**Banco de Dados, PSET 1**

**Gustavo Saraiva Mariano**

**202306629**

**CC1Mc**

[gsaraivam10@gmail.com](mailto:gsaraivam10@gmail.com)

**Questão 1:** São softwares que possibilitam ter um controle sobre as alterações feitas em cima de um código fonte, sendo possível identificar o que, quem, e porque atualizou determinada coisa dentro do código. Além disso, os VCS salvam as “etapas” de um código, que quando desejado, o usuário consegue escolher a etapa que deseja voltar para recuperar algo desejável. Não só, esse sistema também permite criar um novo ramo do código escrito para realizar testes e alterações sem afetar o código fonte principal. Também possuem sistema de defesa próprios para evitar invasões ao código, sem falar que apenas usuários que tem permissão podem editar o código. Possibilitam o sistema de compartilhamento do código, onde diversos desenvolvedores autorizados podem trabalhar no mesmo código ao mesmo tempo, sem a necessidade de compartilhar as edições entre si.

**Questão 2:** O GitHub é uma plataforma na web em que disponibiliza a criação de um ambiente colaborativo entre desenvolvedores, em que utiliza os recursos disponibilizados pelo Git, que é um software, como seu sistema de controle de versões. É possível utilizar eles separadamente, porém a vantagem de utilizá-los simultaneamente, é que o GitHub fornece uma interface gráfica amigável online para trabalhar com o Git sendo executado.

**Questão 3:** O sistema distribuído de controle de versões se diferencia dos VCS locais e centralizados porque, esses, por sua vez, as cópias que eram alteradas ou atualizadas, ficavam salvas em um servidor isolado da máquina utilizada pelo desenvolvedor, gerando uma dependência do estado na máquina, em que se estivesse inoperante, o desenvolvedor não conseguia acessar os códigos em que desejava trabalhar. No sistema distribuído, foi criado a ideia de que cada desenvolvedor tivesse uma cópia dos códigos em sua própria máquina, tirando a necessidade de depender do funcionamento de um servidor separado, e mesmo assim possibilitando o trabalho cooperativo.

**Questão 4:** O diretório de trabalho é o local aonde o usuário realiza as mudanças nos arquivos, escreve o código, altera o código, cria arquivo novo, remove arquivo, dentre outros. Quando é realizado uma alteração dentro do diretório de trabalho, o Git nomeia o arquivo que ocorreu uma mudança com o nome de “modified”.

A área de staging é o local em que o usuário seleciona as mudanças que ele realizou para encaminhar para o histórico de versões do Git, como uma lista de verificação de qual ou quais arquivos vão ser enviados para salvar sua versão. Quando um arquivo é adicionado na área de staging, o Git nomeia ele com “staged”, indicando que o respectivo arquivo vai ser encaminhado para o histórico de versões.

O repositório é o local onde todas as alterações são registradas de forma permanente, onde posteriormente permite o acesso a essa versão permanente pelo próprio desenvolvedor ou os colaboradores do projeto. Quando são enviadas para o repositório, o Git as nomeia com “commited”, indicando que foi registrado as alterações feitas.

**QUESTÃO 5:** Um erro nesse projeto está na tabela “clientes” com o uso de atributos multivalorado na tabela, nas colunas “telefone1”, “telefone2”, “telefone3”, uma forma de resolver esse problema de atributo multivalorado seria deixar apenas a coluna “telefone1” como NOT NULL, com as outras duas colunas de telefone podendo aceitar valores nulos, nos casos de usuários que não possuem mais de um telefone. Também possui um erro de atributo composto na tabela “lojas”, na coluna de “endereço\_fisico”, uma forma de resolver esse problema seria separar a coluna “endereço\_fisico” em partes, como “logradouro”, “cidade”, “estado” dentre outros aspectos de localização.

**QUESTÃO 6:** As tabelas “clientes” e “lojas” possuem um relacionamento N:N intermediado pela tabela “pedidos”, pois uma loja pode ter zero, um ou mais pedidos em processo, e um cliente pode ter zero, um ou mais pedidos feitos em diferentes ou na mesma loja. As tabelas “clientes” e “lojas” também possuem outro relacionamento N:N intermediado pela tabela “envios”, uma loja pode ter zero, um ou mais envios para realizar, enquanto um cliente pode ter zero, um ou mais pedidos para receber. Outras tabelas que possuem um relacionamento N:N são as tabelas “pedidos” e “envios”, intermediadas pela tabela “pedidos\_itens”, pois um pedido pode ter zero, um ou mais itens pedidos, e um envio pode ter zero, um ou mais pedidos de itens para realizar. Por fim, as tabelas “lojas” e “produtos” também possuem um relacionamento N:N, intermediado pela tabela “estoques”, uma loja pode ter zero, um ou mais estoques, e um produto pode ter zero, um ou mais em disponível em estoque. Todas essas tabelas possuem relacionamento N:N pois suas respectivas chaves primárias fazem parte da tabela que as intermedeia em forma de chave estrangeira ou parte da chave composta.

**QUESTÃO 7:** Não necessariamente é errado, vai depender das necessidades específicas do banco de dados. Ao não incluir a coluna “numero\_da\_linha” na chave primária, não haverá restrição que faça com que essa coluna tenha valores únicos, unicidade, ocasionando na ocorrência de linhas duplicadas com o mesmo valor na coluna. Essa ocorrência dificultaria a ter um melhor desempenho em algumas consultas desse banco de dados.

**QUESTÃO 8:** A tabela “pedidos\_itens” faz relacionamento identificados com as tabelas “pedidos” e “produtos” porque fazem uma ligação direta entre essas tabelas, estabelecendo uma ligação entre os itens do pedido e as informações do pedido e dos produtos correspondentes. Já na tabela envios, o relacionamento não é identificado porque a associação entre os itens do pedido e os envios é estabelecido de forma indireta, por meio de outras tabelas, ao em vez de ser de forma direta entre essas duas tabelas.

**QUESTÃO 9:** O único tipo de relacionamento que consegue armazenar dados é o de tipo N:N, pois envolve a necessidade de uma tabela intermediária para relacionar outras duas tabelas principais. Essa tabela intermediária envolvida possui suas próprias colunas que podem servir como complementar das tabelas principais, como exemplo, data de início e término, horário, dentro outros atributos específicos que são mais interessantes manter em uma tabela separada da principal para melhor distribuição e controle dos dados.

**QUESTÃO 10:** O tipo de dado “BLOB” é uma abreviação para “Binary Large OBject”, que transcreve grandes volumes de dados, como imagens, vídeos, áudios, etc., em códigos binários, a fim de serem arquivados dentro de um database. Essa categoria é utilizada quando não é possível armazenar arquivos grandes no formato de dado estruturado. O emprego desse gênero de dado no projeto é adequado, uma vez que possui tabelas, lojas e produtos, nos quais serão armazenados os logos das lojas e as imagens dos produtos, respectivamente.

**QUESTÃO 11:** No PostgreSQL, o banco de dados é a junção de tabelas, funções, etc., que armazenam e gerenciam dados, cada banco de dados pode ser acessado por diferentes usuários que possuem permissões e restrições diferentes dependendo da responsabilidade atribuída. O usuário é uma conta que possui acesso e permissões restritas a determinada parte de um banco de dados. O schema (esquema) é um espaço de nomes (spacename) que serve para organizar o banco de dados, ele funciona agrupando objetos relacionados com o mesmo nome, para evitar conflitos de nomeações, como exemplo, tabela com mesmo nome, além disso, também pode ser usado para definir diferentes níveis de permissões para usuários ou grupos de usuários.

**QUESTÃO 12:** Eles são importantes em diversas funções, como exemplo, organização, agrupando as tabelas, funções logicamente com um nome. Para segurança, ele permite que apenas usuários autorizados tenham acesso a determinados dados, criando níveis de permissões. Para desenvolvimento, permitindo que diferentes equipes trabalhem em grandes bancos de dados sem interferir o processo um do outro.

**QUESTÃO 13:** Se não definir um schema na criação de um banco de dado, eles serão armazenados no schema padrão do PostgreSQL, “public”. Isso é ruim pois além de não ter um sistema de padronização e flexibilidade do banco de dados, que ajuda drasticamente na hora de se trabalhar em grandes bancos de dados, a segurança também é ruim, pois qualquer usuário pode acessar e modificar os dados dentro desse schema.

**QUESTÃO 14:** Não tenho uma sugestão para compartilhar sobre o projeto, achei ele bastante completo e bem detalhado sobre os objetivos desejados.