

Tecnologia em Sistemas para Internet

Tecnologia em Banco de Dados

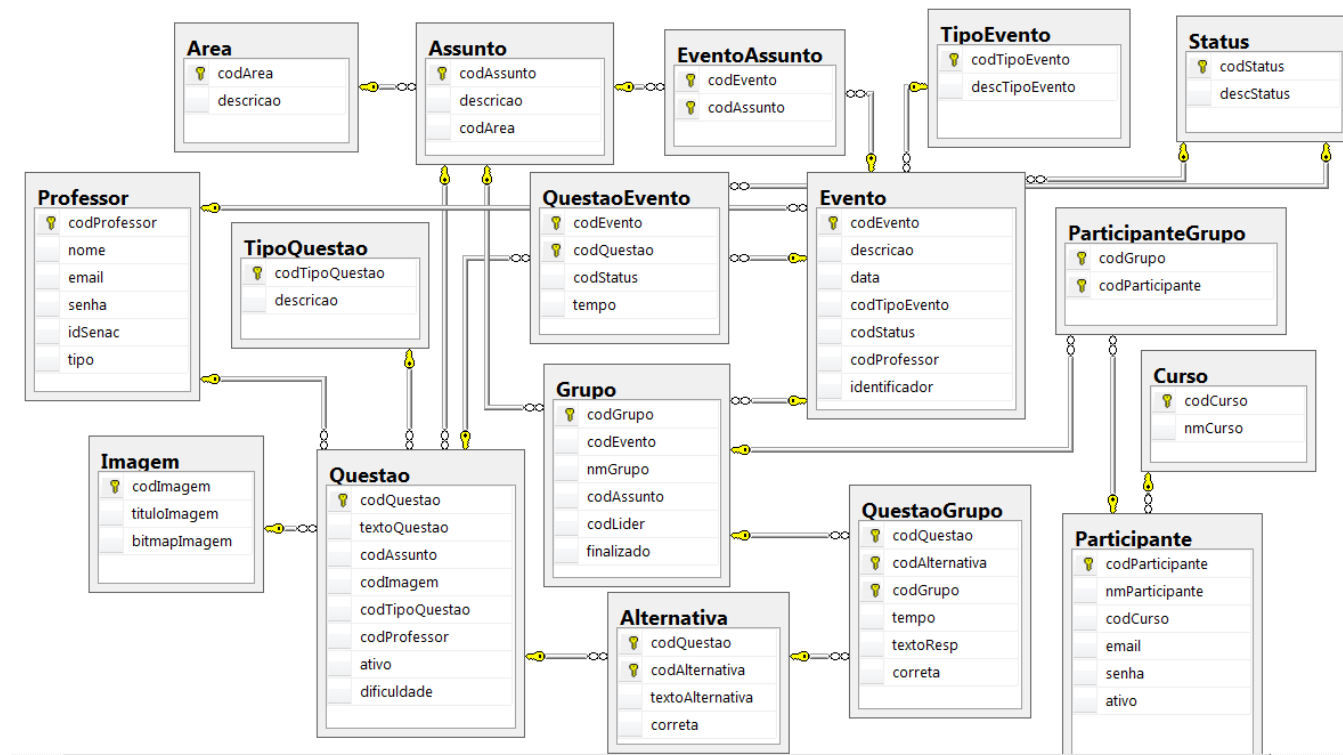
# Manual do Banco de Dados do Projeto Integrador

Por Thiago Claro

Versão 1.1

Mar/2016

Diagrama completo do banco.



Este manual especifica em detalhes o banco de dados que será utilizado no Projeto Integrador, primeiro semestre de 2016.

Para informações sobre as especificações das aplicações a serem desenvolvidas, consulte os outros documentos fornecidos pelos professores de PI.

A leitura deste é obrigatória para todos os componentes do grupo.

## Parte 1 – Tabelas Custodiais.


As tabelas TipoEvento, Status e TipoQuestao são tabelas periféricas que contém dados importantes para as outras tabelas, mais do que para a finalidade do sistema.

Estas tabelas já terão seus dados cadastrados e o usuário TSI que será disponibilizado para a aplicação terá permissões apenas de SELECT nesta tabela.

### 1.1 – Tabela TipoQuestao

Contém os tipos de questões que serão implementadas na aplicação. Poderão ser inclusos novos tipos em uma futura expansão do projeto.

Estrutura da tabela

	Column Name	Data Type	Allow Nulls
	codTipoQuestao	char(1)	<input type="checkbox"/>
	Descricao	varchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>

Conteúdo

codTipoQuest...	Descricao
A	Alternativas
T	Texto Objetivo
V	Verdadeiro ou Falso
NULL	NULL

### 1.2 – Tabela Status

Contém os status possíveis de um evento.


**C** é o primeiro status de um evento, que acabou de ser criado pelo Professor, mas ainda não está recebendo os participantes.

**A** significa que o evento está no lobby, permitindo que os participantes entrem no evento através do identificador e montem seus grupos.

**E** é o status de um evento em execução, ou seja, durante o andamento da partida. O evento só muda para “E” com um comando do professor que criou o evento.

**F** será atribuído aos eventos após encerrados. Ele poderá futuramente ser consultado, mas jamais voltará a um dos status anteriores.

Estrutura da tabela

	Column Name	Data Type	Allow Nulls
	codStatus	char(1)	<input type="checkbox"/>
	descStatus	varchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>

Conteúdo

codStatus	descStatus
C	Criado
A	Aberto
E	Em Execução
F	Fechado
NULL	NULL

### 1.3 – Tabela TipoEvento

Contém a relação de diferentes jogos criados. Neste primeiro momento serão criados 4 jogos diferentes.

Consulte a documentação específica para detalhes sobre cada um dos tipos de evento. Futuramente os valores do atributo descTipoEvento serão alterados, contendo o nome que será definido para cada jogo.

Estrutura da tabela

	Column Name	Data Type	Allow Null
🔑	codTipoEvento	tinyint	<input type="checkbox"/>
	descTipoEvento	varchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>

Conteúdo

codTipoEvento	descTipoEvento
1	Questões Assíncronas
2	Quem Responder Primeiro
3	Todos Respondem Simultâneos
4	Baseado no Master
NULL	NULL

## Parte 2 – Tabelas de cadastros

Estas tabelas terão os módulos de cadastros desenvolvidos neste semestre. O 2º semestre de sistemas para internet fará o cadastro na versão Web em Php e o 3º Semestre do curso de Banco de Dados fará o cadastro para desktop em C#.net

### 2.1 – Tabela Professor

Esta é a estrutura da tabela que irá conter os dados dos professores habilitados a incluir questões e a criar eventos no banco de dados. O banco será criado com um primeiro registro fixo, cujo id é 1, e irá gerar um erro se ocorrer qualquer tentativa de alterar ou excluir este registro.

	Column Name	Data Type	Allow Nulls
🔑	codProfessor	smallint	<input type="checkbox"/>
	nome	varchar(50)	<input type="checkbox"/>
	email	varchar(50)	<input type="checkbox"/>
	senha	varbinary(MAX)	<input checked="" type="checkbox"/>
	idSenac	char(6)	<input checked="" type="checkbox"/>
	tipo	char(1)	<input checked="" type="checkbox"/>

Os atributos deverão conter:

codProfessor – é o código gerado automaticamente (identity)  
 nome – Nome completo do professor  
 email – email do professor, que será usado como login no sistema, por isso é obrigatório  
 senha – senha criptografada. Não devem ser utilizadas funções específicas do php ou .net para criptografar os dados. Consulte o anexo

1 – Criptografia.

IdSenac – para o caso dos professores do Senac, será solicitado seu número de registro, pensando em uma futura integração com outros sistemas.

tipo – dois tipos de professores estão previstos no sistema: 'A' para o professor que faz o papel de gestor Administrador e pode além de incluir questões, também criar eventos, e 'P' para o professor que tem permissão apenas de alimentar o banco de questões.

### 2.2 – Tabela Curso



Apenas o professor com registro na tabela Professor do tipo "A" deve ter acesso para cadastrar, alterar ou remover cursos. Um curso que tenha participantes associados a ele não pode mais ser removido do banco.

	Column Name	Data Type	Allow Nulls
🔑	codCurso	int	<input type="checkbox"/>
	nmCurso	varchar(50)	<input type="checkbox"/>

codCurso - código gerado automaticamente (identity)  
 nmCurso – nome do curso

### 2.3 – Tabela Area

Contém as áreas nas quais serão classificadas as questões. Uma área estará dividida em diversos assuntos, e na verdade a questão estará associada a um determinado assunto que por sua vez pertence à uma área. Exemplos de áreas são: Informática, Contabilidade, Design, Conhecimentos Gerais. Alterações nestas tabelas estão disponíveis apenas para professores do tipo "A".

	Column Name	Data Type	Allow Nulls
	codArea	smallint	<input type="checkbox"/>
	descricao	varchar(50)	<input type="checkbox"/>
			<input type="checkbox"/>


Seus atributos:

codArea - código gerado automaticamente (identity)

descricao – texto com o nome da área.

## 2.4 – Tabela Assunto

Esta tabela contém os assuntos aos quais cada questão se refere. Uma área pode estar relacionada a vários assuntos. Um assunto obrigatoriamente precisa estar associado a uma área. Todos professores cadastrados podem incluir e alterar registros desta tabela.

	Column Name	Data Type	Allow Nulls
	codAssunto	int	<input type="checkbox"/>
	descricao	varchar(30)	<input type="checkbox"/>
	codArea	smallint	<input type="checkbox"/>
			<input type="checkbox"/>

codAssunto – código gerado automaticamente (identity)


descricao – texto definindo o assunto

codArea – chave estrangeira estabelecendo relação com a tabela Área.

## 2.5 – Tabela Questão

Uma das tabelas mais importantes do sistema, ao lado de Evento. Aqui serão armazenadas todas as questões cadastradas pelos professores.

Uma questão deve necessariamente estar associada a um assunto, um tipo de questão e um gestor. Ela pode ou não estar associada a uma imagem.

	Column Name	Data Type	Allow Nulls
	codQuestao	int	<input type="checkbox"/>
	textoQuestao	varchar(300)	<input type="checkbox"/>
	codAssunto	int	<input type="checkbox"/>
	codImagem	int	<input checked="" type="checkbox"/>
	codTipoQuestao	char(1)	<input type="checkbox"/>
	codProfessor	smallint	<input type="checkbox"/>
	ativo	bit	<input type="checkbox"/>
	dificuldade	char(1)	<input type="checkbox"/>

codQuestao - código gerado automaticamente (identity)

textoQuestao – contém o enunciado da questão.

codAssunto – Chave estrangeira relacionando uma questão a um determinado assunto

codImagem - Chave estrangeira opcional relacionando uma questão a uma determinada imagem

codTipoQuestao - Chave estrangeira relacionando uma questão a um determinado tipo

codProfessor - Chave estrangeira relacionando uma questão a um determinado professor

ativo – Indica se a questão está ativa no sistema.

Se a questão já foi utilizada em algum evento, ela não pode ser excluída do banco, e neste caso muda-se o valor deste atributo. 0 (zero) significa que a questão está inativa e 1 (um) significa que está ativa. Este atributo possui o valor padrão 1.

dificuldade – um caractere indicando o nível de dificuldade da questão, sendo:

‘F’ – Fácil


‘M’ – Médio

‘D’ – Difícil

Qualquer outro valor inserido irá gerar um erro do banco de dados.

## 2.5 – Tabela Imagem

Armazena imagens que podem ser associadas a uma questão. Cada questão pode ter uma única imagem. A estrutura do banco de dados permite que mais de uma questão utilize uma mesma imagem, apesar de não existir a previsão de desenvolvermos esta funcionalidade nesta versão do projeto. Não é possível associar uma imagem a uma alternativa.

	Column Name	Data Type	Allow Nulls
	codImagem	int	<input type="checkbox"/>
	tituloImagem	varchar(50)	<input checked="" type="checkbox"/>
	bitmapImagem	varbinary(MAX)	<input type="checkbox"/>

codImagem - código gerado automaticamente (identity)

tituloImagem – Um texto opcional que pode ser associado a uma imagem.

bitmapImagem – campo para o armazenamento dos bytes que compõem a

imagem. A Microsoft sugere a utilização do tipo varbinary no lugar de image, que deve ser removido das versões futuras do SQL Server.

## 2.6 – Tabela Alternativa



Contém as alternativas de cada questão. Esta tabela também armazena as respostas das questões dos tipos “Texto Objetivo” e “Verdadeiro ou Falso”.

Ela irá se comportar de forma diferente para cada tipo de questão relacionada a ela.

Quando a questão é do tipo “Alternativas”, esta tabela contém um registro para cada alternativa cadastrada, e apenas uma delas pode ter o atributo “correta” com valor 1. Ainda neste caso, o atributo “codAlternativa” não é um identity, devendo ser gerado pela aplicação, reiniciando a contagem no 1 para cada questão.

Caso a questão seja do tipo “TextoObjetivo”, ela também pode possuir vários registros associados na tabela alternativa, onde o texto de cada alternativa representa um texto que é aceito como resposta válida para a questão. Neste caso todas as alternativas receberão o valor 1 no atributo “correta”.

Por fim, as questões do tipo “Verdadeiro ou Falso” devem conter um único registro em Alternativa relacionado a elas. Este registro deve ter o codAlternativa = 1 e o atributo “correta” recebe “0” caso a questão seja falsa e “1” caso seja verdadeira.

	Column Name	Data Type	Allow Nulls
	codQuestao	int	<input type="checkbox"/>
	codAlternativa	tinyint	<input type="checkbox"/>
	textoAlternativa	varchar(250)	<input type="checkbox"/>
	correta	bit	<input type="checkbox"/>

codQuestao – chave estrangeira relacionando com a tabela Questao

codAlternativa – compõe a chave primária, sendo um valor único para cada questão.

textoAlternativa – contém o texto da alternativa, ou a resposta da questão.


Correta – contem 1 ou 0, onde 1 representa “correto”.

## 3 – Tabelas Principais

São as tabelas que vão manter o funcionamento dos 4 jogos desenvolvidos, o “core” do sistema. A maioria delas é alimentada pela aplicação desenvolvida pelo terceiro semestre, e lidas pelos jogos do quinto semestre.

### 3.1 – Tabela Participante

Contém os participantes dos eventos, os alunos que vão jogar efetivamente, respondendo as questões. O email do participante deve ser único e será utilizado como login do sistema. Um participante pode ser excluído enquanto não tenha participado de nenhum evento, ou seja, não tenha ingressado em nenhum grupo. Após isso, sua exclusão irá mudar o atributo “Ativo” para 0 (zero). É obrigatório que o participante se identifique em um dos cursos já previamente cadastrados.

	Column Name	Data Type	Allow Nulls
	codParticipante	int	<input type="checkbox"/>
	nmParticipante	varchar(50)	<input type="checkbox"/>
	codCurso	int	<input type="checkbox"/>
	email	varchar(50)	<input type="checkbox"/>
	senha	varbinary(MAX)	<input type="checkbox"/>
	ativo	bit	<input type="checkbox"/>


codParticipante – código gerado pelo sistema (identity).  
nmParticipante – nome completo do participante  
codCurso – Chave estrangeira relacionando o participante a um curso  
email – email a ser utilizado como login, não permite valores duplicados  
senha – hash da senha, deve ser criptografada de acordo com as instruções do anexo I

ativo – contem 1 para os usuários ativos e 0 para os inativos.

### 3.2 – Tabela Evento

Um gestor com tipo = “A” pode criar um registro na tabela evento, que se refere a uma partida de um jogo a ser realizada em uma data futura.

Ao ser criado o evento sempre deve começar com o status = “C”. Um identificador único deve ser gerado para o evento, com uma sequência aleatória de letras de números de 8 caracteres, mas também deve permitir que o professor digite sua própria sequência caso deseje.

	Column Name	Data Type	Allow Nulls
	codEvento	int	<input type="checkbox"/>
	descricao	varchar(80)	<input type="checkbox"/>
	data	smalldatetime	<input type="checkbox"/>
	codTipoEvento	tinyint	<input type="checkbox"/>
	codStatus	char(1)	<input type="checkbox"/>
	codProfessor	smallint	<input type="checkbox"/>
	identificador	varchar(10)	<input type="checkbox"/>



codEvento – código identificador gerado pelo sistema (identity)  
descricao – uma breve descrição sobre o que será o evento  
data – Data e Hora prevista para a realização do evento. (O sistema não irá começar o evento automaticamente, esta é apenas uma previsão).  
codTipoEvento – chave estrangeira da tabela TipoEvento  
codStatus – chave estrangeira da tabela Status

codProfessor – chave estrangeira da tabela Professor , referenciando quem criou o Evento. Apenas este professor poderá iniciar o evento

identificador – identificador que será utilizado pelos participantes para se inscrever no evento. O banco não permite a entrada de um identificador duplicado.

### 3.3 – Tabela QuestaoEvento

Ao criar um evento o gestor deve selecionar as questões que farão parte deste evento, que podem vir a ser selecionadas durante a partida. Estas questões ficam nesta tabela como um “pool” de questões habilitadas a serem sorteadas.

	Column Name	Data Type	Allow Nulls
	codEvento	int	<input type="checkbox"/>
	codQuestao	int	<input type="checkbox"/>
	codStatus	char(1)	<input type="checkbox"/>
	tempo	datetime	<input checked="" type="checkbox"/>

codEvento – Chave estrangeira da tabela Evento  
codQuestao – Chave estrangeira da tabela Questao  
codStatus – Chave estrangeira da tabela Status que registra o status da questão. Ao


ser criado o evento todas as questões são adicionadas com o status “C”.

tempo – timestamp com o momento que a questão foi aberta para respostas. O valor inicial é nulo até que o evento esteja em execução.

### 3.4 – Tabela Grupo

Um participante, após digitar o identificador do evento é levado ao “Lobby” onde poderá se organizar em grupos com os demais participantes. Os grupos criados são armazenados nesta tabela. Cada grupo deve ter um único participante líder, que foi aquele que criou e ingressou primeiramente no grupo. Em alguns tipos de evento, o grupo escolhe um determinado assunto de preferência, conforma a dinâmica do jogo. Neste caso, o atributo codAssunto armazena o código do assunto escolhido.

Nos demais casos esta chave estrangeira fica nula.

	Column Name	Data Type	Allow Nulls
	codGrupo	int	<input type="checkbox"/>
	codEvento	int	<input type="checkbox"/>
	nmGrupo	varchar(20)	<input type="checkbox"/>
	codAssunto	int	<input checked="" type="checkbox"/>
	codLider	int	<input type="checkbox"/>
	finalizado	bit	<input type="checkbox"/>

codGrupo – código gerado pelo sistema (identity)  
codEvento - chave estrangeira da tabela evento, que registra a qual evento este grupo pertence. Grupos não são reaproveitados em novos eventos, devendo ser criado um novo.

nmGrupo – Nome do grupo definido pelo líder no momento da criação.

codAssunto – chave estrangeira da tabela assunto conforme explicado

codLider – chave estrangeira da tabela




participante, relacionando um grupo a um participante que o representará no evento.

finalizado – um indicador que todos os componentes do grupo já estão presentes. O líder do grupo que altera este valor para 1, na aplicação.

### 3.5 – Tabela QuestaoGrupo

Apesar do nome, esta tabela relaciona as alternativas com os grupos, armazenando a resposta dada por um determinado grupo, para uma determinada questão.

Caso a questão seja do tipo “alternativa”, a chave estrangeira vai conter a alternativa escolhida. Caso a questão seja do tipo “texto objetivo”, e a resposta esteja correta, codAlternativa vai apontar para a alternativa cujo texto foi coincidente com o informado, e o campo textoResp armazena o texto da resposta digitada. Caso a questão seja do tipo “Verdadeiro ou Falso”, codalternativa aponta para a única alternativa lá cadastrada, e o textoResp armazena “V” ou “F”. Apesar de ser possível conferir se a resposta está correta através do relacionamento, esta informação é gravada no atributo “correta” desta tabela, permitindo assim que as tabelas Questão e Alternativa sofram alterações no futuro, sem alterar o placar de um evento passado.

	Column Name	Data Type	Allow Nulls
	codQuestao	int	<input type="checkbox"/>
	codAlternativa	tinyint	<input type="checkbox"/>
	codGrupo	int	<input type="checkbox"/>
	tempo	datetime	<input checked="" type="checkbox"/>
	textoResp	varchar(100)	<input checked="" type="checkbox"/>
	correta	bit	<input type="checkbox"/>

codQuestao – Chave estrangeira composta com...

codAlternativa – chave estrangeira da tabela alternativa.

codGrupo – chave estrangeira da tabela Grupo

tempo – tempo que o grupo demorou para submeter a resposta. Pode ser utilizado de forma diferente para cada tipo de evento.

textoResp – Texto informado na resposta



objetiva ou V e F. Deve ficar nulo nas questões de alternativas.

Correta – 1 para o caso de a resposta estar correta e 0 para incorreta.

### 3.6 – Tabela ParticipanteGrupo

Tabela que faz apenas a relação muitos para muitos entre as entidades Participante e Grupo. Indica quais participantes estão compondo um determinado grupo. Um grupo deve ter no mínimo um único participante, caso contrário ele deve deixar de existir. É perfeitamente possível que existam grupos com um único participante no caso de uma atividade individual.





	Column Name	Data Type	Allow Nulls
	codGrupo	int	<input type="checkbox"/>
	codParticipante	int	<input type="checkbox"/>

codGrupo – Chave estrangeira da tabela Grupo  
codParticipante – chave estrangeira da tabela Participante

### 3.7 – Tabela EventoAssunto

Outra tabela simples, “explosão” muitos para muitos entre as entidades Evento e Assunto, pois durante a criação de um evento, o professor deve delimitar qual, ou quais assuntos ele abrange.

	Column Name	Data Type	Allow Nulls
	codEvento	int	<input type="checkbox"/>
	codAssunto	int	<input type="checkbox"/>

codEvento – Chave estrangeira da tabela Evento  
codAssunto – chave estrangeira da tabela assunto

## ANEXO 1 – A criptografia das senhas

Para armazenar as senhas nas tabelas Professor e Participante, utilizaremos um esquema de Hash, ao invés de armazenar a senha como texto. Por diversas razões que são abordadas na disciplina de programação com recursos de segurança do quinto semestre, este é um método muito mais seguro para armazenar senhas.

Como teremos diversas linguagens acessando o mesmo banco de dados, a forma mais simples de fazermos isto, será realizando a operação geração do hash da senha no próprio banco de dados.

Pela própria característica do algoritmo de hash, ele não pode ser descriptografado, portanto, para validar se a senha informada login coincide com a armazenada na tabela, deve-se gerar um novo hash da senha informada e comparar com o hash armazenado.

Para gerar este hash, utilizaremos a função **HASHBYTES** do SQL Server, que possui a seguinte sintaxe:

```
HASHBYTES ( '<algorithm>', { @input | 'input' } )
```

```
<algorithm>::= MD2 | MD4 | MD5 | SHA | SHA1 | SHA2_256 | SHA2_512
```

Nesta aplicação utilizaremos o algoritmo SHA1. Sim, sabemos que não é o mais seguro disponível, e que pode ser quebrado se você dispor de um grande poder computacional, dinheiro e tempo. Também sabemos que estamos desenvolvendo um Quiz e não um HomeBanking, portanto é exatamente este que utilizaremos.

Seguem exemplos de utilização da função HashBytes:

b) Gravando um registro, inserindo o hash na tabela:

```
create table teste
(id int primary key, email varchar(50), senha varbinary(max))

declare @senha varchar(40)
set @senha = 'pipoca'

insert into teste
values (3, 'teste@gmail.com', HASHBYTES('SHA1', @senha) )
```

b) Durante o logon, verificando se o usuário digitou a senha correta:

```
select id from teste where email = 'teste@gmail.com' and senha = HASHBYTES('SHA1', 'pipoca')
```

(Se retornar um registro, a senha está correta).

Note que a função funciona tanto através de uma variável quanto com texto direto. Para mais informações sobre a utilização desta função consulte seu professor de PI ou estes sites:

<https://www.mssqltips.com/sqlservertip/2988/understanding-the-sql-server-hashbytes-hashing-algorithms/>  
[https://msdn.microsoft.com/en-us/library/ms174415\(v=sql.110\).aspx](https://msdn.microsoft.com/en-us/library/ms174415(v=sql.110).aspx)

## Anexo 2 – Instruções de conexão com o banco

O banco de dados estará disponível para acesso a partir do dia 07/03, no seguinte endereço:

Servidor: koo2dzw5dy.database.windows.net

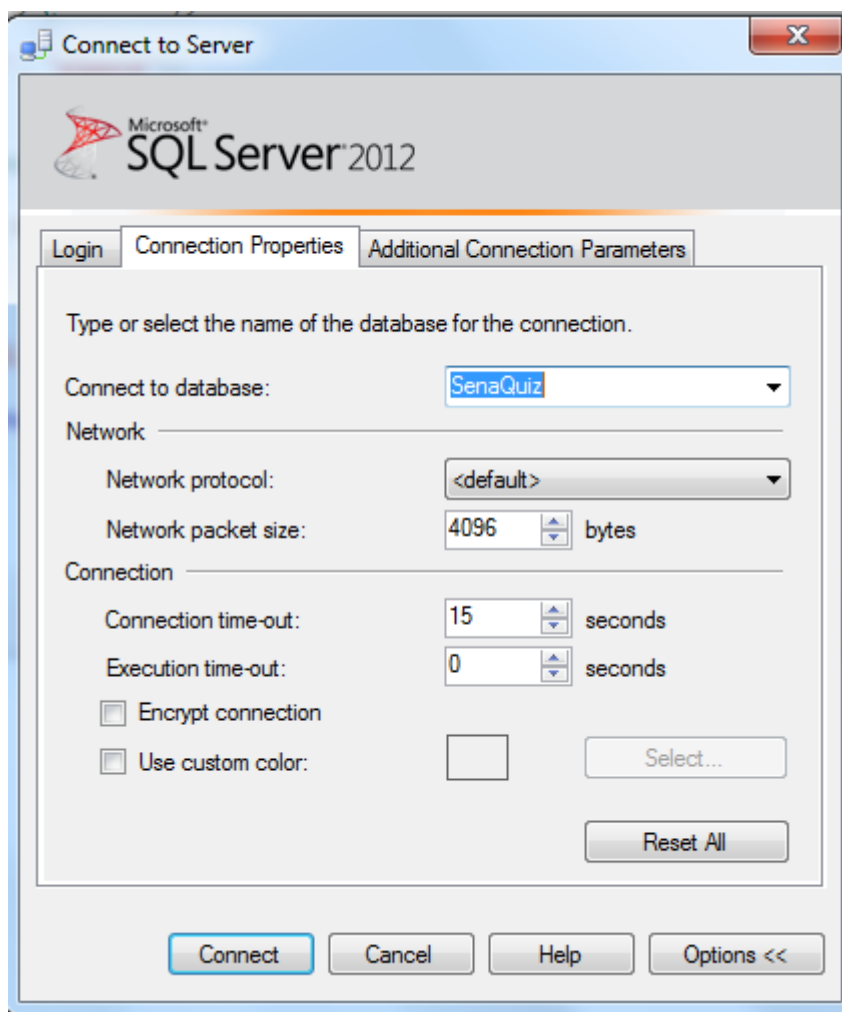
Porta: 1433

Database: SenaQuiz

Login: TSI

Senha: SistemasInternet123

Como o usuário TSI não tem acesso ao banco master do servidor, é necessário especificar o nome do banco de dados na tela de conexão do Management Studio, nas opções avançadas de conexão, conforme imagem abaixo.



### Anexo 3 – Procedures comuns

Algumas rotinas são comuns a vários grupos, e podem ser otimizadas se transferidas para um procedimento armazenado no SGBD. Aqui será atualizada a relação de procedimentos criados com esta finalidade, que devem ser utilizados pelos grupos.

#### 1 – pr\_ExcluirGrupo (@codGrupo int)

Este procedimento exclui todos os registros das tabelas ParticipanteGrupo e Grupo, de acordo com o valor do @codGrupo passado.