

|  |
| --- |
| Complementos de Bases de Dados 2021/2022  Licenciatura em Engª. Informática |
| Relatório Técnico |

Turma: 2ª L\_EI\_SW04

Horário de Laboratório: Quinta-feira 10:30

Docente: Gabriel Pestana

Grupo

Nº202001990, Daniel Baptista  
Nº202001553, Rafael Silva

# Sumário Executivo

Este relatório tem como objetivo a documentação do desenvolvimento do projeto, no âmbito da UC de Complementos de Base de Dados.

Este projeto tinha como objetivo o restruturamento de uma base de dados já existente e com dados, a qual se dirigia a um sistema de gestão de uma escola, este restruturamento resultou numa nova base de dados melhorada com base na antiga base de dados.

Seria necessário então que houvesse um sistema de autenticação do estudante e do encarregado de educação que tinha com fim guardar os dados do estudante permitindo quer o estudante quer o encarregado de educação ver as notas obtidas entre outras coisas. No âmbito de melhoramento semântico alterou-se o nome dos atributos da base de dados antiga para ser mais intuitivo no caso de se ter que fazer uma alteração qualquer na base de dados, e foi também implementado um sistema de Logs para guardar os dados relativos as notas e inscrições de anos anteriores.

# Especificação de Requisitos

|  |  |  |
| --- | --- | --- |
| ID | Descrição | Implementado  (S/N) |
| RF01 | O sistema deverá implementar dados multi-idioma | S |
| RF02 | O sistema deverá permitir registar cada aluno nas disciplinas para um determinado ano letivo | S |
| RF03 | O sistema deverá permitir fazer a gestão das notas | S |
| RF04 | O sistema deverá permitir fazer a gestão dos anos letivos, sabendo que só um pode estar em aberto | S |
| RF05 | O sistema deverá reter informação de anos letivos passados relativamente as notas e as inscrições dos alunos | S |
| RF06 | O sistema deverá permitir que cada aluno tenha acesso as suas notas, desde que este esteja registado no sistema | S |
| RF07 | O sistema deverá permitir que o encarregado de educação possa visualizar as notas do seu educando | S |
| RF08 | O sistema deverá permitir a autenticação ao sistema por parte do aluno e do encarregado de educação através do email e da password | S |
| RF09 | O sistema deverá permitir fazer a recuperação da password, desde que o aluno ou o encarregado de educação dentro de 1h troquem a password e coloquem o codigo solicitado | S |
| RF10 | O sistema deverá permitir fazer a alteração da password do utilizador desde que este indique a password antiga corretamente e a nova password 2 vezes | S |
| RF11 | O sistema deverá notificar o utilizador, sempre que este mude a password, por email | S |

# Alterações/Melhorias à Fase 1

Nesta 2ª fase do projeto foram melhoradas quase todas as partes apresentadas na 1ª fase, em maior detalhe as alterações foram:

* Foi criada a entidade escola que se relaciona com o aluno na forma de matrícula, pois certos atributos que estavam previamente no aluno não faziam sentido.
* Criadas as entidades FamilySize, FatherJob e MotherJob pois fica mais fácil verificar se os dados inseridos estão corretos.
* Implementação do sistema multi-idioma não implementado na fase anterior.
* Implementado as procedures para abrir e fechar o ano letivo.

# Modelo Relacional (Modelo de dados)

## Diagrama do Modelo Entidade Relação

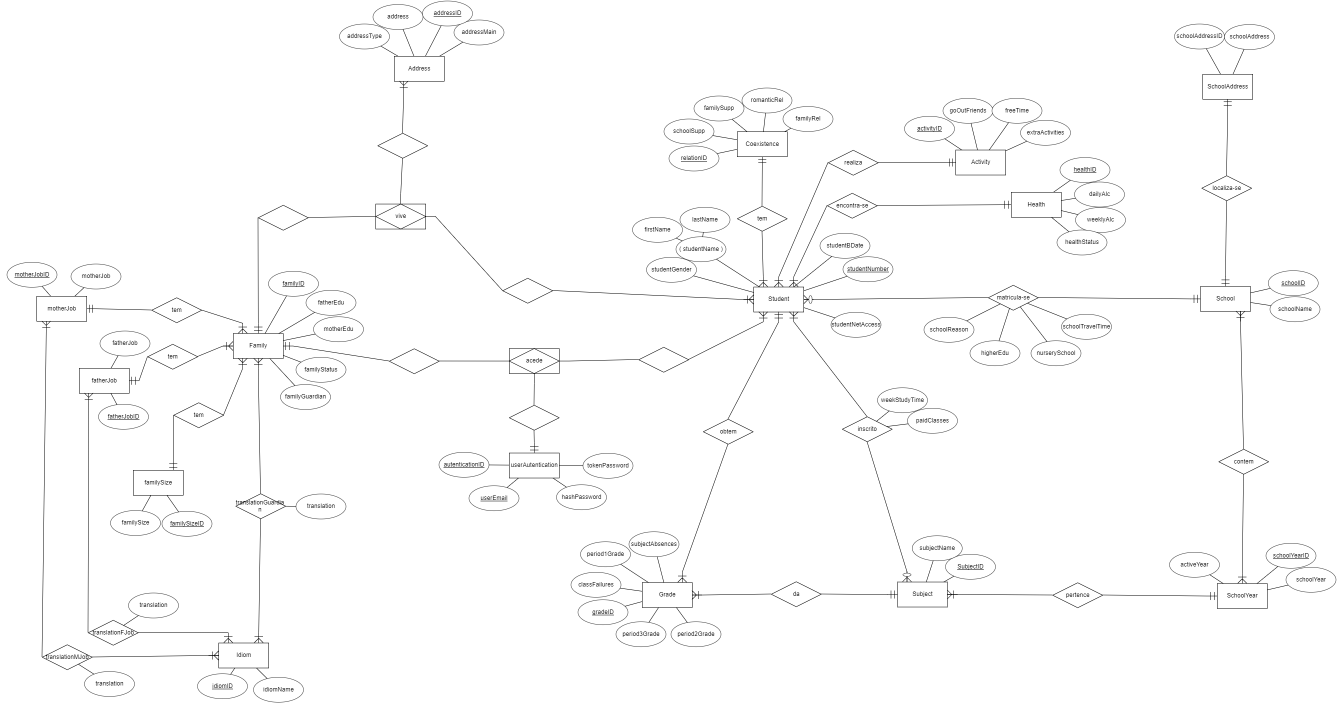


Figura 1. Diagrama Entidade Relacionamento

## Diagrama do Modelo Entidade Relação

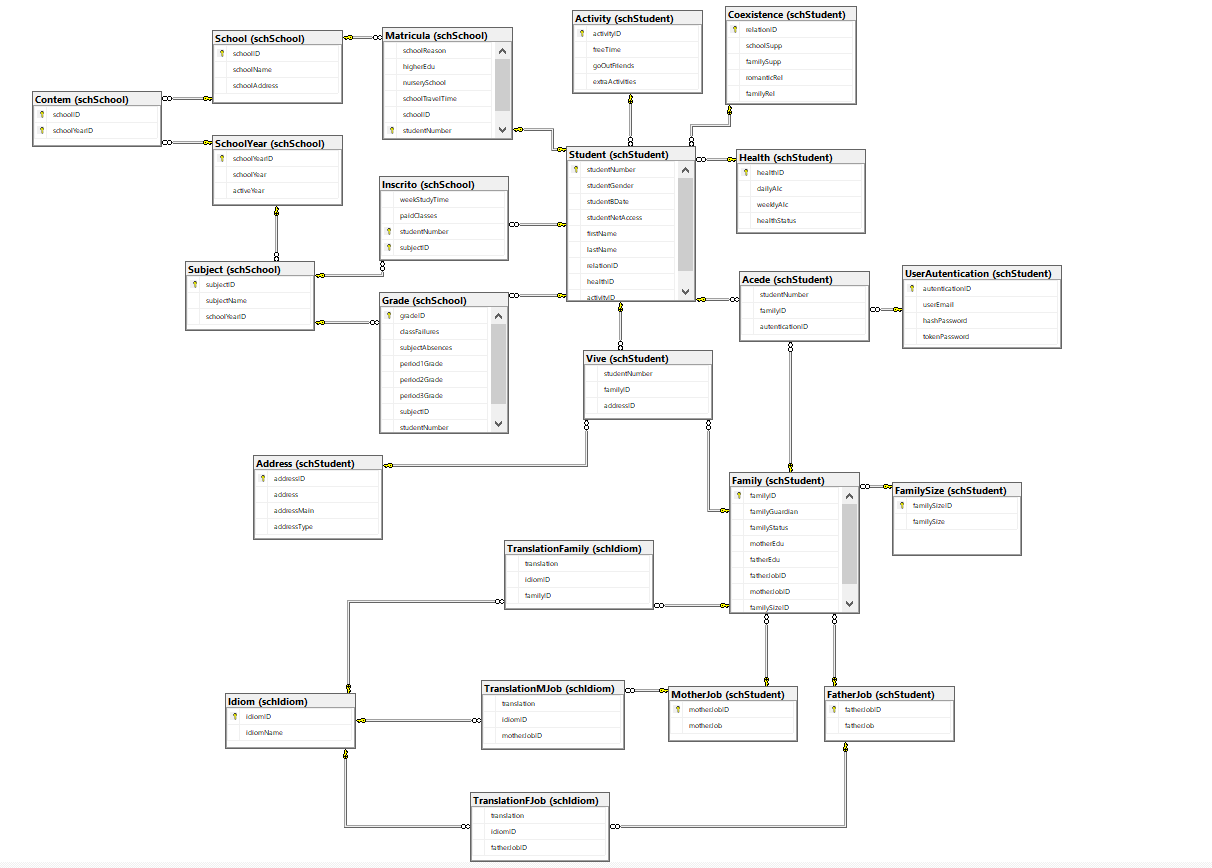


Figura 2. Modelo Relacional

# Dimensionamento do Layout

|  |  |  |
| --- | --- | --- |
| Nome Tabela | Dimensão do Registo | Nº de Registos (inicial/final) |
| Activity | 7 bytes | Vamos ter 50 registos no total.  7 \* 50 = 350 bytes |
| Health | 7 bytes | Vamos ter 125 registos no total.  7 \* 125 = 875 bytes |
| Coexistence | 8 bytes | Vamos ter 40 registos no total.  8 \* 40 = 320 bytes |
| Address | 262 bytes | Idealmente com os dados de teste teríamos 43947 moradas.  43947 \* 262 = 11 514 114 bytes |
| Family | 59 bytes | Idealmente com os dados de teste teríamos 43947 familias.  43947 \* 59 = 2 592 873 bytes |
| UserAutentication | 364 bytes | Idealmente com os dados de teste e de import teríamos 87894 registos(43947 alunos e 43947 encarregados).  87894 \* 364 = 31 993 416 bytes |
| Student | 101 bytes | Ao início teremos 1947 registos + 42000 com dados de teste com 600-800 estudantes por ano(60 anos).  43947 \* 101 = 4 438 647 bytes |
| SchoolYear | 9 bytes | Teremos 60 registos(1960 a 2020).  60 \* 9 = 540 bytes |
| Grade | 38 bytes | Num dado ano haverá uma média de 6300 registos(12 disciplinas e 700 alunos em dados de testes) sem contar com notas de alunos chumbados.  8400 \* 38 = 319 400 bytes |
| Subject | 48 bytes | Teremos 720 registos(12 disciplinas por ano, 60 anos letivos).  720 \* 48 = 34560 bytes |
| School | 244 bytes | Teremos 6 escolas, logo 6 \* 244 = 1464 bytes |
| Matricula | 51 bytes | Teremos 43947 registos(número total de alunos).  43947 \* 51 = 2 241 297 bytes |
| Contem | 8 bytes | Teremos 360 registos pois temos 6 escolas e 60 anos letivos.  360 \* 8 = 2880 bytes |
| Inscrito | 10 bytes | Num dado ano haverá uma média de 8400 registos(12 disciplinas e 700 alunos em dados de testes) sem contar com notas de alunos chumbados.  8400 \* 10 = 84 000 bytes |
| Vive | 12 bytes | Idealmente com os dados de teste e de import teríamos 43947 registos.  43947 \* 12 = 527 364 bytes |
| Acede | 12 bytes | Idealmente com os dados de teste e de import teríamos 87894 registos.  87894 \* 12 = 1 054 728 bytes |
| FamilySize | 12 bytes | Teremos cerca de 20 registos por estimativa.  20 \* 12 = 240 bytes |
| FatherJob | 84 bytes | Teremos cerca de 50 registos por estimativa.  50 \* 84 = 4200 bytes |
| MotherJob | 84 bytes | Teremos cerca de 50 registos por estimativa.  50 \* 84 = 4200 bytes |
| ClosedGrades | 46 bytes | Com os dados de teste haverá cerca de 395523 registos(total Alunos\*número disciplinas) sem contar com repetentes.  527364 \* 46 = 24 258 744 bytes |
| ClosedInscrito | 22 bytes | Com os dados de teste haverá cerca de 395523 registos(total Alunos\*número disciplinas) sem contar com repetentes.  527364 \* 22 = 11 602 008 bytes |
| LogsPassword | 101 bytes | Por estimativa se todos os utilizadores apenas fizerem 1 mudança de password teríamos 87894 registos.  87894 \* 101 = 8 877 294 bytes |
| Idiom | 34 | Teremos 2 línguas logo 2 registos.  2 \* 34 = 68 bytes |
| TranslationFamily | 48 | Teremos 43947 familias(encarregados) e 2 linguas logo 87894 registos.  87894 \* 48 = 4 218 912 bytes |
| TranslationFJob | 48 | Teremos por estimativa, cerca de 50 trabalhos e 2 línguas, logo 100 registos.  100 \* 48 = 4800 bytes |
| TranslationMJob | 48 | Teremos por estimativa, cerca de 50 trabalhos e 2 línguas, logo 100 registos.  100 \* 48 = 4800 bytes |

Em anexo vai um ficheiro para perceber melhor as contas feitas para calcular o tamanho que cada registo ocupa (Ver anexo 1).

Especificação dos FIlegroups

|  |  |  |
| --- | --- | --- |
| Nome Filegroup | Tabelas associadas | Parâmetros |
| StudentFG | Student, Activity, Health, Coexistence, UserAutentication, Acede, Family, Address, Vive, FamilySize, FatherJob e MotherJob | Dimensão inicial: 100MB  Dimensão final: 300MB  Taxa de Crescimento: 50MB |
| SchoolFG | Subject, SchoolYear, Grade, Inscrito, School, Matricula e Contem | Dimensão inicial: 5MB  Dimensão final: 15MB  Taxa de Crescimento: 2MB |
| LogsFG | ClosedGrade, ClosedInscrito e LogsPassword | Dimensão inicial: 100MB  Dimensão final: 300MB  Taxa de Crescimento: 50MB |
| IdiomFG | Idiom, TranslationFamily, TranslationFJob, TranslationMJob | Dimensão inicial: 6MB  Dimensão final: 18MB  Taxa de Crescimento: 3MB |

# Schemas

|  |  |
| --- | --- |
| Nome | Descrição |
| Student | Este esquema tem como objetivo organizar todos os dados que tenham a ver com o estudante, ou seja, a autenticação, a familia, as atividades, etc... |
| School | Este esquema tem como objetivo organizar os dados relativos a escola, ou seja, a própria escola, as notas, as disciplinas, o ano letivo e os alunos inscritos. |
| Logs | Este esquema tem como objetivo agrupar os dados antigos da base de dados que serão as notas e as inscrições. |
| Idiom | Este esquema tem como objetivo agrupar guardar os dados relativos a tradução de certos valores para outras línguas. |

# Views

|  |  |
| --- | --- |
| Nome | Descrição |
| view\_studentsInformationGP e MS | Estas views permite obter uma lista de informação sobre os alunos da escola GP ou MS |
| view\_studentsHealthStatusGP e MS | Estas views permite obter uma lista do estado de saude de cada aluno que esteja inscrito na escola GP ou MS |
| view\_studentActivitiesGP e MS | Estas views permite obter uma lista de todas as atividades de cada aluno que esteja inscrito na escola GP ou MS |
| view\_studentGradesGP e MS | Estas views permite obter uma lista das notas dos alunos da escola GP ou da escola MS |
| view\_studentInscritosGP e MS | Estas views permite obter uma lista dos alunos inscritos na escola GP ou na escola MS |
| view\_studentFamilyInformationGP e MS | Estas views mostra uma os dados familiares dos estudantes da escola GP ou da escola MS |
| view\_schoolYearInformationGP e MS | Estas views mostra uma lista com os dados do Ano letivo escolar da escola GP ou MS |
| view\_logGradesGP e MS | Estas views mostra uma lista com os dados de histórico de notas dos estudantes da escola GP ou MS |
| view\_logInscritosGP e MS | Estas views mostra uma lista com os dados de histórico de inscrições dos estudantes da escola GP ou MS |

# Functions

|  |  |  |  |
| --- | --- | --- | --- |
| Nome | Tipo | Atributos | Descrição |
| fnEncriptarMorada | VARBINARY(256) | @address VARCHAR(128) | Permite fazer a encriptação da morada do utilizador de acordo com uma chave especifica. |
| fnCodificaPassword | VARCHAR(128) | @password VARCHAR(20) | Permite fazer a codificação da password do utilizador |
| fnMakeStudentNumber | INT |  | Ira criar um número de aluno de acordo com as seguintes regras: Ano letivo corrente + número de 5 digitos |
| fnFindCoexistenceID | INT | @schoolSupp CHAR @familySupp CHAR @romanticRel CHAR @familyRel TINYINT | Devolve o ID corresponde aos dados de relações fornecidos. |
| fnFindActivityID | INT | @freeTime TINYINT @goOutFriends TINYINT @extraActivities CHAR | Devolve o ID corresponde aos dados de atividades fornecidos. |
| fnFindHealthID | INT | @dailyAlc TINYINT @weeklyAlc TINYINT @healthStatus TINYINT | Devolve o ID corresponde aos dados de saúde fornecidos. |
| fnFindMotherJobID | INT | @motherJob NVARCHAR(40) | Devolve o ID corresponde ao trabalho da mãe fornecido. |
| fnFindFatherJobID | INT | @fatherJob NVARCHAR(40) | Devolve o ID corresponde ao trabalho do pai fornecido. |
| fnFindFamilySizeID | INT | @familySize CHAR(4) | Devolve o ID corresponde ao tamanho da família fornecido. |
| fnFindSchoolID | INT | @schoolName VARCHAR(20) | Devolve o ID corresponde ao nome da escola fornecido. |
| fnFindSubjectByName | INT | @subjectName NVARCHAR(20) | Devolve o ID corresponde ao nome da disciplina fornecido. |
| fnBuscarAnoAberto | INT |  | Devolve o ano letivo que se encontra ativo. |
| fnAutenticarUtilizador | BIT | @email VARCHAR(50) @password VARCHAR(128) | Permite fazer a autenticação do utilizador |
| fnCalcularNotaFinalAluno | INT | @studentNumber INT @subjectID INT | Dados um número de aluno e o ID da disciplina, vai calcular a nota final do aluno. |
| fnCalcularTaxaCrescimento | FLOAT | @schoolYear INT | Dado um ano letivo, irá calcular a taxa de crescimento comparado com o ano anterior, em função da quantidade de alunos. |
| fnCalcularPercentagemNotaMaiorQue15 | FLOAT | @schoolYear INT | Dado um ano letivo, irá calcular a percentagem de alunos com uma nota final maior ou igual a 15. |

# Stored procedures

|  |  |  |
| --- | --- | --- |
| Nome | Atributos | Descrição |
| spOpenKeys |  | Abre a chave para uso na encriptação ou decriptação da morada do utilizador. |
| spAddSubjectToYear | @subjectName NVARCHAR(20)  @schoolYear INT | Adiciona uma dada disciplina ao ano letivo dado. |
| spRegistarUtilizadorAutenticacao | @password VARCHAR(20)  @id INT | Permite registar o utilizador no sistema |
| spVerNotasAluno | @email VARCHAR(50), @password VARCHAR(20) | Permite visualizar as notas do Aluno |
| spIncreverAlunos | @subjectName NVARCHAR(20)  @studentNumber INT, @paidClasses CHAR(1), @weekStudyTime TINYINT | Permite inscrever os Alunos nas disciplinas |
| spMudarPassword | @email VARCHAR(50), @password VARCHAR(20), @newPassword VARCHAR(128), @newPassword2 VARCHAR(128) | Permite fazer a mudança da password do utilizador |
| spCriarTokenPassword | @email VARCHAR(50) | Permite fazer a criação do token para mudança da password do utilizador |
| spMudarPasswordToken | @email VARCHAR(50)  @token INT  @newPassword VARCHAR(128)  @newPassword2 VARCHAR(128) | Permite usar o token criado para fazer a mudança da password |
| spLancarNotas | @studentNumber INT  @subjectName INT  @grade1 FLOAT  @grade2 FLOAT  @grade3 FLOAT  @classFailures TINYINT  @subjectAbsences TINYINT | Permite criar um registo para as notas do aluno fornecido. |
| spMatricularAluno | @schoolReason VARCHAR(20)  @higherEdu CHAR(1)  @nurserySchool CHAR(1)  @schoolTravelTime TINYINT  @schoolname VARCHAR(50)  @studentNumber INT | Faz a matricula de um aluno na escola escolhida. |
| spFecharAno | @schoolYear INT | Fecha o ano letivo escolhido, caso este esteja ativo. |
| spAbrirAno | @schoolYear INT | Abre o ano letivo escolhido, caso não haja outro aberto. |
| spInscreverAlunosChumbados |  | Irá inscrever os alunos chumbados no ano letivo anterior nas mesmas disciplinas, no novo ano letivo, caso estas disciplinas existam. |
| spTestGenerateFailedStudentGrades |  | Irá gerar notas aleatórias para os alunos chumbados. |
| spEnrollStudent | @studentNumber INT | Dado um número de estudante, ira o inscrever em todas as disciplinas do ano corrente e gera notas para todas elas. |
| spMassCreateStudents |  | Irá criar 600 a 800 alunos com dados de teste e os inscreve nas disciplinas. |
| spCreateSchools |  | Irá criar as escolas usadas nos dados de teste. |
| spTestSubjects |  | Irá criar as disciplinas de teste para o ano letivo corrente. |
| spMakeTestYears |  | Irá gerar dados de teste na base de dados desde os anos 1960 ate 2020. |

# Triggers

|  |  |  |  |
| --- | --- | --- | --- |
| Nome | Tipo | Tabela | Descrição |
| schSchool.trg\_backup\_grades | AFTER INSERT | schSchool.Grades | Guarda as notas na tabela schLogs.ClosedGrades após estas serem inseridas |
| schSchool.trg\_backup\_inscritos | AFTER INSERT | schSchool.Inscrito | Guarda as inscrições dos alunos nas disciplinas na tabela schLogs.ClosedInscrito após estes se inscreverem |
| schSchool.trg\_change\_activeYear | AFTER INSERT | schSchool.SchoolYear | Vai alterar o valor do atributo activeYear do ano letivo prévio para 0 de modo a apenas haver 1 ano letivo aberto. |
| schStudent.trg\_email\_user\_password\_change | AFTER UPDATE | schStudent.UserAutentication | Irá fazer a simulação do envio de um email a um utilizador sempre que este mude a palavra-passe. |

# Consultas

## Verificação da conformidade dos dados

De modo a verificar a conformidade dos dados foram feitas as várias queries, a primeira delas sendo uma query que mostra o total de alunos na base de dados, estes ordenados pela sua respetiva escola.

A próxima consulta calcula a media do ano letivo por escola fazendo a media da soma as notas dos períodos e a divisão do resultado da média pelo número de períodos, depois agrupa as medias por escola e por cada ano letivo.

A última consulta a ser criada mostra a media de notas por período, estas sendo agrupadas por escola e por ano letivo.

# Índices

## Views

|  |  |
| --- | --- |
| Nome | Descrição |
| view\_taxaCrescimento | Mostra a taxa de crescimento de cada ano, face ao ano anterior, em função do número de alunos. |
| view\_AlunosNotaMaiorQue15 | Mostra a percentagem de alunos com media maior ou igual a 15 em cada ano letivo. |
| view\_EscolaMelhorMediaAnual | Mostra a escola com a melhor media final para cada ano letivo. |

## Índices

|  |  |  |
| --- | --- | --- |
| Designação | Tabela | Justificação/Consultas |
| indexCrescimento | ClosedInscritos | Antes:    Depois:    Colunas Indexadas: subjectID |
| indexNota15 | ClosedGrade | Antes:    Depois:    Colunas Indexadas: subjectID |
| indexNota15Recom | ClosedGrade | Antes:    Depois:    Colunas Indexadas: subjectID  Colunas Incluidas: studentNumber |
|  |  | Para a view\_EscolaMelhorMediaAnual não foi criado um índice, pois já demonstrava resultados rápidos e o SQL Tuning Advisor também não recomendou nenhum índice. |

## Otimização e Execução de Consultas

- Sobre a base de dados original (não normalizada).

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

Custo de 0,095(58% da query).

- Sobre a base de dados otimizada (normalizada) sem índices.

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com mesa

Descrição gerada automaticamente

Custo de 0.0509(13% da query)

- Sobre a base de dados otimizada (normalizada) com índices.

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com mesa

Descrição gerada automaticamente

# Backup e Recuperação

Foi decidido que inicialmente quando há abertura do novo ano letivo que significa que irá haver inscrição de vários alunos, tendo este ponto podemos calcular que irá haver uma grande afluência de dados nas tabelas para a inscrição dos estudantes, daí pensou-se que seria melhor para não perder qualquer dado do aluno, diariamente até finalizar-se as inscrições dos alunos a noite fazer um backup completo da base de dados.

Quando então o ano letivo começa depois de todos se inscreverem visto que serão poucas as alterações as tabelas, decidiu-se fazer uma política de 2 backups diferenciais diariamente, um na parte da manha e o outro seria feito a noite, e ao fim de cada mês fazer-se um backup completo para tentar-se minimizar qualquer perda de dados que aconteça.

Após a terminação do ano letivo voltamos a adotar a política de backups completos diariamente visto que serão feitas várias alterações nas tabelas da base de dados, ou seja, irá haver uma grande afluência de dados dentro da base de dados.

Como as tabelas Activity, Health, Coexistence, Idiom, TranslationFJob, TranslationMJob, TranslationFamily, FamilySize, MotherJob e FatherJob são tabelas que não sofrem quase alterações o que poder-se-ia fazer era passá-las para um FileGroup de read-only, pois esta informação é excluída dos backups o que faz com que o tamanho e o tempo sejam menores do que esta a incluir toda a informação, ou seja, usaríamos o backup parcial para fazer a otimização do plano de backups da base de dados.

# Segurança e Controlo de Acessos

## Níveis de acesso à informação

No âmbito do projeto foram definidos 5 roles:

* Administrador, que tem acesso a toda a informação da base de dados.
* UtilizadorGP, que tem acesso a informação da escola GP, ou seja, notas, inscrições e a sua informação como aluno, apenas em modo de consulta através de views.
* UtilizadorMS, que tem acesso a informação da escola MS, ou seja, notas, inscrições e a sua informação como aluno, apenas em modo de consulta através de views.
* EscolaGP, que tem acesso a informação da escola GP e dos seus alunos matriculados, quer como consulta ou para alterar dados. Pode também visualizar e alterar dados de histórico, mas não os pode eliminar.
* EscolaMS, que tem acesso a informação da escola MS e dos seus alunos matriculados, quer como consulta ou para alterar dados. Pode também visualizar e alterar dados de histórico, mas não os pode eliminar.

## Encriptação

Para a morada do utilizador decidiu-se usar encriptação pois será necessário fazer a decriptação da morada para que esta possa ser mostrada ao utilizador caso este deseje ou para uso em funções.

O hashing foi utilizado para esconder a password do utilizador pois a única coisa feita com a password é a sua comparação com uma password fornecida para a autenticação, para qual não nos interessa saber a password não encriptada.

# Controlo de Concorrência

Os níveis de isolamento adotados no projeto foi o serializable, como é um nível que impede qualquer falha de isolamento achamos que fosse o mais indicado para não termos qualquer incoerência a nível de transações.

# MongoDB

*Descrição do código implementado em MongoDB, justificando as opções tomadas.*

# Descrição da Demonstração

## Requisitos implementados

|  |  |
| --- | --- |
| ID | Solução |
| RF01 | Schema Idiom e as suas respetivas tabelas. |
| RF02 | SP\_spInscreverAlunos |
| RF03 | SP\_spLancarNotas |
| RF04 | Atributo activeYear na tabela SchoolYear e Trigger trg\_change\_activeYear |
| RF05 | Triggers schSchool.trg\_backup\_grades e schSchool.trg\_backup\_inscritos |
| RF06 | SP\_spRegistarUtilizadorAutenticacao e SP\_spVerNotasAluno |
| RF07 | SP\_spRegistarUtilizadorAutenticacao e SP\_spVerNotasAluno |
| RF08 | Function fnAutenticarUtilizador |
| RF09 | SP\_spCriarTokenPassword e SP\_spMudarPasswordToken e atributo tokenPassword na tabela userAutentication |
| RF10 | SP\_spMudarPassword |
| RF11 | Trigger schStudent.trg\_email\_user\_password\_change e tabela EmailPW |

## Scripts de demonstração

|  |  |  |
| --- | --- | --- |
| Número | Nome do ficheiro | Descrição |
| 1 | Proj\_BD.sql | Criação da base de dados. |
| 2 | Triggers.sql | Criação dos triggers(Ignorar erros iniciais, isso acontence pq sql server não suporta “create or alter trigger”) |
| 3 | Functions.sql | Criação das funções usadas na base de dados. |
| 4 | StoredProcedures.sql | Criação das procedures usadas na base de dados. |
| 5 | Encriptação.sql | Criação da chave e certificado para encriptação. |
| 6 | ViewsEscolaGP.sql | Criação das views relacionadas a escola GP. |
| 7 | ViewsEscolaMS.sql | Criação das views relacionadas a escola MS. |
| 8 | Roles.sql | Criação dos roles, logins e users na base de dados. |
| 9 | Backup.sql | Definição da estratégia de backup para a base de dados normalizada. |
| 10 | ImportBDNova.sql | Importar para a base de dados normalizada os dados pertencentes a base de dados antiga. |
| 11 | ImportBDVelha.sql | Criação da base de dados original(não normalizada) |
| 12 | Queries.sql | Execução de queries para verificar a conformidade dos dados da base de dados normalizada. |
| 13 | ProceduresIndices.sql | Criação e execução das procedures relacionadas com o carregamento da base de dados com dados de teste. |
| 14 | ViewsIndices.sql | Criação de views e índices com base nos dados de teste. |
| 15 | BDProceduresTests.sql | Testagem das funcionalidades da base de dados normalizada relacionada com stored procedures. |
| 16 | BDTriggerTests.sql | Testagem das funcionalidades da base de dados normalizada relacionada com triggers. |
| 17 | BDRoleTests.sql | Testagem das funcionalidades da base de dados normalizada relacionada com roles. |
| 18 | BDFunctionTests.sql | Testagem das funcionalidades da base de dados normalizada relacionada com functions. |

# Conclusões

Na conclusão desta 2ªFase do projeto da Unidade Curricular Complementos de Base de Dados, no geral correu bem, com algumas complicações ao nível de Backup e Recuperação porque não foi possível simular um crash da bd, pois o Windows atirava um erro a dizer que os ficheiros da bd estavam a ser usados e não os deixava modificar/apagar.

Também houve “dificuldades” na procedure de criação de dados de teste, pois devido a normalização da base de dados é necessário haver vários loops/cursores para inserir os dados nas tabelas necessárias, o que não permitiu carregar os 2000-2500 alunos por demorar demasiado tempo a executar. Isto fez com que esta etapa da 2ªfase fosse a que demorou mais tempo pois para alem de necessitar de mecanismos da fase passada, necessitou de muito tempo para testar se estava correto.

Embora com estes impasses, todo o resto do trabalho que esteve no nosso correu bem e sem grandes dificuldades.