Universidade Federal de Itajubá
Instituto de Matemática e Computação
SPAD02 – BANCO DE DADOS II
Profª Vanessa Souza

BANCO DE DADOS II
DEFINIÇÃO DO TRABALHO PRÁTICO

Metodologia de Ensino

Objetivo de Aprendizagem:

Aplicar e analisar conceitos relacionados à conexão entre aplicações, banco de dados e APIs de dados. Revisitar conceitos de modelagem, segurança e otimização em banco de dados. Aprender e aplicar métodos de análise de performance em bancos de dados.

Conteúdo Programático:

Integração dos conteúdos segurança, indexação, projeto de banco de dados, conexão com banco de dados, consultas dinâmicas.

Metodologia de Ensino:

Nesse trabalho, a metodologia de ensino utilizada é a **Aprendizagem Baseada em Projetos** (do inglês *Project Based Learning* (**PBL**)). PBL é uma metodologia ativa de ensino que teve suas origens em 1900, quando o filósofo americano John Dewey (1859 – 1952) comprovou o "aprender mediante o fazer". Um dos paradigmas da PBL é que *a construção da aprendizagem é algo que só acontece quando o aluno é ativo, quando está interessado no que está fazendo, quando sua motivação é intrínseca, não extrínseca. Isso significa, que a aprendizagem, para ser bem-sucedida, é autogerada e também, auto conduzida e autossustentada. Ela decorre daquilo que o aluno faz, não de algo que o professor mostre para ele ou faça por ele*.

As principais características dessa metodologia são:

- O aluno é o centro do processo;
- Desenvolve-se em grupos de trabalho;
- Caracteriza-se por ser um processo ativo, cooperativo, integrado e interdisciplinar e orientado para a aprendizagem do aluno.
- Há uma mudança radical no papel do professor que deixa de ser o transmissor do saber e passa a ser um estimulador e parceiro do estudante na descoberta do conhecimento. O professor orienta a discussão de modo a abordar os objetivos previamente definidos

a serem alcançados naquele problema. Em síntese, o professor deve ajudar os alunos a atingirem os objetivos do projeto.

Portanto, no PBL, os alunos são apresentados a um problema que requer a aplicação de conceitos e habilidades de múltiplas disciplinas. Eles então trabalham em equipe para investigar, planejar, executar e apresentar soluções para esse problema. Durante o processo, os alunos podem desenvolver uma variedade de habilidades, como resolução de problemas, colaboração, comunicação, pensamento crítico e criatividade.

Projeto: Desenvolvimento de uma aplicação para geração de relatórios personalizados (*ad hoc*)

Objetivo Geral do Projeto:

Nesse projeto, o grupo deverá desenvolver uma aplicação onde o usuário consiga criar relatórios personalizados, chamados relatórios *ad hoc*. Um relatório *ad hoc* é um tipo de relatório gerado de forma não programada ou pré-definida. Ou seja, nessas aplicações, os usuários podem acessar dados e selecionar as métricas e parâmetros desejados para criar relatórios personalizados conforme suas necessidades específicas. Essa capacidade de criação de relatórios sob demanda permite uma análise ágil e adaptativa, facilitando a tomada de decisões. Relatórios *ad hoc* são conhecidas por serem robustas ferramentas de BI.

Relatórios Personalizados - *Ad hoc*

Relatórios *ad hoc* são relatórios gerados mediante solicitação. Geralmente são criados para um uso específico. Isto implica que tais relatórios são normalmente utilizados poucas vezes ou até mesmo, apenas uma vez. Porém, podem ser muito importantes mesmo que não sejam usados com frequência. Os relatórios ad hoc podem referir-se a qualquer parte da atividade de uma empresa, lidar com combinações aleatórias de dados e assumir diversos formatos [1].

De maneira geral, nas empresas existem os colaboradores que têm como uma de suas atribuições, gerar tais relatórios direto no banco de dados. A cada nova demanda dos líderes de equipe e/ou até mesmo da equipe de desenvolvimento, tais profissionais precisam gerar as

[1] Guzik, M. (2011). Ad Hoc Reporting. In: CFO Techniques. Apress. https://doi.org/10.1007/978-1-4302-3757-0_25

consultas para atender aquela demanda específica que, como dito anteriormente, pode ser utilizada apenas uma vez.

Dessa forma, mais recentemente, as aplicações têm investido em automatizar os relatórios *ad hoc*. Nesse caso, o usuário acessa uma tela onde ele mesmo escolhe os atributos, filtros, operadores de agregação e/ou operadores lógicos que necessita para gerar seu relatório personalizado. Portanto, a ideia central de um relatório *ad hoc* é que o usuário crie esse relatório conforme suas necessidades específicas. E, para tanto, o ambiente precisa ser flexível o suficiente para permitir que o usuário faça sua consulta.

Existe uma grande diferença entre relatório dinâmico e relatório *ad hoc*. No dinâmico, o usuário define valores de atributos pré-definidos. Já no *ad hoc*, o usuário define quais são esses atributos, além dos seus valores.

Suponha a base Northwind que usamos em sala de aula. Um relatório dinâmico tradicional poderia ter uma tela onde o usuário escolha as datas inicial e final para obter o total de pedidos. Além da data, ele poderia também escolher a categoria do pedido, ou o cliente. No entanto, independente do que ele escolha, a saída do relatório é sempre a mesma: uma listagem dos pedidos com atributos específicos pré-definidos pelo programador.

Por outro lado, no relatório *ad hoc*, o usuário começa escolhendo 'as tabelas'. Suponha que ele queira dados de pedido, cliente e produto. Ele consegue escolher as três tabelas porque elas têm uma ligação entre elas. Além disso, ele escolhe quais atributos de qual tabela ele quer no relatório final. Define também os filtros e, quando possível, operadores lógicos (*and* e *or*) e de agregação (*count, sum*, etc).

Os relatórios *ad hoc* são um ramo da inteligência de negócios frequentemente incluídos no contexto do Self-Service BI. O self-service BI é uma abordagem em que os usuários de negócios têm a capacidade de acessar e analisar dados por conta própria, sem depender exclusivamente de especialistas em TI ou analistas de dados. Portanto, com a ajuda de ferramentas automatizadas, os usuários podem criar facilmente relatórios *ad hoc* conforme necessário, sem conhecimento técnico, diminuindo a necessidade de demandar a equipe de banco de dados.

As Figuras 1, 2 e 3 apresentam exemplos de interface para gerar um relatório ad hoc. Nesse projeto, não é necessário implementar interfaces tão sofisticadas. As figuras estão apenas para ilustrar o conceito. Sugiro que acessem os links para ver as imagens em alta resolução.

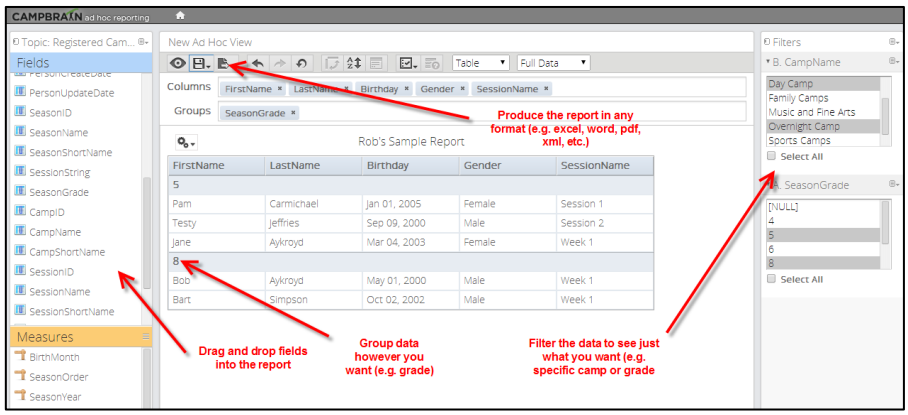

Figura 1: Exemplo de interface para relatório ad hoc. Fonte: Campbrain[2]

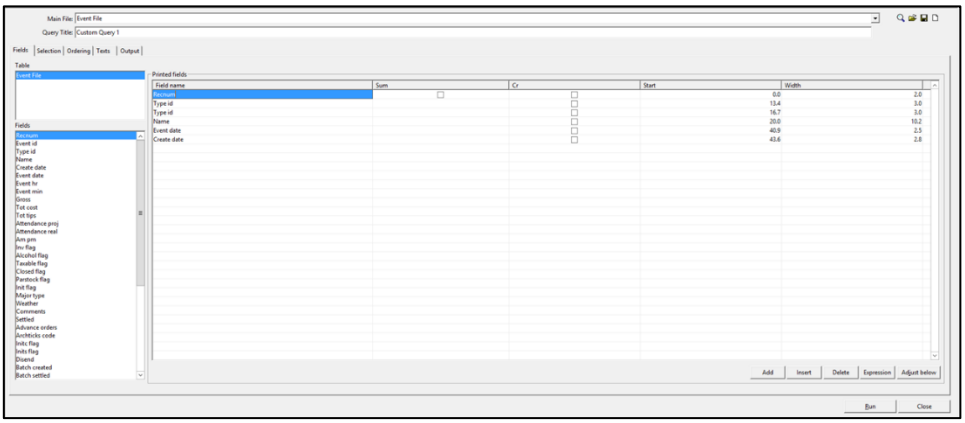

Figura 2: Exemplo de interface para relatório ad hoc. Fonte: Oracle[3]

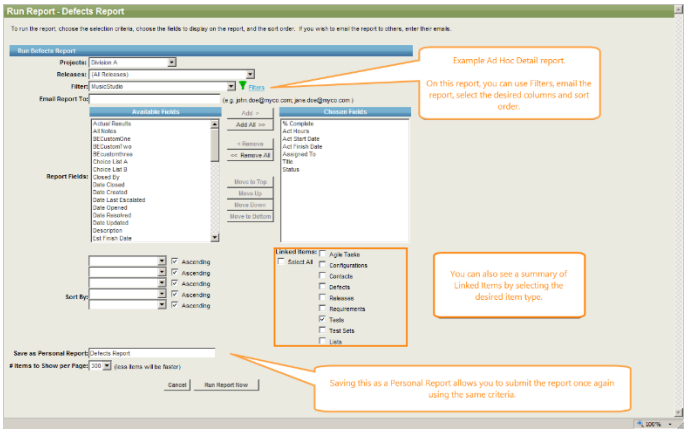

Figura 3 : Exemplo de interface para geração de relatório ad hoc. Fonte: Smartbear[4]

[2] https://campbrain.com/blog/ad_hoc_reporting/

[3] https://docs.oracle.com/cd/E93646_01/doc.311/e93648/t_system_utilities_adhoc_reporting.htm#AdHocReporting-48AC0AD7

[4] https://support.smartbear.com/qacomplete/docs/user/reports/legacy.html

Execução do Projeto:

Esse projeto é dividido em diversas etapas e cada etapa tem um objetivo prático e um objetivo conceitual. Por ser um trabalho no contexto de metodologia ativa, faz parte do trabalho que os alunos estudem os conceitos. Assim como faz parte, que o professor interaja com a equipe para alinhar objetivos e tirar dúvidas. A Figura 1 ilustra as etapas do projeto, que serão detalhadas nas sessões a seguir.

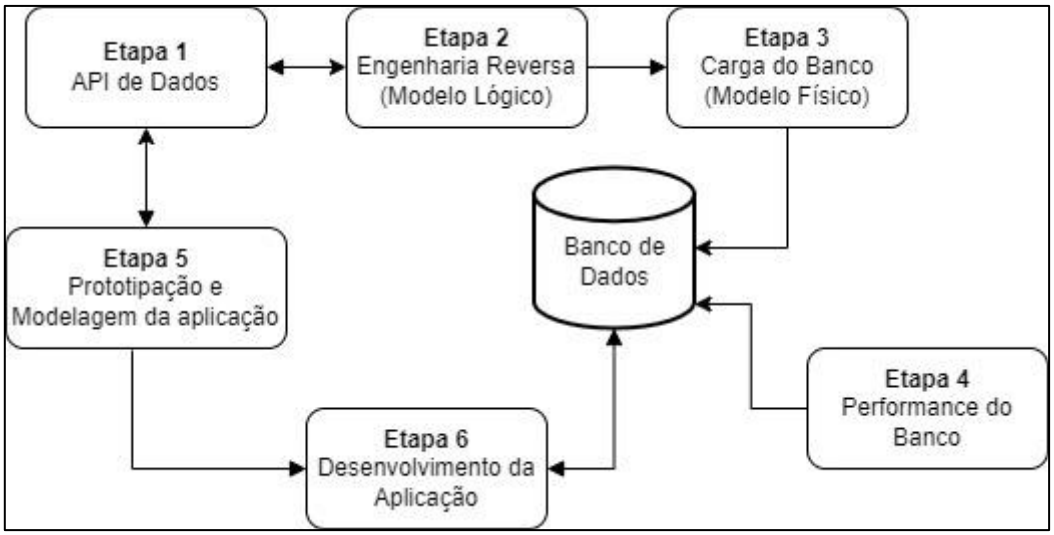

Figura 4: Etapas do projeto para desenvolvimento de uma aplicação que gere relatórios ad hoc.

Etapa 1: 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 a exposição de uma série de ferramentas, métodos de programação e protocolos, com o objetivo de facilitar a programação de uma aplicação[5]. Portanto, uma API de Dados é um serviço web que disponibiliza dados. Para obter os dados é preciso obedecer aos padrões definidos pela interface.

Objetivo Geral da Etapa 1: escolher uma API de dados que será utilizada no projeto.

Objetivos Conceituais

- Aprender sobre APIs de dados
  - O que é, como se acessa, como os dados são entregues, quais são as restrições impostas pela API para acessar os dados, quais são os motivos que levam aos provedores de dados impor tais restrições?
  - Como se implementa uma API de dados?

Objetivos Práticos

- Escolher uma API de dados
  - Essa escolha não é trivial. Para escolher a API, o grupo deverá entrar em consenso sobre o tema do trabalho.
  - A API deverá estar atrelada ao relatório ad hoc que a aplicação disponibilizará no final do projeto.

Entregas

O grupo deverá informar qual API será utilizada. Ademais, deverão responder as seguintes perguntas:

- Qual o público-alvo para a aplicação que será desenvolvida?
- Qual a motivação para desenvolver uma aplicação para esse público-alvo?

Instruções Adicionais

É importante que o grupo escolha uma aplicação com a qual os integrantes tenham alguma intimidade e, principalmente, interesse pelo tema. Vocês terão papeis de desenvolvedores e também de clientes dessa aplicação. O grupo definirá os requisitos e, por isso, é importante que entendam e tenham interesse no dado.

As seguintes APIs estão banidas e não podem ser utilizadas para esse projeto:

- spotify
- cartola

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

- pokemon
- IBGE (cidades/estados/capitais)
- filmes (qualquer uma com esse tema)
- jogos (qualquer uma com esse tema)

Outro aspecto importante é que a aplicação deverá consumir dados que venham da API. O grupo não pode gerar nenhum tipo de dado, mas pode combinar APIs, caso ache necessário.

Para ter bons relatórios *ad hoc*, a base de dados deve ter um equilíbrio entre atributos descritivos e quantitativos. Bases muito quantitativas limitam a geração de relatórios. Por outro lado, bases muito descritivas não permitem inclusão de operadores de agregação, por exemplo.

A seguir, algumas características desejáveis da API:

- Que seja possível mapear em um conjunto de, ao menos, 4 tabelas do banco relacional
- Que os dados não sejam muito resumidos e que possam ser qualificados. Isso ajudará na hora de gerar o relatório dinâmico.
- Que tenha dados suficientes para estressar o sistema durante os testes de performance. Não existe uma quantidade ideal. Depende do tamanho dos registros e da capacidade da máquina que executará os testes. No entanto, espera-se bases com mais de 2 mil registros em ao menos uma das tabelas.

Etapa 2: Engenharia Reversa

No contexto desse projeto, a engenharia reversa de dados consiste em analisar os dados ofertados pela API de dados escolhida pelo grupo (Etapa 1) e, a partir do mapeamento desses dados, gerar a documentação para o banco de dados. A documentação será formada pelo modelo entidade-relacionamento, modelo relacional e dicionário de dados.

Não é necessário utilizar todos os dados ofertados pela API. A escolha dos atributos deve estar associada ao objetivo da aplicação final. Ou seja, o público-alvo que utilizará o relatório *ad hoc*.

O dicionário de dados é um metadado dos dados do banco. Isto é, é um documento que descreve o significado de cada tabela e, para cada atributo do banco, além do significado, descreve o tipo de dado, valores possíveis e outras características relevantes.

Objetivo Geral da Etapa 2:

Gerar a documentação do banco de dados, que inclui: modelo entidade-relacionamento, modelo relacional e dicionário de dados.

Objetivos Conceituais

- Aprender sobre engenharia reversa de dados
    - A engenharia reversa pode ser muito útil quando nos deparamos com um banco de dados legado, sem documentação. Para esse projeto, os dados vêm da API, mas o processo se assemelha ao de um BD.
- Aprofundar nos conceitos sobre documentação do banco de dados

- Os modelos entidade-relacionamento e relacional são as documentações principais de uma base de dados relacional. Ademais, o dicionário de dados é também muito útil para perfeito entendimento do conjunto de dados. Nesse link[6] há um modelo de dicionário de dados bom para iniciar o entendimento do conceito. Mas, procurem demais referências.

Objetivos Práticos

- Conhecer os dados
  - É importante estudar o conjunto de dados, porque ele é a base para a execução do projeto. Se a base for ruim, o relatório *ad hoc* fica limitado.
- Gerar a documentação para o banco que será implementado
  - Modelo entidade-relacionamento, modelo relacional, dicionário de dados

Entregas

Documentação do banco de dados

Instruções Adicionais

É comum que as APIs de dados tenham dados de baixa qualidade, o que acarreta atributos com muitos valores nulos ou faltantes; identificadores que não são únicos; *strings* desformatadas. Por isso, identificar bem os atributos bons nesse momento do projeto ajuda a não ter retrabalhado depois. Além disso, ao identificar problemas nos dados, vocês podem já pensar em alguma solução e incorporá-la no código que fará a carga no banco. Isso também poupa bastante tempo.

Um ponto importante é que o dicionário de dados vai evoluir com o andamento do projeto. Nessa etapa talvez não seja possível identificar o range total de valores para cada atributo, ou se tem valores nulos, por exemplo. Não tem problema. Foquem mais na descrição dos atributos e seus relacionamentos.

Etapa 3: Carga no Banco

Nessa etapa, o grupo deverá gerar uma aplicação para consumir dados da API e popular o banco de dados da aplicação, implementando um processo chamado ETL.

ETL (Extração, Transformação e Carga) é um processo para extrair dados de um sistema, processá-los, modificá-los e posteriormente inseri-los numa base (banco de dados). A concepção de um processo ETL incide sobre o mapeamento dos atributos dos dados de uma ou várias fontes para os atributos das tabelas do banco de dados.

[6] https://medium.com/psicodata/dicionario-de-dados-ac3ce726c34b

O grupo pode escolher qualquer linguagem de programação e qualquer SGBD relacional. O grupo deverá acessar a API por suas rotas, processar o dado e salvar os dados no banco de dados local.

Após concluída a carga, o grupo deverá implementar no banco e incluir em sua documentação, aspectos relativos à performance e segurança.

Objetivo Geral da Etapa 3:

Implementar o modelo físico do banco e popular a partir da API de dados

Objetivos Conceituais

- Aprender sobre ETL
  - A ETL é um processo muito comum em aplicações que envolvem dados. O grupo deve estudar quais atividades podem ser executadas durante a ETL. Exemplo dessas atividades é a limpeza de dados. Ao mapear um atributo que tem muitos registros nulos, por exemplo, durante a ETL, o grupo remove esse atributo da carga; ou, ao identificar valores fora do padrão (string em campos numéricos), durante a ETL, esses registros são removidos ou transformados. Sugestão de referência , (Costa, 2009)[7]sessões 2.3 a 3.3.
- Praticar conceitos aprendidos sobre segurança e performance
  - Estudar a base e a aplicação para definir quais índices e quais perfis de usuários devem ser criados
- Praticar os conceitos aprendidos sobre conexão com banco de dados
  - Para essa etapa não é necessário utilizar ORM, mas, caso o grupo queira, é permitido.

Objetivos Práticos

- Implementar uma aplicação que faça o processo de ETL
- Popular o banco de dados, de forma que os dados estejam íntegros e consistentes, de acordo com o modelo definido previamente.
- Implementar os índices
- Criar usuários e definir privilégios

Entregas

- Vídeo explicando a aplicação desenvolvida e o processo de ETL
- Atualização da documentação do BD
- Print com o *count* das tabelas
- Link para o código no gitHub

Instruções Adicionais

É comum que as APIs de dados tenham dados de baixa qualidade, o que acarreta atributos com muitos valores nulos ou faltantes; identificadores que não são únicos; *strings* desformatadas.

[7] http://monografias.ice.ufjf.br/tcc-web/exibePdf?id=76

Por isso, identificar bem os atributos bons nesse momento do projeto ajuda a não ter retrabalhado depois. Além disso, ao identificar problemas nos dados, vocês podem já pensar em alguma solução e incorporá-la no código que fará a carga no banco. Isso também poupa bastante tempo.

Um ponto importante é que o dicionário de dados vai evoluir com o andamento do projeto. Nessa etapa talvez não seja possível identificar o range total de valores para cada atributo, ou se tem valores nulos, por exemplo. Não tem problema. Foquem mais na descrição dos atributos e seus relacionamentos.

De maneira análoga, os índices também evoluem. Nessa etapa, a geração dos índices será baseada na ideia inicial dos relatórios. Posteriormente, pode ser necessário reavaliar.

Etapa 4: Performance do Banco de Dados

Nessa etapa, o grupo fará uma análise de performance no banco de dados criado na Etapa 3. A principal métrica utilizada é a latência. Define-se latência como o intervalo de tempo decorrido entre o envio de uma solicitação de acesso aos dados e o recebimento da resposta correspondente. A latência é influenciada por diversos fatores, como a velocidade de processamento do hardware, a eficiência dos algoritmos de consulta, a capacidade de rede e a carga de trabalho do sistema. Quanto maior a latência, menor o desempenho do sistema. Nesse projeto, focaremos na latência de consulta, que corresponde ao tempo que o sistema leva para executar uma consulta específica e retornar os resultados desejados para o cliente.

Existem softwares que ajudam realizar testes de latência. Nesse projeto, utilizaremos o JMeter[8]. No JMeter é possível fazer diversos tipos de testes, incluindo latência em função do número de usuários no banco e a latência em função do número de requisições realizadas ao mesmo tempo. Para executar essa etapa, o grupo escolherá uma consulta custosa no banco para realizar as medidas de performance. O grupo deverá medir latência nas seguintes condições:

1. Mantém a quantidade de usuários (*threads*) fixa e aumenta a quantidade de requisições (*loop count*) 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.

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

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 (*loop count*) manualmente. Cada vez que rodarem, façam 30 repetições do teste e extraiam a média. Essa média é o valor que vai para o gráfico. Na Figura 5 há exemplos de gráfico de *latência x número de usuários* e *latência x número de usuários*. No primeiro caso, o número de requisições por usuário é fixado em 1. No segundo caso, o que fica fixo é o número de usuários.

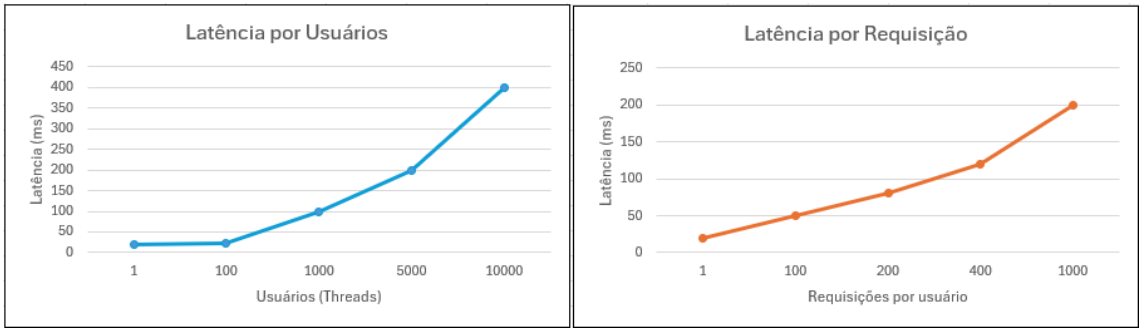

Figura 5 : Exemplo de gráfico que mostra a latência em função de: a) aumento do número de usuários conectados no banco e; b) aumento do número de requisições por usuário.

A Figura 6 mostra a diferença conceitual dos testes. A latência por usuários corresponde ao *thread count*. Já as requisições por usuário, corresponde ao *loop count*.

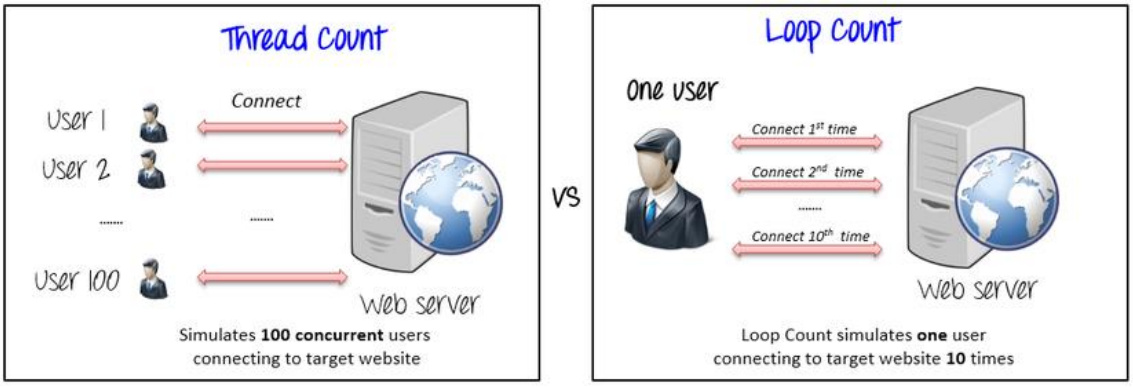

Figura 6: Diferença entre os testes. Fonte:[9] [10]

[9] https://ducmanhphan.github.io/2020-01-21-How-to-use-JMeter-to-test-performance/
[10] https://trailheadtechnology.com/performance-and-load-testing-with-jmeter/

Objetivo Geral da Etapa 4: Teste de performance utilizando o JMeter

Objetivos Conceituais

- Aprender sobre latência
  - O objetivo desse teste é informar ao cliente o limite do sistema. Ao conhecer esse limite, pode-se evitar gargalos no desempenho do sistema.
- Aprender sobre ferramentas de teste de desempenho
  - O JMeter não é um software específico para banco de dados. Com ele é possível testar desempenho de aplicações web como um todo.
  - Conhecer ao menos uma ferramenta de teste de desempenho é importante para profissionais de TI. Assim como é importante entender um gráfico de latência.

Objetivos Práticos

- Realizar os testes de desempenho utilizando o JMeter
- Discutir os resultados

Entregas

Relatório com os resultados dos testes e a discussão desses resultados.

Instruções Adicionais

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.

Não utilizem o gráfico gerado pelo JMeter de forma automatizada. Vocês devem aumentar os valores manualmente e gerarem os gráficos.

Atentem-se à estética dos gráficos. Eles precisam ser simples de analisar.

Atentem-se à discussão dos resultados. Qual significado dos valores obtidos?

Para validar os testes, é imprescindível que o grupo informe a arquitetura da máquina utilizada nos testes.

Etapa 5: Prototipação e Modelagem da Aplicação

Nessa etapa, o grupo deverá modelar os relatórios *ad hoc* e definir a arquitetura da aplicação. Para tanto, devem definir a linguagem de programação, que deve ser, obrigatoriamente, uma linguagem orientada a objetos. Isso porque, a aplicação que será desenvolvida na Etapa 6, deverá utilizar um *framework* ORM.

Objetivo Geral da Etapa 5: Prototipar a aplicação e definir sua arquitetura

Objetivos Conceituais

- Reforçar os conceitos de Relatório personalizado (*Ad hoc*)
  - O grupo deverá refletir se a aplicação que está sendo modelada, de fato atende aos requisitos de um relatório *ad hoc*.
- Reforçar os conceitos de modelagem de sistema utilizando o padrão de arquitetura MVC
  - O grupo deverá definir a arquitetura do sistema utilizando o padrão MVC

Objetivos Práticos

- Prototipar a interface da aplicação para o relatório ad hoc
- Definir a arquitetura da aplicação (*back-end* e *front-end* e comunicação entre eles)

Entregas

Protótipo da interface da aplicação e o diagrama de arquitetura em camadas.

Instruções Adicionais

Não é preciso um *front-end* 'bonito'. O mais importante é que ele seja funcional.

Essa etapa pode ser feita desde o início do projeto. A interface do relatório personalizado tende a evoluir, mas para o sucesso do projeto, é importante que a base dele seja estabelecida desde o início.

O protótipo pode ser feito utilizando qualquer tecnologia. Ou até mesmo desenhando.

Etapa 6: Desenvolvimento da Aplicação

Nessa etapa, o grupo deverá implementar a aplicação que permite ao usuário gerar seu relatório a*d hoc*. A aplicação deve ser desenvolvida em uma linguagem orientada objetos e fazer uso de um *framework* ORM.

Para que de fato o relatório seja personalizado, a consulta realizada pela aplicação deve ser gerada em tempo real, considerando as tabelas, atributos e filtros selecionados pelo usuário no front-end.

A chave para o relatório ad hoc são consultas dinâmicas. Define-se consulta dinâmica como uma consulta a um banco de dados que é construída em tempo de execução (dinamicamente), em vez de ter uma estrutura de consulta estática definida no código. Ou seja, os parâmetros da consulta, como condições de filtro, ordenação, projeção de colunas e outros critérios variam em função da necessidade do usuário. É ele quem escolhe.

Na Figura 7 tem um exemplo de consulta dinâmica usando o ORM Prisma. Nesse caso, no front-end, o usuário teria escolhido as tabelas user (atributos name e email) e posts (atributos title e

contente); filtrou por posts que tenham ou no título ou no conteúdo, a palavra ‘dados’; e o usuário também escolheu ordenar os resultados pelo atributo name. Essas informações foram passadas do front para o back e, com as opções escolhidas, o back cria a consulta e a envia para o banco. Os resultados que voltam do banco são enviados para o front, que imprime na tela. Portanto, uma premissa dessa aplicação é que a consulta enviada para o banco deve ser aquela ‘criada’ pelo usuário no front. Ou seja, não pode fazer um select em tudo no banco e filtrar no front.

```
async function getUsersWithPostsContainingWord() {
  const usersWithPostsContainingWord = await prisma.user.findMany({
    select: {
      name: true,
      email: true,
      posts: {
        select: {
          title: true,
          content: true,
        },
        where: {
          OR: [
            { title: { contains: 'dados' } },
            { content: { contains: 'dados' } }
          ]
        },
      },
    },
    orderBy: {
      name: 'asc',
    },
  });
  console.log(usersWithPostsContainingWord);
}
```

Figura 7: Exemplo de código de consulta dinâmica com o ORM Prisma, criado pelo ChatGPT.

Objetivo Geral da Etapa 6: Implementar a aplicação que gera relatórios ad hoc.

Objetivos Conceituais

- Aprender o conceito de consulta dinâmica
- Reforçar conceitos sobre ORM e conexão com banco de dados

Universidade Federal de Itajubá
Instituto de Matemática e Computação
SPAD02 – BANCO DE DADOS II
Profª Vanessa Souza

Objetivos Práticos

- Implementar uma aplicação utilizando ORM e consulta dinâmica

Entregas

- Apresentação da aplicação funcionando
- Relatório atualizado do projeto, incluindo os resultados obtidos nas etapas 5 e 6
- Link para o código no gitHub

Instruções Adicionais

A consulta dinâmica é difícil de ser implementada sem ORM.

Prazos e Entregas

As etapas 1, 2, 3 e 4 devem ser entregues até o dia **03/06**. Nesta ocasião, o grupo deverá compilar as entregas definidas nas etapas em um relatório.

Nos dias **01 e 03 de Julho**, os grupos farão a apresentação final do projeto para a turma. A ordem de apresentação será sorteada no início da aula do dia 01/07. No entanto, todos os grupos deverão entregar a apresentação no **dia 01/07 até às 20h**.

Durante a apresentação, o grupo deverá seguir os seguintes tópicos:

- Descrever a API utilizada
- Descrever as ferramentas utilizadas (linguagem, SGBD, frameworks...)
- Contextualização: quem seria o cliente da sua aplicação?
- Modelo Relacional
- Testes de Performance
- Rodar a aplicação em tempo real, demonstrando o relatório ad hoc funcionando
- Considerações Finais: impressões do grupo sobre o trabalho (dificuldades, estratégias, contribuições...)
- Link para o código desenvolvido (github)

Cada etapa vale 50% da nota referente ao Trabalho Prático 1.

GRUPOS

Grupos de 5 pessoas. Cada integrante do grupo deverá assumir um papel de responsabilidade.

Os papeis são:

- DBA: Responsável pelo banco de dados (modelagem, implementação, documentação, qualidade dos dados e performance)
- Eng. Software: Responsável em garantir os requisitos da aplicação, modelar a aplicação e o relatório ad hoc. Além disso, irá gerenciar o grupo.
- Documentador: responsável pela documentação geral do trabalho.

- Programador: Responsável pelos códigos.

Apesar dos papeis, todos devem entender e executar o projeto por completo.

DINÂMICA

Será aberto um fórum no SIGAA e os grupos deverão informar:

- Participantes do grupo
- API que irá utilizar
- Durante o semestre há aulas reservadas para consultorias do trabalho. Nessas aulas, o grupo deve apresentar o andamento do trabalho e tirar dúvidas. A presença nas consultorias pontua na entrega final.

Considerações Importantes

- 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.
- A professora não interfere na formação dos grupos. Quem precisar, utilizar o próprio fórum.
- Nos dias das apresentações todo grupo deve estar presente.

Rubrica

Abaixo, os critérios de avaliação do trabalho e o que é esperado em cada etapa.

Universidade Federal de Itajubá
Instituto de Matemática e Computação
SPAD02 – BANCO DE DADOS II
Profª Vanessa Souza

Trabalho Prático						
Etapas	**Critérios**	**Pontuação**	**Desempenho**			
			1	**2**	**3**	**4**
1 a 4	Realizou a carga no banco de dados corretamente. Leu os dados de diferentes rotas da API de dados, aplicou processos de limpeza, correção ou complementação dos dados e populou o banco com quantidade suficiente de dados.	2	Excelente. Cumpriu todos os requisitos e processou os dados para alcançar a modelagem realizada. 100%	Bom. Fez a carga sem necessidade de processar os dados para alcançar a modelagem realizada. 70%	Satisfatório. A carga veio de uma única rota da API. 50%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Modelagem do Banco de Dados. O grupo estudou os dados da API escolhida, pensou no público alvo, nos relatórios ad hoc e fez uma modelagem que atende a esses requisitos.	2	Excelente. Cumpriu todos os requisitos e criou corretamente os modelos entidade-relacionamento e relacional. 100%	Bom. Modelo gerado sem conexão direta com o objetivo do banco de dados ou pobre em semântica. 70%	Satisfatório. Há erros nos diagramas. 50%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Segurança e performance do banco de dados	2	Excelente. Criou grupos de usuários e justificou corretamente. Criou índices justificados pelas consultas que atendem ao relatório ad hoc. Pensou em triggers ou funções que podem auxiliar na lógica do banco. 100%	Bom. Criou usuários e índices bem justificados. 70%	Satisfatório. Criou apenas usuários ou índices, sem boas justificativas. 40%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Testes de Performance	3	Excelente. Fez os testes corretamente, seguindo o roteiro do projeto. Gerou os dois gráficos. Os gráficos são legíveis e esteticamente bonitos. Houve interpretação dos resultados. 100%	Bom. Realizou os testes, mas os gráficos são confusos. Houve interpretação dos resultados. 70%	Satisfatório. Fez os testes, mas a interpretação dos resultados é pouco conclusiva. 40%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Relatório	1	Excelente. O relatório é bem formatado, não apresenta erros gramaticais. As figuras são referenciadas no texto, possuem legenda. Todas as fontes são corretamente referenciadas. 100%	Bom. O relatório apresenta erros gramaticais ou de formatação. 70%	Satisfatório. O relatório não apresenta conclusões satisfatórias. 50%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%

Total: 10 pts

Extra: 0.5pt se usar o latex

Universidade Federal de Itajubá
Instituto de Matemática e Computação
SPAD02 – BANCO DE DADOS II
Profª Vanessa Souza

Trabalho Prático						
Etapas	**Critérios**	**Pontuação**	**Desempenho**			
			1	**2**	**3**	**4**
5 e 6	Modelagem e Prototipação	1	Excelente. Prototipação e diagrama corretos. 100%	Bom. Prototipação e/ou diagrama incompletos ou incorretos. 60 a 80% (depende do caso)	Satisfatório. Não fez a prototipação ou diagrama. 30%	Insatisfatório. Não entregou. 0%
	Código	2	Excelente. Utilizou ORM de maneira adequada. 100%			Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Consultas Dinâmicas	3	Excelente. As consultas são dinâmicas. Parâmetros como nome da tabela, atributos e filtros são lidos no front e enviados para o back. A consulta roda no back e retorna apenas os registros e atributos que atendem a consulta. 100%		Satisfatório. As consultas são dinâmicas, mas não atende completamente o que foi solicitado. 50%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Relatório Ad hoc	2	Excelente. O relatório atende ao público que foi pensado. O grupo justificou bem seu uso por eventuais usuários. 100%		Satisfatório. O relatório não está conectado com o uso para o qual foi pensado. 40%	Insatisfatório. Não fez ou não cumpriu os requisitos. 0%
	Apresentação Oral	2	Excelente. Grupo organizado, slides bem formatados e esteticamente bonitos. Grupo soube responder as questões. 100%	Bom. Grupo não se mostrou bem preparado para apresentação. Apresentação centrada em alguns componentes. 50%	Satisfatório. Grupo não apresentou os resultados esperados. 30%	Insatisfatório. Grupo não fez a parte principal do trabalho que são as consultas dinâmicas utilizando ORM. 0%

Total: 10 pts

Extras: (+) 0.5pt se gerar gráficos dinâmicos associados às consultas ad hoc

(+) 0.5pt se aplicação em uma máquina e BD em outra. O BD não pode estar na nuvem.

(-) 1pt se não participar das consultorias

Nota Final: ((FASE 1 + extras) + (FASE 2 + extras) - Desconto))/ 2