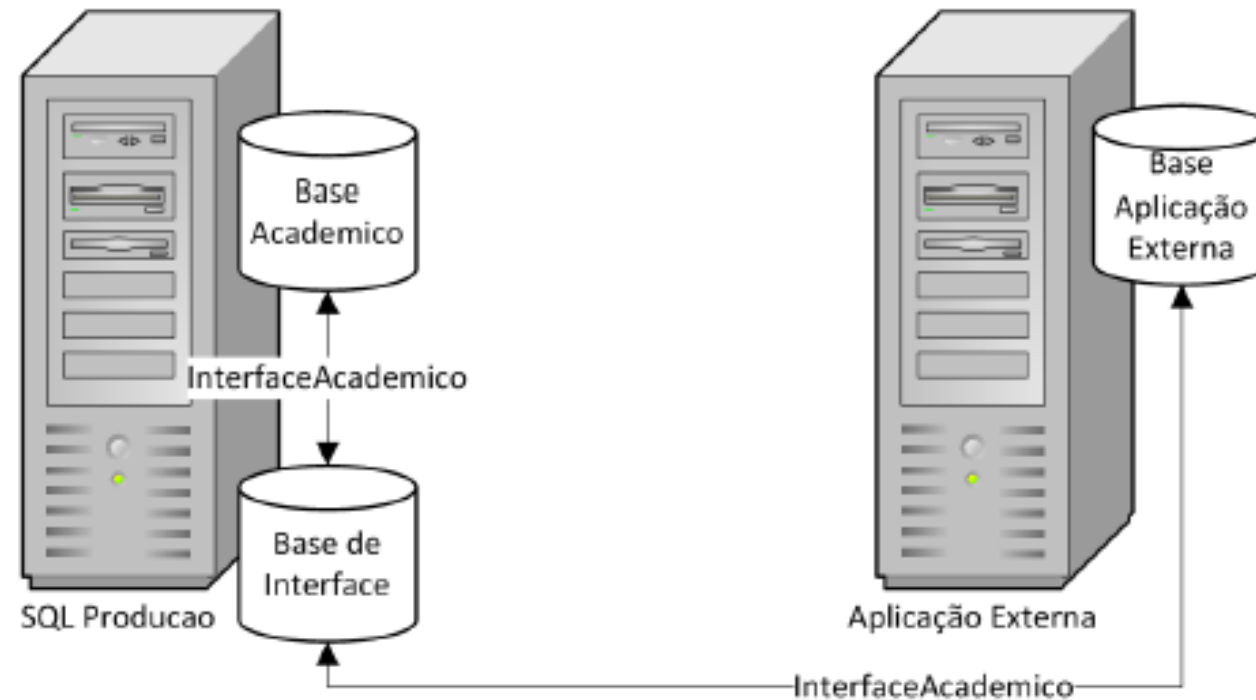
# Integração Através de Base de Interface

- Cenário de exemplo:
  - No entanto, dessa vez, em vez de criarmos apenas rotinas de acesso, vamos **criar uma nova base de dados**.
  - Esta base vai conter as informações fornecidas pelos dois sistemas para que se comuniquem utilizando apenas os dados necessários à integração.



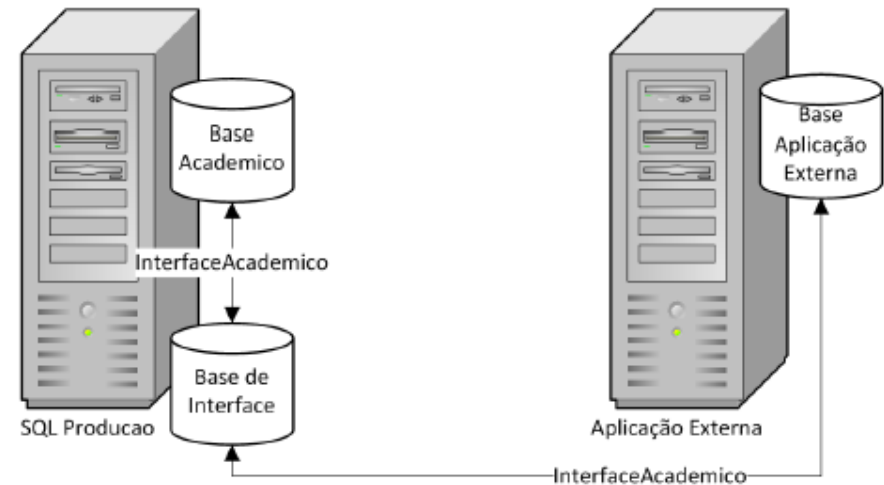
# Integração Através de Base de Interface

- Cenário de exemplo:



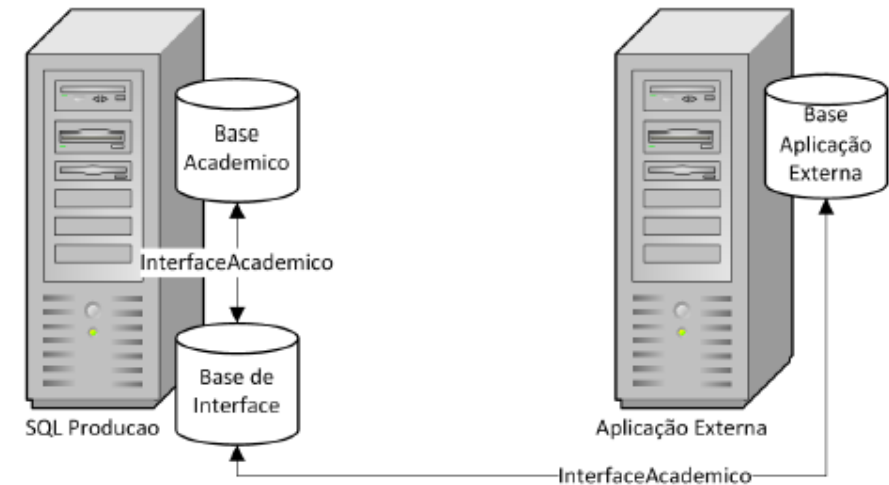
# Integração Através de Base de Interface

- Cenário de exemplo:
  - Adição de uma nova base de dados para comunicação entre os dois sistemas ,
  - será necessária a criação de rotinas que devem ser executadas por cada um deles.
    - uma rotina para consulta e
    - uma rotina para escrita na base de interface para cada sistema envolvido.



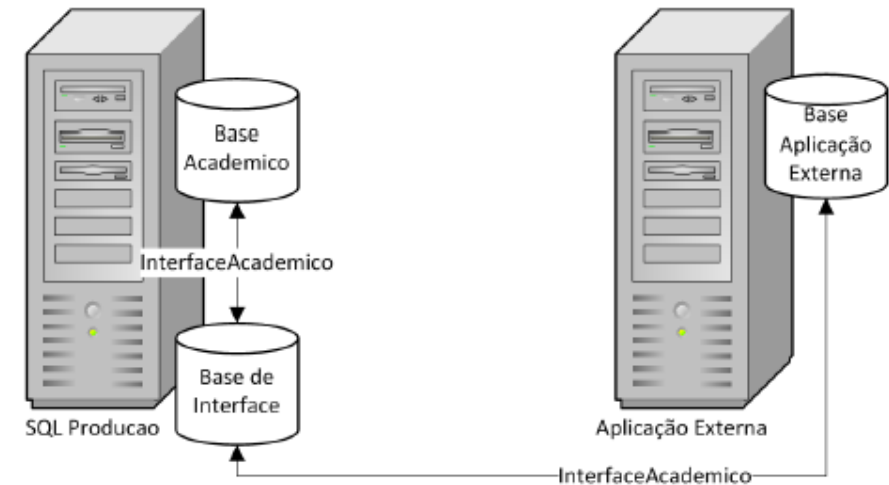
# Integração Através de Base de Interface

- Cenário de exemplo:
  - As rotinas necessárias para a montagem deste cenário são:
    - O sistema acadêmico irá inserir os lançamentos em aberto na nova base de dados de interface;
    - O sistema acadêmico irá consultar os pagamentos convertidos na base de interface;
    - O sistema de cobrança irá consultar os lançamentos em aberto que estão na base de interface e foram incluídos pelo sistema acadêmico;
    - O sistema de cobrança irá inserir na base de interface os pagamentos que foram convertidos através de acordos, de modo que possam ser consultados pelo sistema acadêmico.



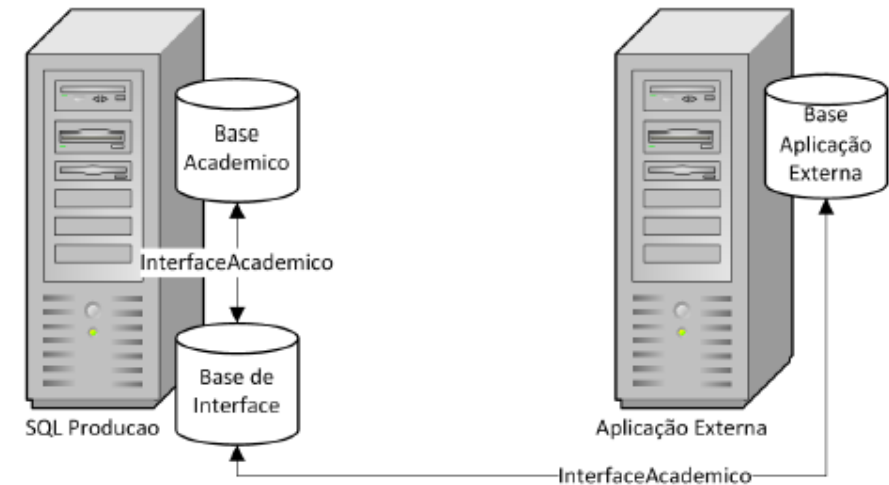
# Integração Através de Base de Interface

- Cenário de exemplo:
  - As rotinas necessárias para a montagem deste cenário são:
    - O sistema acadêmico irá inserir os lançamentos em aberto na nova base de dados de interface;
    - O sistema acadêmico irá consultar os pagamentos convertidos na base de interface;
    - O sistema de cobrança irá consultar os lançamentos em aberto que estão na base de interface e foram incluídos pelo sistema acadêmico;
    - O sistema de cobrança irá inserir na base de interface os pagamentos que foram convertidos através de acordos, de modo que possam ser consultados pelo sistema acadêmico.



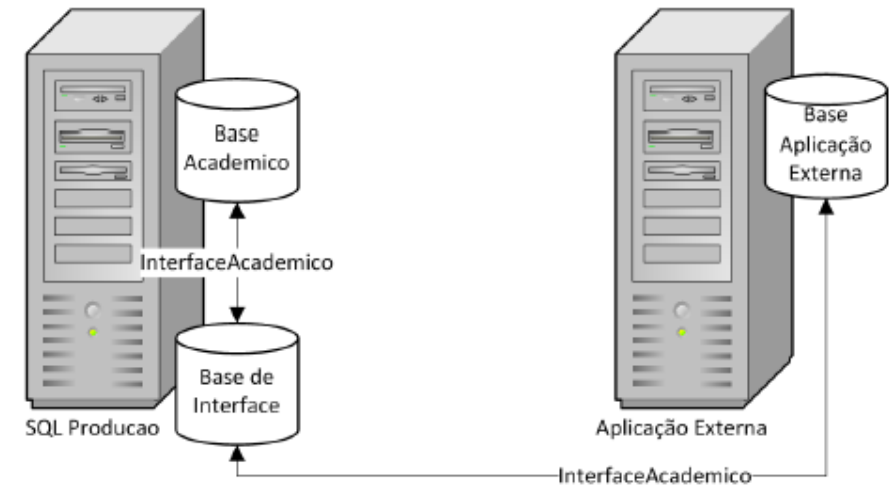
# Integração Através de Base de Interface

- Cenário de exemplo:
  - As rotinas necessárias para a montagem deste cenário são:
    - O sistema acadêmico irá inserir os lançamentos em aberto na nova base de dados de interface;
    - O sistema acadêmico irá consultar os pagamentos convertidos na base de interface;
    - O sistema de cobrança irá consultar os lançamentos em aberto que estão na base de interface e foram incluídos pelo sistema acadêmico;
    - O sistema de cobrança irá inserir na base de interface os pagamentos que foram convertidos através de acordos, de modo que possam ser consultados pelo sistema acadêmico.



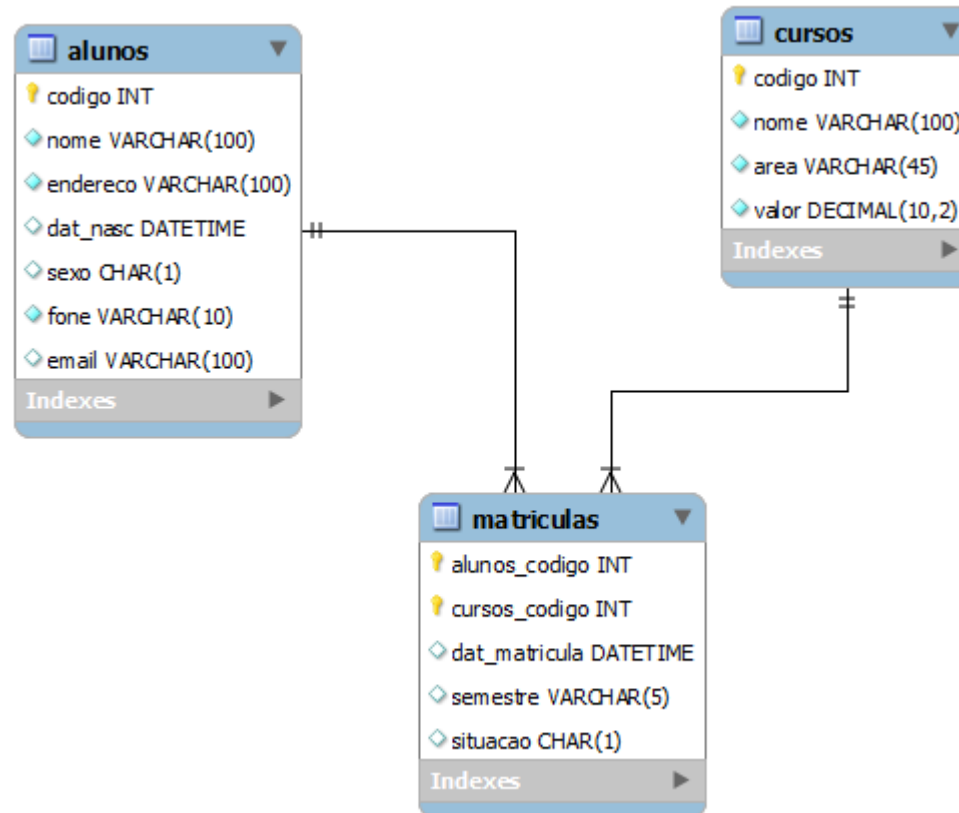
# Integração Através de Base de Interface

- Cenário de exemplo:
  - As rotinas necessárias para a montagem deste cenário são:
    - O sistema acadêmico irá inserir os lançamentos em aberto na nova base de dados de interface;
    - O sistema acadêmico irá consultar os pagamentos convertidos na base de interface;
    - O sistema de cobrança irá consultar os lançamentos em aberto que estão na base de interface e foram incluídos pelo sistema acadêmico;
    - O sistema de cobrança irá inserir na base de interface os pagamentos que foram convertidos através de acordos, de modo que possam ser consultados pelo sistema acadêmico.



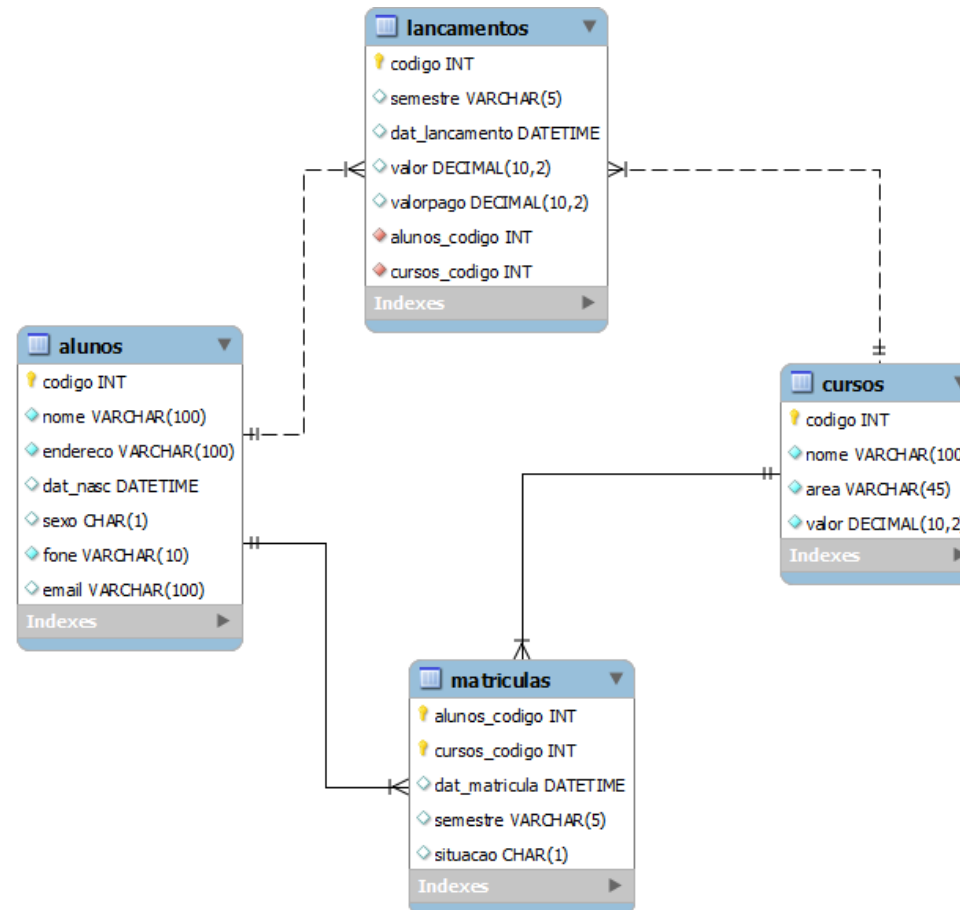
# Integração Através de Base de Interface

- Alterações necessárias no diagrama do banco e dados do Cenário 1.



# Integração Através de Base de Interface

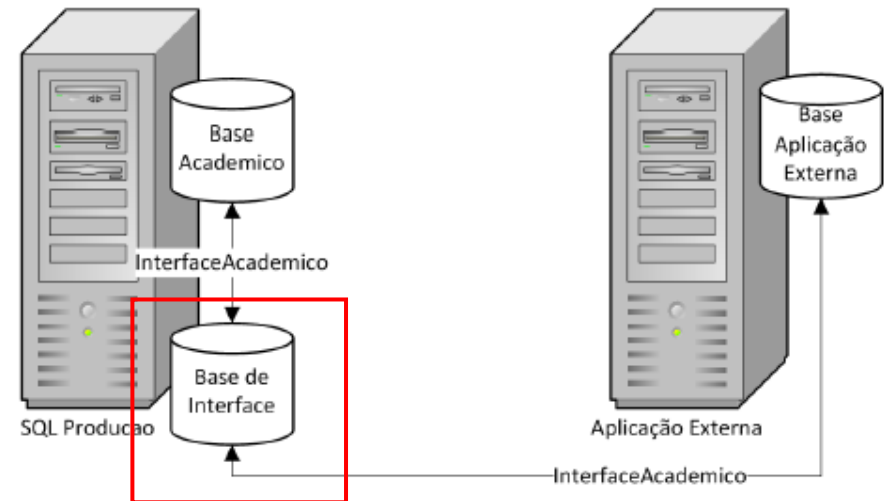
- Alterações necessárias no diagrama do banco e dados do Cenário 1.





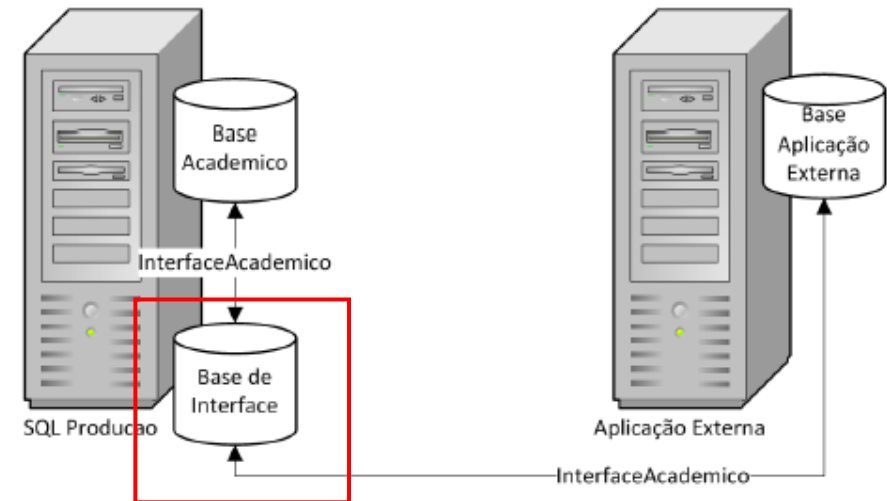
# Integração Através de Base de Interface

- Criação de um novo banco de dados, que fará a interface entre a nossa base Acadêmica e a base do sistema externo de cobrança
- Uma única tabela que vai receber as informações dos lançamentos em aberto, vindas do sistema acadêmico
- servirá para consulta a partir do sistema externo que vai fazer as cobranças necessárias.



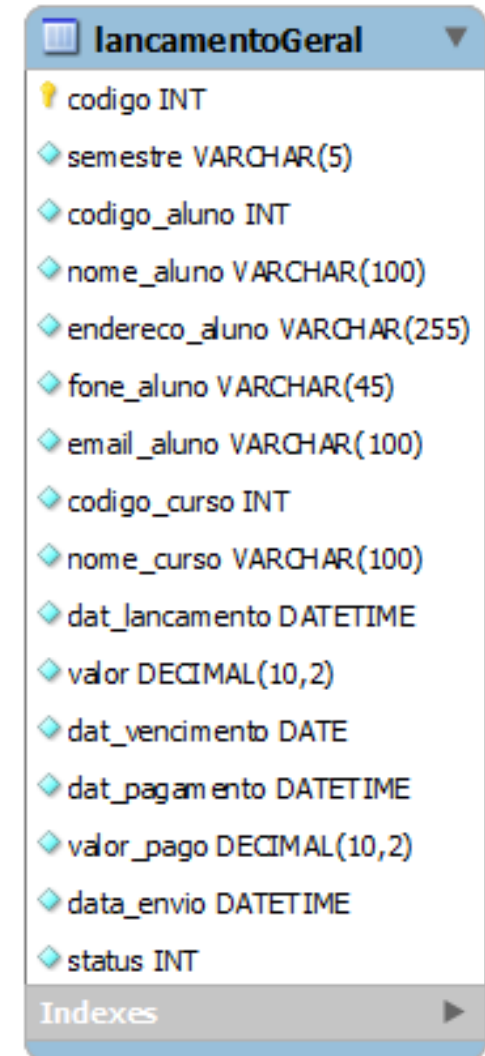
# Integração Através de Base de Interface

lancamentoGeral	
codigo	INT
semestre	VARCHAR(5)
codigo_aluno	INT
nome_aluno	VARCHAR(100)
endereco_aluno	VARCHAR(255)
fone_aluno	VARCHAR(45)
email_aluno	VARCHAR(100)
codigo_curso	INT
nome_curso	VARCHAR(100)
dat_lancamento	DATETIME
valor	DECIMAL(10,2)
dat_vencimento	DATE
dat_pagamento	DATETIME
valor_pago	DECIMAL(10,2)
data_envio	DATETIME
status	INT
Indexes	



# Integração Através de Base de Interface

- Normalmente, neste modelo de integração, os sistemas externos já possuem arquivos de layout que definem o formato dos dados para integração através de uma base de interface ou transferência de arquivos.



lancamentoGeral	
codigo	INT
semestre	VARCHAR(5)
codigo_aluno	INT
nome_aluno	VARCHAR(100)
endereco_aluno	VARCHAR(255)
fone_aluno	VARCHAR(45)
email_aluno	VARCHAR(100)
codigo_curso	INT
nome_curso	VARCHAR(100)
dat_lancamento	DATETIME
valor	DECIMAL(10,2)
dat_vencimento	DATE
dat_pagamento	DATETIME
valor_pago	DECIMAL(10,2)
data_envio	DATETIME
status	INT
Indexes ▶	

# Integração Através de Base de Interface

- Normalmente, neste modelo de integração, os sistemas externos já possuem arquivos de layout que definem o formato dos dados para integração através de uma base de interface ou transferência de arquivos.



lancamentoGeral	
codigo	INT
semestre	VARCHAR(5)
codigo_aluno	INT
nome_aluno	VARCHAR(100)
endereco_aluno	VARCHAR(255)
fone_aluno	VARCHAR(45)
email_aluno	VARCHAR(100)
codigo_curso	INT
nome_curso	VARCHAR(100)
dat_lancamento	DATETIME
valor	DECIMAL(10,2)
dat_vencimento	DATE
dat_pagamento	DATETIME
valor_pago	DECIMAL(10,2)
data_envio	DATETIME
status	INT
Indexes ▶	

# Rotinas de leitura e escrita para o sistema Acadêmico

```
DELIMITER $$
CREATE PROCEDURE InsereLancamentosNaoPagos ()
BEGIN
    INSERT INTO InterfaceCobranca.lancamentosGeral
    SELECT
        L.codigo, L.semestre, L.aluno,
        A.nome AS nomeAluno,
        A.endereco AS enderecoAluno,
        A.fone AS foneAluno,
        A.email AS emailAluno,
        C.codigo AS curso,
        C.nome AS nomeCurso,
        L.dataLanc,
        L.valor,
        NULL AS dataPgto,
        0.00 AS valorPgto,
        GETDATE() AS dataEnvi,
        0 AS status
    FROM Lancamentos L
        INNER JOIN Matriculas M ON (L.aluno = M.aluno AND L.curso= M.curso AND M.semestre = L.semestre)
        INNER JOIN Alunos A ON (M.aluno = A.codigo)
        INNER JOIN Cursos C ON (M.curso = C.codigo)
    WHERE L.dataPgto IS NULL AND L.dataVcto < GETDATE ()
END
```

# Rotinas de leitura e escrita para o sistema Acadêmico

```
DELIMITER $$  
CREATE PROCEDURE alteraLancamentosPagosCobranca()  
BEGIN  
    UPDATE Lancamentos SET dataPgto = G.dataPgto, valorPgto = G.valorPgto  
    FROM Lancamentos L  
    INNER JOIN InterfaceCobranca.dbo.LancamentosGeral G ON (L.codigo = G.codigo)  
    WHERE G.status = 1  
END
```

# Rotinas de leitura e escrita para o sistema de Cobrança (externo)

```
DELIMITER $$  
CREATE PROCEDURE consultaLancamentosNaoPagos( IN dataEnvio datetime )  
BEGIN  
    SELECT  
        codigo,  
        semestre,  
        aluno,  
        nomeAluno,  
        enderecoAluno,  
        foneAluno,  
        emailAluno,  
        curso,  
        nomeCurso,  
        dataLanc,  
        valor,  
        dataPgto,  
        valorPgto,  
        dataEnvi,  
        status  
    FROM lancamentosGeral  
    WHERE status = 0 AND dataEnvi = dataEnvio  
END
```

# Rotinas de leitura e escrita para o sistema de Cobrança (externo)

```
DELIMITER $$  
CREATE PROCEDURE alteraLancamentosCobranca (  
    IN cod int,  
    IN dat_pag datetime,  
    IN valPgto datetime  
)  
BEGIN  
    UPDATE lancamentosGeral SET status = 1, dataPgto = dat_pag, valorPgto = valPgto  
    WHERE codigo = cod  
END
```



# Permissões

- Atribuir permissões necessárias ao usuário, tanto na nova base de dados quanto nas rotinas criadas.
  - Rotinas criadas na base Academico.
  - Depois, vamos associar o nosso usuário à base InterfaceCobranca e
  - conceder as permissões às demais rotinas.

```
USE academico;  
GRANT EXECUTE ON PROCEDURE InsereLancamentosNaoPagos TO 'externo'@'localhost';  
GRANT EXECUTE ON PROCEDURE AlteraLancamentosPagosCobranca TO 'externo'@'localhost';
```

```
USE InterfaceCobranca  
CREATE USER 'interfaceAcademico'@'localhot' IDENTIFIED BY '123456';  
  
GRANT EXECUTE ON PROCEDURE ConsultaLancamentosNaoPagos TO 'interfaceAcademico'@'localhot'  
GRANT EXECUTE ON PROCEDURE AlteraLancamentosCobranca TO 'interfaceAcademico'@'localhot'
```

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- O fluxo na interface não interfere no desempenho da base de produção.
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- O fluxo na interface não interfere no desempenho da base de produção.
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- **A criação de um usuário específico para a base de interface;**
- O fluxo na interface não interfere no desempenho da base de produção.
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- **O fluxo na interface não interfere no desempenho da base de produção.**
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- O fluxo na interface não interfere no desempenho da base de produção.
- **Quanto aos pontos negativos, vale destacar:**
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- O fluxo na interface não interfere no desempenho da base de produção.
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.

# Pontos positivos e negativos

- O sistema externo não enxerga a base de dados de produção, apenas a interface;
- A complexidade da base de produção será encapsulada através das cargas na base de interface;
- A criação de um usuário específico para a base de interface;
- O fluxo na interface não interfere no desempenho da base de produção.
- Quanto aos pontos negativos, vale destacar:
- A base de interface só enxerga os novos dados após as cargas e por menor que seja o intervalo de tempo para essas atualizações, isso gera um delay em relação à produção;
- Houve um ligeiro aumento na complexidade para montagem deste cenário em relação ao anterior.



# Integração online

# Integração online

- Duas bases que compartilham dados e precisam refletir as mesmas informações em tempo real
- Caso uma linha seja alterada em uma tabela de uma das bases, a mesma linha deve ser alterada na outra base de dados.

# Integração online

- Para exemplificar este tipo de integração vamos considerar mais uma expansão do nosso ambiente acadêmico.
  - Criar uma nova base de dados para uma aplicação que deve registrar as informações das concessões de bolsas de estudo que,
  - vai compartilhar os dados de alunos com o sistema acadêmico já existente.

# Integração online

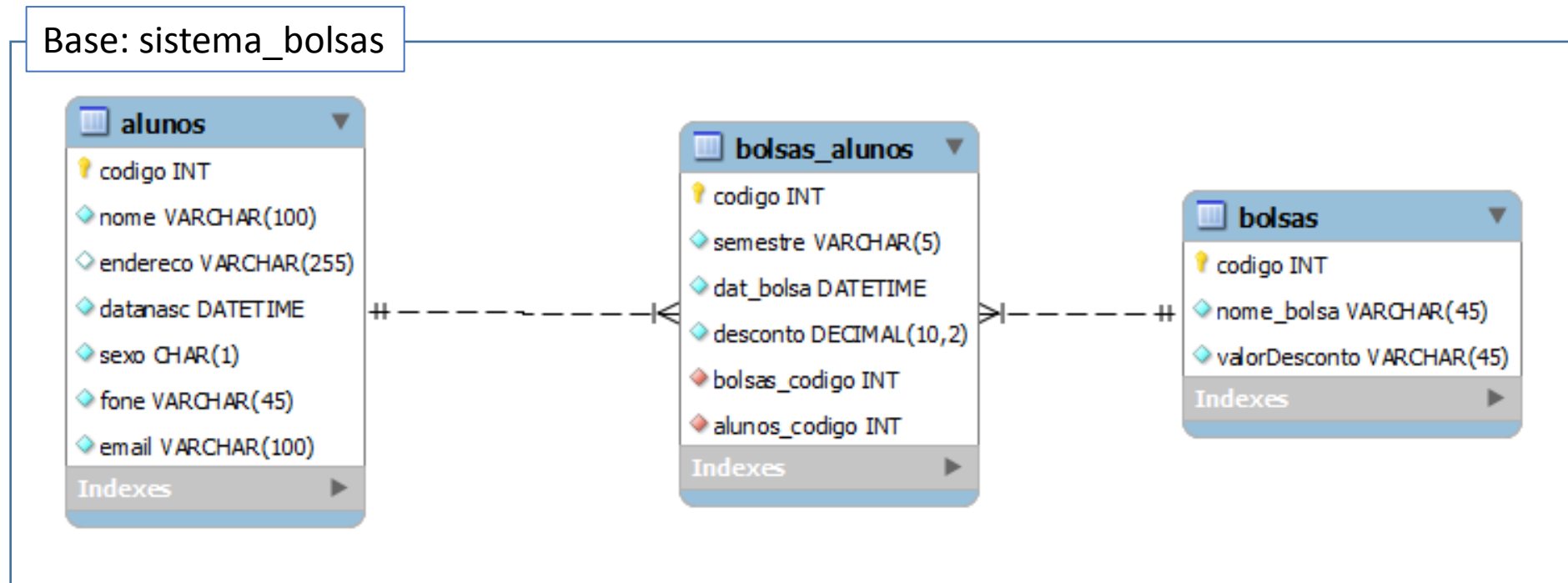
- Processo:
  - No processo de cadastro de bolsas, os alunos precisam realizar uma atualização cadastral, que é feita diretamente no sistema de bolsas.
  - Caso a integração não existisse, a base do ambiente acadêmico não conseguiria enxergar estas atualizações e o contrário também seria verdadeiro.
  - Assim, caso ocorra uma atualização diretamente no ambiente acadêmico, esta não seria replicada no sistema de bolsas sem a integração.

# Integração online

- Pra resolver esse problema, vamos criar uma integração através de *triggers*.
- Os gatilhos (triggers) serão ativados a partir de uma alteração em qualquer uma das duas bases de dados e vai replicar esta alteração no outro banco.

# Integração online

- Banco de dados do sistema de bolsas



# Integração online

- Rotinas que farão a integração entre os dois sistemas:
  - *trigger* que vai escrever os dados, a partir do sistema acadêmico, na base de bolsas.
    - Como o sistema acadêmico é o único responsável pelo cadastro de novos alunos, só teremos as inserções de alunos a partir dele.
    - O módulo de bolsas apenas utiliza as informações disponibilizadas pelo ambiente acadêmico.
    - Além disso, o aluno também pode ter seus dados alterados a partir desse ambiente e as informações devem ser refletidas no cadastro de bolsas.
  - *trigger* para atualizar os dados a partir do sistema de bolsas.
    - após uma informação da tabela de *Alunos* na base *SistemaBolsas* ser alterada, vai executar o comando *update* direto na tabela de *Alunos* que se encontra na base do Sistema Acadêmico.

*trigger* que vai escrever os dados, a partir do sistema acadêmico, na base de bolsas.

```
DELIMITER $$
CREATE TRIGGER trIntegracaoAlunosBolsa
AFTER INSERT
ON Alunos
FOR EACH ROW
BEGIN

    --CASO 1: Inserção
    IF EXISTS (SELECT 1 FROM NEW) AND NOT EXISTS (SELECT 1 FROM OLD)
    BEGIN
        INSERT INTO SistemaBolsas.Alunos(codigo, nome, endereco, datanasc, fone, email)
        SELECT codigo, nome, endereco, datanasc, fone, email
        FROM NEW
    END

    --CASO 2: Alteração
    IF EXISTS (SELECT 1 FROM NEW) AND EXISTS (SELECT 1 FROM OLD)
    BEGIN
        UPDATE SistemaBolsas.Alunos SET nome = I.nome, endereco = I.endereco, datanasc = I.datanasc,
        fone = I.fone, email = I.datanasc
        FROM SistemaBolsas.dbo.Alunos A
        INNER JOIN inserted I ON (A.codigo = I.codigo)
    END

END$
```



*trigger* que vai escrever os dados, a partir do sistema acadêmico, na base de bolsas.

- A trigger deve verificar se ela foi acionada a partir de um insert ou update para inserir ou atualizar os dados dependendo de como a informação foi originada.
  - Para isso, foi feito um teste simples a partir das tabelas NEW e OLD da trigger e verificada a existência de registros utilizando o comando EXISTS.

*trigger* para atualizar os dados a partir do sistema de bolsas

- Após uma informação da tabela de Alunos na base SistemaBolsas ser alterada, vai executar o comando update direto na tabela de Alunos que se encontra na base do Sistema Acadêmico.
- A trigger foi criada apenas para eventos de alteração na tabela Alunos, considerando que o Sistema de Bolsas não faz a inserção de dados, apenas atualiza algumas informações.
- Para este cenário, lembre-se que toda inserção vem do sistema acadêmico

*trigger* para atualizar os dados a partir do sistema de bolsas

```
USE SistemaBolsas;  
CREATE TRIGGER trIntegracaoAlunosAcademico  
AFTER UPDATE  
ON Alunos  
FOR EACH ROW  
BEGIN  
    UPDATE Academico.Alunos SET nome = I.nome, endereco = I.endereco, datanasc = I.datanasc, fone = I.fone, email = I.datanasc  
    FROM Academico.Alunos A INNER JOIN NEW I ON (A.codigo = I.codigo)  
END
```

