UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO

SCC 0241 – Laboratório de Bases de Dados – Turma 4
Prof. José Fernando Rodrigues Júnior – 2º./2010

Projeto Final

Entrega: 24/11 – 23:59h
Datas de Apresentação: 26/11 e 03/12

Projeto em grupos de 4 alunos.

Objetivo: implementar um protótipo, com interface gráfica, com parte da funcionalidade de um sistema real para a base de dados de um Sistema de E-Learning. O projeto final deve ser equivalente ao que é apresentado por sistemas já consolidados como o Moodle e o Tidia.

Ferramentas a serem utilizadas:

- SGBD: Oracle - versão 10g ou 11g.
- Linguagem de programação: livre
- Integração com a base de dados: API da linguagem para Oracle (ex: ODBC, JDBC) ou Pré-Compilador (ex: C/C++)

Observações quanto às ferramentas utilizadas:

- no desenvolvimento deve ser utilizada diretamente a API ou o pré-compilador.
- **NÃO DEVEM** ser utilizados
  - classes DAO;
  - qualquer componente (de ferramentas de desenvolvimento) que faça implicitamente comandos SQL.

Ou seja, **todas** as operações no SGBD devem ser comandos SQL ou PL/SQL **explícitos**.

Funcionalidades a serem implementadas partindo-se da base de dados inicial fornecida para as aulas:

1. (1.5) Interface para listar e navegar os dados do sistema: cursos eletrônicos, participantes de cada curso, recursos disponibilizados por cada curso, professores, suas disciplinas e turmas,... Todas as listagens devem ser apresentadas acompanhadas de seus dados complementares; por exemplo, o tipo de cada participante de um dado curso (aluno, professor, funcionário), a disciplina e o professor vinculado a um dado curso, e assim por diante.

2. (1.5) Interface para edição do sistema. Criação/remoção de um curso, importação dos alunos da disciplina vinculada, adição de participantes externos, descrição do curso, e assim

por diante de acordo com a semântica do sistema e dos dados que são esperados. Criação/remoção de professor/aluno/curso/disciplina/turma, edição dos dados de cada uma destas entidades.

3. (1.5) Interface para criação de grupos de trabalho e para a proposição/entrega de trabalhos para um dado curso. Os trabalhos devem ter prazo de entrega, e a interface deve gerenciar os prazos automaticamente, não permitindo *upload* de arquivos fora de prazo. Deve ser possível atribuir uma nota a cada trabalho.

4. (1.5) Interface para adição de recursos eletrônicos. Deve ser possível escolher qual o tipo do recurso, fazer o *upload* para a base de dados, descrevê-lo e, automaticamente, ter sua disponibilização para acesso. Cada acesso deverá gerar um log de acesso preenchido com todos os dados esperados pela base.

Obs.: o trabalho deve ser mais bem elaborado com relação à manipulação dos dados do que com relação à sua interface. Assim, para a geração de logs de acesso não há necessidade de gerenciamento de sessões HTTP, pode-se supor que o sistema é acessado por um único usuário por vez, o que simplifica a coleta de dados para preenchimento de log de acesso.

5. (1.5) Interface para inserção de notas aos participantes. Deve ser possível adicionar uma nota geral, a qual será combinada (por média ponderada, por exemplo) com as notas dos trabalhos de um dado curso para a geração de uma nota final. Deve ser possível também, exportar as notas finais para o sistema de matrículas, caso o participante não seja externo.

6. (2.5) Interface de auditoria de uso do sistema. Esta funcionalidade deve permitir contagens que caracterizem como o sistema tem sido usado. Para isso deve-se ter um pré-processamento para a criação de visões necessárias à análise, seguido de consultas usando a cláusula GROUP BY definidas e exibidas dinamicamente. Por exemplo: quantos acessos a recursos eletrônicos ocorridas em: um dado curso, ou em todos os cursos, ao longo do tempo, por dia, por mês ou por ano. Quantos acessos a um dado recurso, ou aos recursos particionados por tipo, por um dado aluno, ou por todos os alunos particionados em quartiles de nota. Essas contagens deverão ser realizadas sobre a tabela de logs de acesso em conjunto com as outras tabelas de dados.

Pontos extras:

ATENÇÃO: os pontos extras só serão contabilizados se as funcionalidade de 1 a 6 forem implementadas por inteiro.

- Uma consulta **relevante** adicional às pedidas poderá valer até 2 pontos extras. Será considerado: relevância da consulta, complexidade de implementação e recursos utilizados - melhor ainda se forem recursos não explorados nas aulas e nas práticas (recursos SQL, PL/SQL, objeto-relacional, ...).
- Ou, controle de acesso por tipo de usuário, valendo até 2 pontos extras. Os usuários terão tipos pré-definidos: estudante, professor, administrador, ... e cada um terá

permissões de acesso específicas que determinarão quais opções estarão disponíveis e quais dados poderão ser lidos.

IMPORTANTE:

- Defina pelo menos uma ***view* com junção** na implementação das funcionalidades. A visão deve ser parte da estratégia de implementação. Explique se a *view* é atualizável e justifique.
- Implemente **procedimentos e/ou funções usando pacotes**.
- Implemente ***triggers*.** (Exemplo: para a atualização das notas finais dos participantes).
- Faça o devido **tratamento de erros e exceções**, com mensagens claras para o usuário, indicando o problema e como proceder.
- Se cabível, defina transações.
- Para este sistema, é recomendada a implementação da lógica de negócio usando essencialmente os recursos SQL e PL/SQL.
- Todas as operações devem ser realizadas via interface gráfica. O usuário não deve inserir nenhum comando SQL. Na correção, será considerado, por exemplo, se a entrada de dados é simplesmente a digitação de uma informação que já existe na base de dados (ex: nome de um aluno) ou se são feitos filtros SQL que listam na interface as informações pertinentes para o usuário selecionar o que precisa.

Documentação (1.0):

- Documentação interna
    - explicação do objetivo de cada consulta e das soluções de implementação que forem necessárias para o entendimento do código;
    - explicação da funcionalidade de visões, *triggers*, procedimentos, funções e pacotes.

- Documentação externa:
    - lista dos softwares, ferramentas, linguagens e APIs utilizados, incluindo as versões;
    - indicação do que contém cada arquivo entrege, destacando principalmente onde estão os códigos SQL gerados para o sistema;
    - comentários, discussões e explicações necessárias para a correção, incluindo comentários sobre as funcionalidades extras, caso existam;
    - não incluir código na documentação externa.

Entregar no Tidia (em um único zip):

- *scripts* para criação de objetos na base de dados: *views*, *triggers*, procedimentos, funções e pacotes;
- *script* de inserção de dados usados para os testes;
- arquivos fonte com o código do sistema e o executável; não é necessário incluir os *drivers* utilizados: apenas indique na documentação qual a versão utilizada e o *site* para *download*;
- relatório com a documentação externa.

Apresentação presencial: os trabalhos serão apresentados pelo grupo (presença obrigatória do grupo todo) no LAB 5, dentro do horário de aula. Os membros do grupo apresentarão o sistema e responderão perguntas. Esta avaliação será considerada na nota final.

O sistema deverá ser instalado em uma máquina do LAB 5 antes da apresentação. Os grupos que quiserem, poderão levar máquina própria. Os grupos deverão fazer uma breve apresentação (*slides*) das características gerais do sistema.

Datas de Apresentação: 26/11 e 03/12

OBS: Em novembro, durante as aulas, serão solicitados os nomes dos alunos de cada grupo para a determinação de dias e horários de apresentação.