Universidade Federal de Itajubá
Instituto de Matemática e Computação
COM-231 – BANCO DE DADOS II
Profª Vanessa Souza

BANCO DE DADOS II
DEFINIÇÃO DO TRABALHO PRÁTICO I

Introdução

Este documento descreve as características do Trabalho Prático de Banco de Dados II do curso de graduação em Ciência da Computação da Universidade Federal de Itajubá – 2º semestre de 2021.

O trabalho consiste no consumo e disponibilização de dados obtidos por uma API de Dados.

Objetivo

Coletar dados de uma API de Dados, modelá-los utilizando o modelo relacional, realizar a carga e otimização do banco de dados e implementar um *relatório ad-hoc* para apresentação desses dados.

API de Dados

Segundo Alvarenga et al. (2011), há muitas formas de disponibilizar os dados: eles podem ser publicados em páginas da web, podem ser expostos via uma interface de consulta em um website, ou podem ser acessados diretamente por sistemas eletrônicos via uma API (interface de programação de aplicativo).

Dentre essas formas, o uso de APIs (serviços web), apresenta diversos benefícios, tais como (Kong, 2015):

- Interoperabilidade entre os sistemas, garantindo escalabilidade, facilidade de uso, além de possibilitar atualização de forma simultânea e em tempo real.
- a economia de tempo e custo dos pedidos de acesso à informação
- a possibilidade de cruzar dados de diferentes órgãos gerando novas agregações
- aumento da participação social
- estímulo a inovação
- economia de tempo e dinheiro
- melhora nos serviços governamentais

Na prática, uma API é simplesmente a exposição de uma série de ferramentas, métodos de programação e protocolos, com o objetivo de facilitar a programação de uma aplicação[1].

[1] https://sensedia.com/blog/apis/o-que-sao-apis-parte-1-introducao/

Portanto, uma API de Dados é um serviço web que disponibiliza dados. Para obter os dados é preciso obedecer aos padrões definidos pela interface.

Análise de Dados

A análise de dados busca técnicas que tragam *insights* para os tomadores de decisão. Portanto, o objetivo principal da análise de dados é dar suporte ao tomador de decisão, para que ele interprete de forma adequada os dados e, juntamente com sua experiência e conhecimento do negócio, chegue a conclusões assertivas sobre o problema. O tipo de análise depende do tipo de dado da organização. No contexto de inteligência do negócio, Cruz (2017) cita algumas dessas técnicas, das quais interessa a este projeto : relatórios padrão (*standard reporting*), *dashboards* e relatórios *ad hoc*.

Os relatórios simples são aqueles pré-definidos, sem muita oportunidade para o usuário interagir com o dado. Podem ser utilizados filtros, pode-se permitir exportar e imprimir os dados, mas o conteúdo do relatório não é variável. São úteis para apresentar análises estatísticas e informações cadastrais.

Por sua vez, os Relatórios *Ad-hoc* têm conteúdo variável. Ou seja, permitem ao usuário montar um relatório conforme sua própria necessidade, escolhendo os campos, filtros e, às vezes, até o tipo de gráfico (linha, coluna, tabela, etc.). Essa técnica de análise é uma ferramenta poderosa, pois permite aos diferentes tipos de usuários do sistema, explorar os dados de forma customizada e personalizar seus relatórios.

Dashboards são painéis que unem diferentes visualizações do conjunto de dados numa mesma tela. São visualizações dinâmicas e interativas, que integram, num único ambiente, dados de monitoramento, análise e comunicação.

Metodologia

As etapas do trabalho estão listadas na figura 1. Na **primeira etapa** o grupo deverá estudar o conceito de API e escolher alguma para trabalhar. Cada grupo deve escolher uma API diferente. Para garantir que não haverá grupos utilizando a mesma API, valerá a ordem de apresentação da API pelo grupo no Classroom. A preferência será por “ordem de chegada”. Ou seja, se o seu grupo escolheu uma API que outro grupo já postou no Classroom, vocês deverão escolher outra. Nessa etapa existe uma restrição importante. É necessário escolher uma API que disponibilize dados via interfaces. Ou seja, não será aceito o uso de APIs que retornam um único arquivo (CSV, JSON) com toda a base de dados.

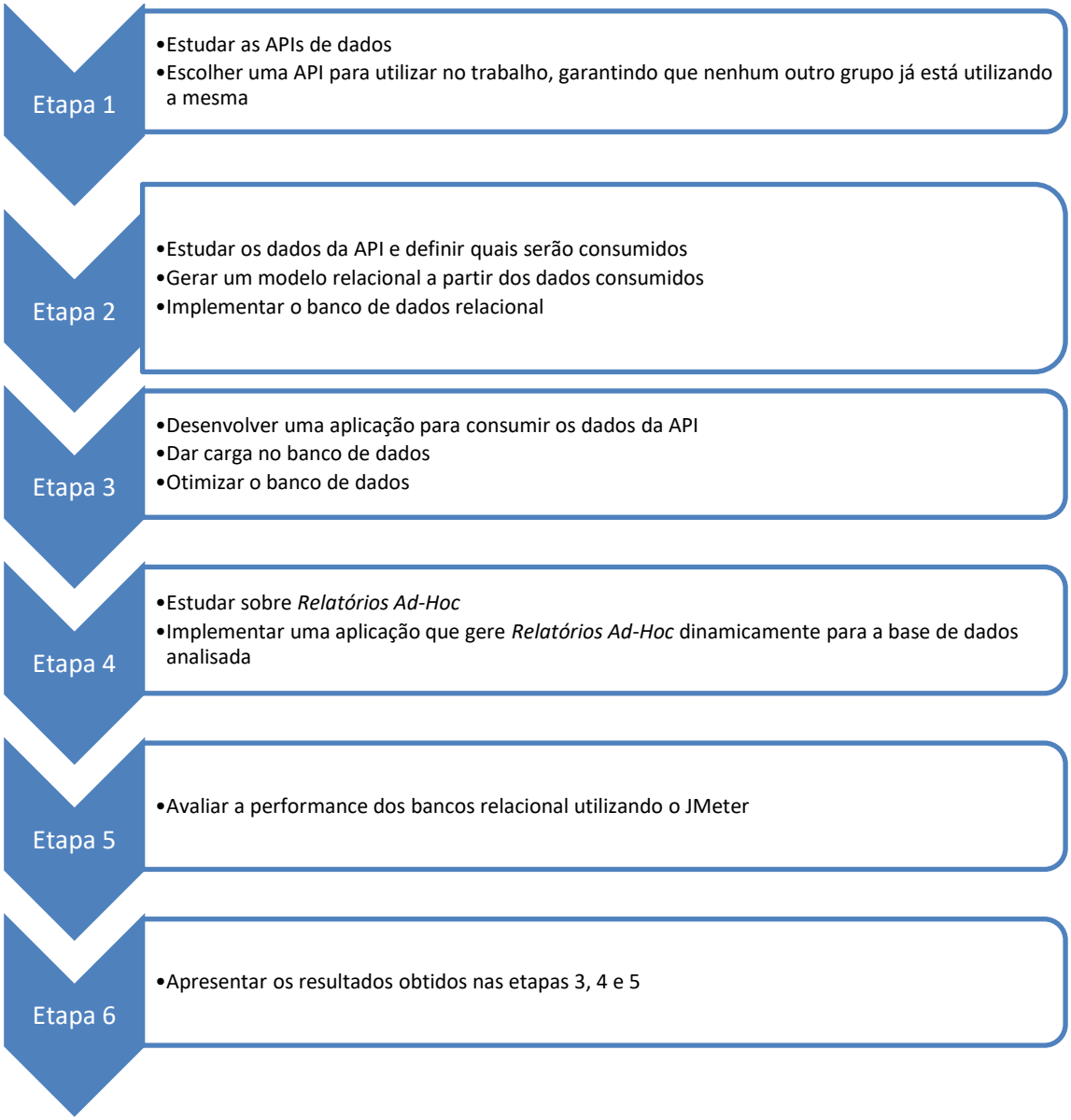

Figura 1 : Etapas do trabalho prático.

Na **segunda etapa**, o grupo fará o trabalho de engenharia reversa, ou seja, irá estudar os dados fornecidos pela API e decidirá quais irão consumir. Em geral, cada API fornece um conjunto enorme de dados. O grupo deverá pensar no conjunto que deseja trabalhar. Os produtos finais dessa etapa serão o modelo relacional e o banco de dados implementado (modelo físico). O SGBD relacional é de livre escolha do grupo.

Essa é uma etapa fundamental no projeto. É preciso entender o dado e o que é possível extrair deles. Algumas questões importantes são:

a) Conhecer o processo
b) Definir as informações mais estratégicas do banco
c) Definir os domínios de cada atributo do banco
d) Pensar em como os dados poderão ser utilizados por diferentes níveis de usuários (operacional, estratégico, gestão).

Na **terceira etapa**, o grupo irá implementar uma aplicação que consome os dados dessa API. A linguagem de programação é livre. Essa aplicação deverá cumprir os seguintes requisitos (figura 2):

1. Acessar a API e fazer o *download* do conjunto de dados a partir de um filtro. Em geral, as APIs retornam dados em algum formato texto (Json, XML, CSV). O grupo pode escolher qual utilizar. A professora prefere JSON ou XML.
2. Extrair os dados do documento texto e dar carga nas tabelas do banco de dados.

É necessário baixar um alto volume de dados. O limite depende do *hardware* utilizado. Baixem dados suficientes para ter uma boa amostra, mas no limite em que o banco retorne os comandos em tempo plausível.

Com o banco populado, o grupo fará as otimizações, como a criação de índices, definição de usuários e de privilégios no banco.

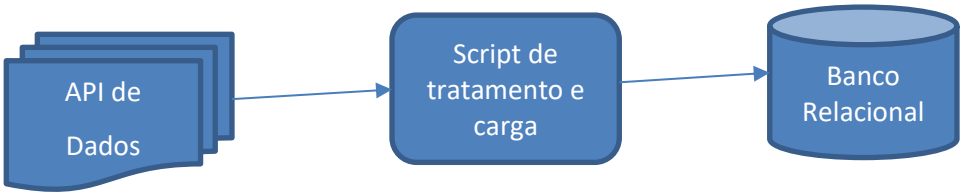

Figura 2 : Atividades da terceira etapa.

A entrega das etapas 1, 2 e 3 deve ser feita no dia **03/12**. Nesta ocasião, o grupo deverá entregar:

a) Relatório do banco de dados
   - Diagrama Relacional, definição de grupos de usuários e suas permissões, definição de índices e justificativas, definição de views, triggers, procedures, funções e suas justificativas, caso existam.
   - Print com o count das tabelas
b) Link para o código desenvolvido (github).
c) Um vídeo mostrando a aplicação funcionando. Neste vídeo é importante mostrar a conexão com o banco, funções que fazem a requisição dos dados da API, funções que salvam os dados tratados no banco.

Na **quarta etapa**, o grupo deverá estudar sobre *relatórios Ad-Hoc* e criar uma aplicação que gere um relatório desse tipo dinamicamente. A linguagem é livre. O importante é permitir que os diferentes níveis de usuários do seu banco possam obter, por meio da interface implementada, um relatório personalizado. Ou seja, o usuário poderá escolher os campos e filtros que deseja e sua aplicação irá gerar o relatório. A dica é utilizar as interfaces de metadados da consulta dos *frameworks* para compor o relatório (número de linhas, atributos, nomes dos campos, tamanho, etc).

A figura 3 apresenta uma interface de criação de um relatório *ad-hoc*.

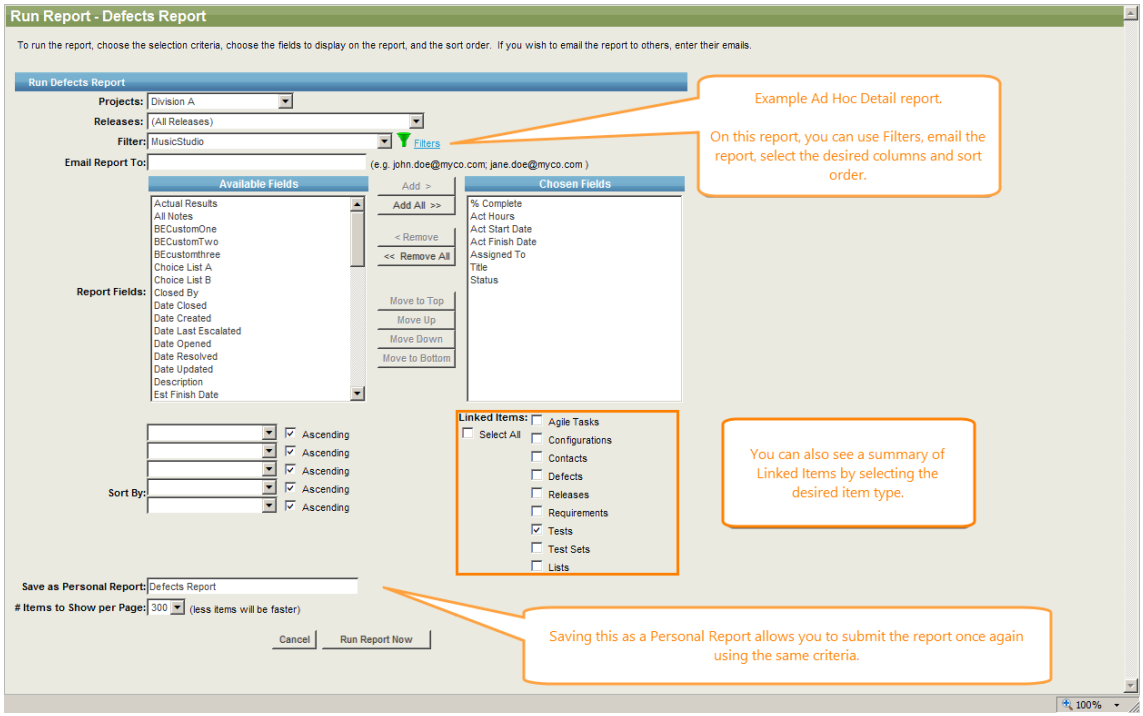

Figura 3 : Exemplo de interface para geração de relatório ad-hoc.
Fonte : https://support.smartbear.com/qacomplete/docs/user/reports/legacy.html.

Nesse trabalho, basta que entreguem os dados em formato tabular. No entanto, quem gerar gráficos e/ou *dashboads* (a partir da consulta *ad-hoc*), terá uma pontuação extra.

Na **quinta etapa**, o grupo escolherá uma consulta custosa no banco para realizar as medidas de performance utilizando o software JMeter[2]. O grupo deverá medir latência nas seguintes condições:

[2] https://jmeter.apache.org/

1. Mantém a quantidade de usuários (*threads*) como 1 e aumenta a quantidade de requisições até o teste retornar erro. Esse será o número máximo de requisições suportadas pelo banco para essa consulta.
2. Defina uma quantidade de requisições fixa e aumente a quantidade de usuários até o teste retornar erro. Esse será o número máximo de usuários suportados pelo banco para essa consulta.

Cada análise deverá gerar um gráfico (latência x número de requisições; latência x número de usuários). Vocês devem aumentar a quantidade de *threads* e requisições manualmente. Cada vez que rodarem, façam 30 repetições do teste e extraiam a média. Essa média é o valor que vai para o gráfico.

É importante ressaltar que os SGBDs definem, por padrão, um número máximo de requisições e usuários que podem se conectar ao mesmo tempo no banco. Esse número é configurável. Sendo assim, o grupo deve alterar esses valores antes de iniciar os testes.

A **sexta etapa** acontecerá nos dias **13 e 15 de Dezembro**, durante as aulas, onde os grupos apresentarão para toda a turma a aplicação e bancos desenvolvidos. A ordem de apresentação será sorteada no início da aula do dia 13. No entanto, todos os grupos deverão entregar a documentação no dia 13 até o meio-dia, que inclui:

a) Relatório:
   - Apresentação dos gráficos gerados, incluindo o público-alvo e o objetivo do mesmo.
   - Resultados dos testes de performance obtidos com o JMeter
b) Link para o código desenvolvido
c) Apresentação que será usada pelo grupo na aula síncrona.

Grupos

7 grupos de 5 pessoas e 1 grupo de 4 pessoas

- Será aberto um fórum no Classroom e os grupos deverão informar os participantes lá até **22/10**.
- A etapa 1 (escolha da API) deve ser feita até o dia **31/10**. Haverá um fórum no classroom também.
- A professora não interfere na formação dos grupos. Quem precisar, utilize o próprio Classroom para formar o grupo.

Considerações Importantes

- A análise da base de dados é FUNDAMENTAL para que vocês tenham um bom trabalho. É MUITO IMPORTANTE que vocês estudem bem a base antes de escolhê-la.

- NÃO é permitido ao grupo criar dados na base. A aplicação deve girar em torno dos dados disponíveis. Vocês podem combinar dados de diferentes APIs.
- NÃO é permitido baixar dados completos, ou seja, arquivos com o conjunto inteiro de dados.
- NÃO UTILIZAR A API DE DADOS IRÁ ZERAR O TRABALHO
- Não é permitido o uso de softwares como Tableau e Power BI para a implementação da consulta ad-hoc. O relatório deve ser implementado por vocês.
- É praxe dos alunos deixar o teste de performance (JMeter) para o final e isso acarreta perda de nota, uma vez que muitos grupos não conseguem executar o JMeter como deveria. Sugiro que vocês elejam um membro do grupo para aprender usar a ferramenta com maestria.
- Dividam o trabalho entre os integrantes do grupo.
- Grupos que quiserem apresentar as etapas 4, 5 e 6 antes do prazo poderão fazê-lo, apresentando apenas para a professora.

PONTUAÇÃO E PRAZOS:

- **Etapas 1, 2 e 3**
  - ENTREGA : 03/12
  - 15 pontos
- **Etapas 4, 5 e 6***
  - ENTREGA : 13/12*
  - Etapa 4 : 10 pontos
  - Etapa 5 : 10 pontos

* Nos dias das apresentações todo grupo deve estar presente. Alunos que não apresentarem o trabalho (etapa 6) terão a etapa 4 e 5 zeradas. O código das etapas 1, 2 e 3 pode ser solicitado novamente.