SISTEMA DE banco de dados

PLANEJAMENTO DE JOGOS DIGITAIS
PARA MULTIPLATAFORMAS

No desenvolvimento de jogos digitais, podemos acabar nos fixando demais nos algoritmos que modelam os comportamentos de personagens e os trechos de códigos, que criam as regras do jogo. Porém, para um jogo digital funcionar corretamente, se faz necessária a manipulação e o armazenamento de grande quantidade de dados, bem como o compartilhamento de tais informações com outros sistemas.

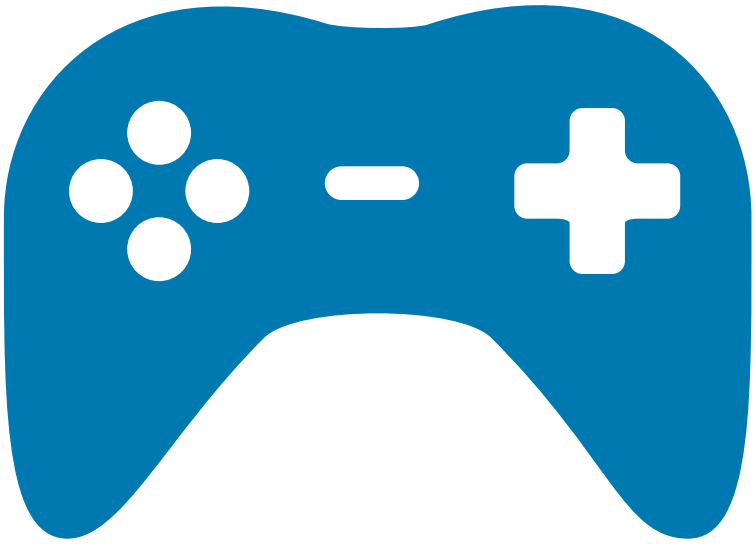
PARA
REFLETIR

Quais informações vocês acham que devem ser armazenadas?

Para exemplificar, vamos considerar um jogo hipotético de RPG *on-line*, em que os jogadores podem cooperar uns com os outros, batalhando contra monstros do mundo virtual, ou guerreando entre si, com seus próprios personagens. Cada um dos personagens possui diversas características, a serem definidas pelo jogador – classe, raça, atributos, aparência, equipamentos, poderes etc.

Quais as entidades, ou elementos do jogo, podemos destacar no exemplo anterior como relevantes para armazenar?

Para os jogadores interagirem no mundo virtual, seria necessário que trocassem informações sobre características de seus personagens e as ações que eles estão tomando no jogo. As informações dos personagens, geralmente, são modificadas pelos usuários, conforme as ações ocorrem no jogo, e precisam ser armazenadas para que todos os jogadores envolvidos possam acessar os dados, posteriormente, e modificá-los, caso seja necessário.



O que aconteceria se cada um dos usuários armazenasse as informações de forma separada?

Se cada um dos usuários armazenar de forma separada, em seu computador, os dados de personagens de outros jogadores, teremos um problema chamado *redundância de dados não controlada*, situação em que são geradas cópias, desnecessárias, de um mesmo conjunto de informações, em diversos sistemas diferentes. Isso faz com que a responsabilidade pela manutenção e sincronia entre essas diversas cópias esteja com o usuário e não com um software.

Pense sobre o problema da redundância e identifique outro problema resultante desta abordagem.

Agora, avaliem o problema exposto anteriormente em situações de centenas de jogadores, nos quais, um usuário precisaria enviar, manipular, receber e armazenar informações sobre outros personagens, com as outras centenas de usuários que estão *on-line*, e o mesmo valeria para os outros jogadores. Claramente, problemas poderiam surgir e numa escala mais preocupante. Se supomos um jogo *multiplayer on-line*, muitos usuários poderiam estar requisitando e modificando um mesmo conjunto de dados, ao mesmo tempo, e algumas dessas modificações poderiam não ser aplicadas no conjunto de dados de um ou mais usuários. As informações sobre um mesmo personagem poderiam ser armazenadas, de forma inconsistente, entre diferentes jogadores e, no fim, gerar erros no jogo.

Figura 1: Componente multijogador on-line do jogo de RPG Dragon Age: Inquisition. Diversos usuários trocam informações de dados sobre seus personagens, e suas ações, em tempo real.

Quando avaliamos um contexto de jogo de RPG *on-line*, como o que observamos anteriormente, as informações do jogo digital representam, literalmente, a ideia de um banco de dados: uma coleção de dados inter-relacionados, representando informações sobre um domínio específico, neste caso, os dados dos usuários, personagens e monstros, que compõe o mundo virtual.

Exercitem com algum jogo que estejam jogando, ou gostem. Quais informações seriam relevantes serem representadas?

Continuando a jornada sobre banco de dados, pudemos até aqui lidar com uma grande quantidade de informações, e que os problemas consequentes da manipulação deste banco de dados, de forma segura e eficaz, são complexos e exigem uma série de análises e cuidados, para que tudo ocorra corretamente.

Desta forma, surgiram, por volta da década de 1970, os primeiros sistemas gerenciadores de banco de dados (SGBD).
Um sistema de gerenciamento de banco de dados é um software com recursos específicos, que facilita a manipulação das informações dos dados e o desenvolvimento de programas e aplicativos.
O SGBD é uma coleção de programas, que permite que usuários criem e mantenham bancos de dados. Você já deve conhecer alguns destes exemplos de SGBDs: PostgreSQL, MySQL, Oracle, MongoDB, DB2, Microsoft Access e Microsoft SQL Server.

CURIO-SIDADE

Que tal dar uma olhada nas características de um SGBD no site do proprietário? Podemos começar pelo MySQL, disponível em www.mysql.com.

A seguir, veremos as questões referentes ao projeto de um banco de dados, que se dá em três etapas: modelagem conceitual, projeto lógico e projeto físico. Você irá perceber que os SGBDs não se aplicam a todos os modelos lógicos e a sua escolha do SGBD será em função do tipo de representação lógica que utilizará para representar os elementos de um jogo.

SAIBA+

Para saber mais sobre o ranking de popularidade de SGBDs existentes, acesse:

www.itcareersuccess.com/tech/database.htm

Modelo conceitual

O modelo conceitual descreve a estrutura de um banco de dados de forma independente de um SGBD particular, ou seja, o modelo registra quais dados podem aparecer no banco dados, mas não indica como esses dados estão armazenados em nível de SGBD. A técnica de modelagem conceitual mais difundida é o diagrama entidade-relacionamento (diagrama E-R). A figura 2 apresenta uma descrição parcial para nosso jogo de RPG *on-line* utilizando um diagrama E-R. Este modelo indica que o banco de dados contém informações sobre as entidades *jogador e personagem* (retângulos no diagrama), indicando quais dados são armazenados de cada entidade.

O relacionamento ***possui*** (losango no diagrama) indica que os dados do ***jogador*** e do ***personagem*** precisam ser associados um ao outro, indicando que um (1) jogador possui N (vários) personagens, e que cada personagem possui somente um jogador. O diagrama também indica

quais atributos possuem cada uma das entidades, no caso de jogador, o diagrama E-R indica que é necessário guardar os dados de *login*, senha *e-mail* do jogador. Para as entidades *personagem*, é necessário manter informações sobre nome, classe e raça do personagem.

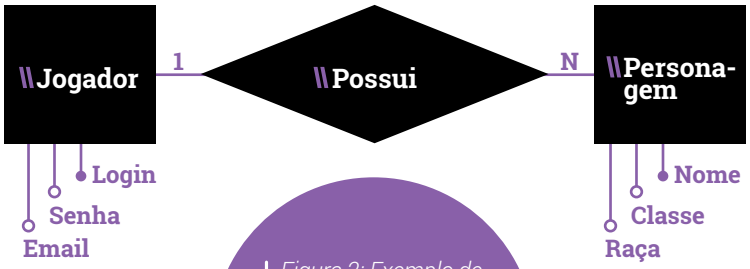
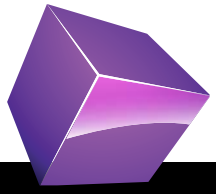


Figura 2: Exemplo de modelo conceitual usando diagrama de entidade-relacionamento.

Projeto Lógico

A etapa de projeto lógico transforma o modelo conceitual criado em uma definição de como o banco de dados será implementado em um SGBD específico. Assim, o modelo lógico depende do tipo de SGBD que será usado. Três tipos de banco de dados são bem conhecidos: modelo hierárquico, modelo relacional e modelo orientado a objetos.



MODELO DE BANCO DE DADOS HIERÁRQUICO

O modelo hierárquico foi o primeiro a ser reconhecido como um modelo de dados. Esse modelo organiza seus dados conectando *registros* diferentes, por meio de *ligações*, de tal modo que cada tipo de registro tenha apenas um pai, formando, assim, uma estrutura parecida como uma árvore com ramificações, a partir de uma raiz. Cada registro é formado por uma coleção de campos, ou atributos, e cada campo contém, somente, um valor de dado. A figura 3 representa o diagrama, em árvore, do diagrama E-R da figura 2. A figura 4 é um exemplo de banco de dados estruturado de forma hierárquica. Atualmente, os modelos hierárquicos estão em desuso devido às dificuldades técnicas de se implementar relacionamentos complexos entre dados.

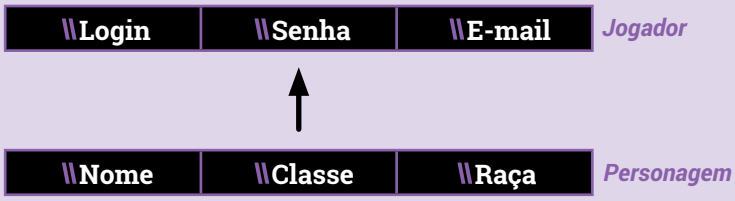
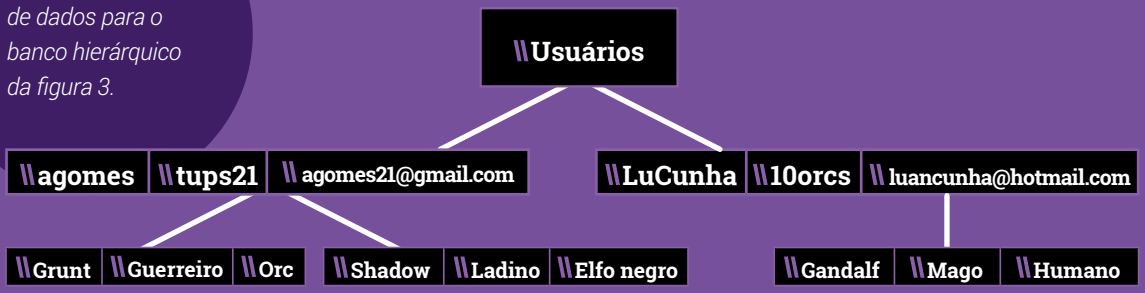


Figura 3: Diagrama em árvore do modelo entidade-relacionamento da figura 2.

Figura 4: Exemplo de dados para o banco hierárquico da figura 3.



MODELO DE BANCO DE DADOS RELACIONAL

Um banco de dados relacional modela o conjunto de dados, de uma forma que eles sejam percebidos pelo usuário, como tabelas, ou mais formalmente, como relações. Uma tabela é um conjunto não ordenado de linhas, também conhecidas como tuplas. Cada linha da tabela é composta por uma série de valores. Por exemplo, nas figuras 5a e 5b temos a tabela para as entidades *jogador* e *personagem*, respectivamente, onde cada coluna diz respeito a um atributo dos dados das entidades descritas no diagrama E-R da figura 2. A figura 5c é uma representação, em tabela, do relacionamento *possui* existente entre as entidades *jogador* e *personagem*. É através desta tabela que podemos saber qual personagem pertence a qual jogador.

(a) Tabela jogador

Login	Senha	E-mail
agomes	tups21	agomes21@gmail.com
LuCunha	10orcs	luancunha@hotmail.com

(b) Tabela personagem

Nome	Classe	Raça
Gandalf	Mago Cinza	Humano
Shadow	Ladino assassino	Elfo Negro
Grunt	Guerreiro berserk	Orc

(c) Tabela possui

Jogador	Personagem
LuCunha	Gandalf
agomes	Shadow
agomes	Grunt

Figura 5: Tabelas para um banco de dados relacional. (a) Tabela para a entidade jogador. (b) Tabela para entidade personagem. (c) Tabela para o relacionamento possui.

MODELO DE BANCO DE DADOS ORIENTADO A OBJETOS

O modelo de banco de dados orientado a objetos aprimora o modelo relacional, fornecendo uma série de recursos que facilitam a criação de tipos de dados mais complexos. Assim como no modelo relacional, o modelo orientado a objetos utiliza tabelas para organizar as informações, porém podemos representar **informações mais complexas em um atributo.**

Digamos que fosse necessário subdividir o atributo *classe* da entidade *personagem*, indicando a classe principal e a sua *subclasse* de forma separada, seria possível criar uma nova coluna chamada subclasse na tabela, porém uma série de fatores tornaria a programação desse banco de dados relacional mais complexa. Utilizando um

banco de dados orientado a objetos é possível indicar que certo atributo é *multivalorado*, isto é, pode possuir mais de um valor. A figura 6 representa a entidade *personagem* com duas modificações, agora tanto o atributo classe como o atributo *raça* são multivalorados. Por exemplo, o personagem *Grunt* tem classe principal guerreiro e subclasse berserk. Já o personagem *Gandalf* pertence a raça dos humanos sendo da sub-raça nortenho.

Figura 6: Tabela de um banco de dados orientado a objetos para a entidade personagem com os atributos classe e raça sendo multivalorados.

Nome	Classe	Raça
Gandalf Shadow Grunt	(Mago) (Ladino assassino) (Guerreiro, berserk)	(Humano, nortenho) (Elfo, negro) (Orc)

Projeto Físico

No projeto físico, o modelo do banco de dados é enriquecido com detalhes que determinam questões de desempenho do banco de dados, mas não influenciam a funcionalidade da implementação, e depende do SGBD escolhido. Na prática, o projeto físico é um processo contínuo, que ocorre mesmo depois do banco de dados já estar implementado e em funcionamento