Java Advanced

Prof. Dr. Marcel Stefan Wagner

Aula 13 – *Flyway*

Introdução ao *Flyway*

Aula 13

Prof. Dr. Marcel Stefan Wagner

Curso **Análise e Desenvolvimento de Sistemas**

FIAP – Faculdade de Informática e Administração Paulista

Campus – **Lins** / **Paulista**

Curso de **graduação – 2025/2**

Turma – **TDS**

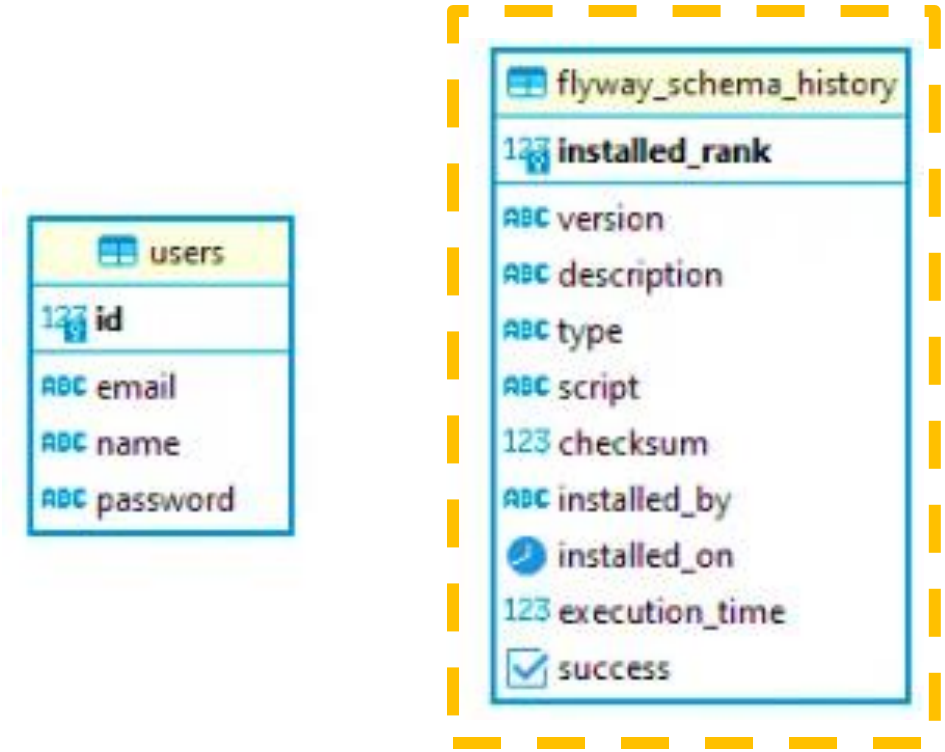
Flyway

Flyway

O que é *Flyway*?

Flyway é uma ferramenta de migração de Bancos de Dados (BD) que automatiza e versiona alterações de esquema (*scripts* SQL ou Java), mantendo o BD sincronizado com a aplicação e rastreando o histórico de execuções numa tabela *flyway_schema_history*.

É ideal para o ecossistema Java com *Spring Boot* ou *Quarkus*, pois ele simplifica a gestão de banco de dados, permitindo criar, atualizar e, em alguns casos, até reverter alterações, tudo isso de forma automática e com controle de versão.



Por que usar uma ferramenta para versionar meu banco de dados?

As aplicações Java com *Spring*, em sua grande maioria, usam o ORM *Hibernate* para gerar o *schema* do BD mapeando os seus objetos para tabelas, sendo assim, **cada alteração, criação e exclusão em nossas entidades refletem de maneira automática no nosso banco de dados**.

Pensando-se em ambiente de desenvolvimento e até no ambiente de testes, isso é perfeito, pois economiza um bom tempo e “não precisamos saber a mágica por trás”, que ocorre de forma automática. Mas quanto ao ambiente de produção, não é um pouco arriscado que essas mudanças automáticas ocorram?

Por que usar uma ferramenta para versionar meu banco de dados?

Olhando a documentação do Hibernate temos o seguinte “alerta”:

Como diz na documentação, o *Schema Generation* em produção não é uma boa prática. O recomendado é fazer o uso de *scripts* incrementais. Então basta usarmos qualquer ferramenta de administração de banco dados como o *DBeaver* ou *DataGrip*, por exemplo, e fazer os *scripts* de criação na mão, certo? Isso até resolveria, mas como saber qual *script* foi ou não executado? Também temos o problema de que, quando a equipe é grande e tem muitas mudanças para fazer no *schema*, acarretaria facilmente em dezenas de *scripts* SQL onde a cada nova alteração todos os membros da equipe vão ter que rodar todos os *scripts* manualmente.

Por que usar uma ferramenta para versionar meu banco de dados?

O problema fica ainda pior quando entra um novo membro na equipe, onde ele vai ter que fazer isso “um a um” até chegar ao final. O mesmo problema acontece quando ainda não existe ambiente de teste e queremos criar um com a estrutura e dados idênticos aos de produção, teríamos dezenas de *scripts* para serem executados um de cada vez seguindo a ordem de criação.

Agora que entendemos os problemas, podemos resolvê-los com o uso do *Flyway*.

Como o *Flyway* funciona?

- **Rastreamento do Histórico**: O *Flyway* cria e gerencia uma tabela (*flyway_schema_history* por padrão) que registra todos os *scripts* de migração executados e seu *status*.

- **Escaneamento de Migrações**: A ferramenta escaneia um diretório (como **db.migration** no *Spring Boot*) em busca de *scripts* SQL ou Java que sigam um padrão de nomenclatura com numeração sequencial.

- **Aplicação de Novas Migrações**: Durante a inicialização da aplicação, o *Flyway* compara os *scripts* encontrados com o histórico, aplicando apenas os *scripts* novos ou não executados para manter o banco de dados atualizado.

Principais funcionalidades

- **Versionamento de Banco de Dados**: Mantém um registro de todas as alterações do esquema, permitindo a sincronização com a versão da aplicação.

- **Automatização**: Aplica os *scripts* de forma automática quando a aplicação é iniciada ou através de outras integrações.

- ***Scripts* em SQL ou Java**: Permite escrever migrações tanto em SQL quanto em Java para flexibilidade.

Principais funcionalidades

- **Controle de Versão**: Garante que apenas as mudanças previstas e executadas sejam aplicadas no banco.

- **Reversão (*Rollback*)**: Possui um mecanismo de reversão para desfazê-las caso necessário, embora seja para casos raros.

Vantagens de uso

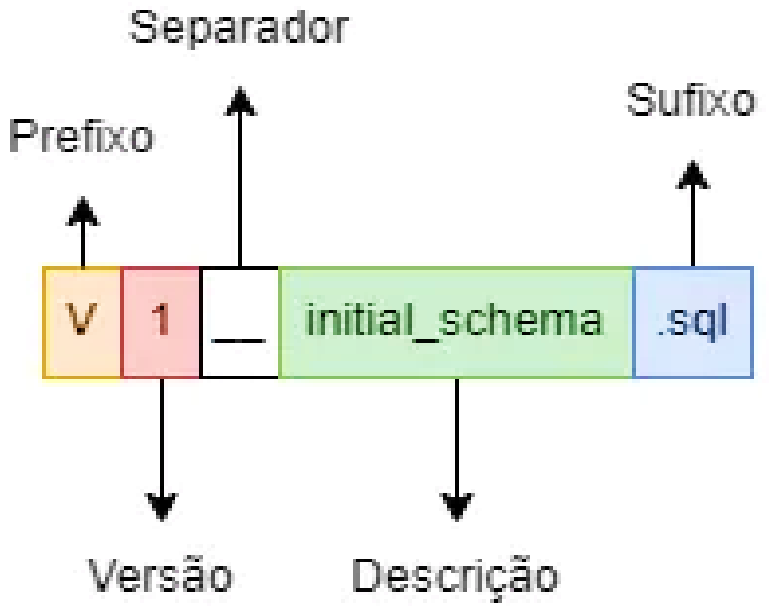
- **Simplicidade**: É uma ferramenta que favorece a convenção em vez da configuração complexa.

- **Integração com Java**: Integra-se facilmente em projetos Java como o *Spring Boot* e o *Quarkus*, utilizando dependências e configurando *plugins*.

- **Multiplataforma**: Pode ser usado em diversos ambientes (por exemplo: em nuvem ou local) e com diferentes bancos de dados.

Formatação

- **Prefixo**: deve iniciar com V em maiúsculo.
- **Versão**: número da versão, deve ser incremental e pode ser separado por **.** ou **_** (como 1.1, 1_1, 2, 2.2).
- **Separador**: usamos o **__** (2 *underscores*).
- **Descrição**: breve descrição do que esta sendo alterado.



- Exemplo: **V1.0__create_user.sql**
- Exemplo: **V202401252036__update_users.sql** (data e horário)

Referências

George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. **Sistemas Distribuídos: Conceitos e Projeto**. Bookman Editora, 5 edition, 2013.

Harvey M Deitel, Paul J Deitel, David R Choffnes, et al. **Sistemas Operacionais**. Pearson/Prentice Hall, 3 edition, 2005.

Maarten Van Steen and A Tanenbaum. **Sistemas Distribuídos: Princípios e Paradigmas**. Pearson/Prentice Hall, 2 edition, 2007.

Harvey M Deitel and Paul J Deitel. **Java, como programar**. Ed. Pearson/Prentice Hall, 8 edition, 2010.

StackOverflow. Disponível em: <https://pt.stackoverflow.com/>. Acesso em: abril de 2023.

GAMMA, Erich et al. **Elements of Reusable Object-Oriented Software**. Design Patterns, 1995.

COOPER, James William. **Java design patterns: a tutorial**. 2000.

GUERRA, Eduardo. **Design Patterns com Java: Projeto orientado a objetos guiado por padrões**. Editora Casa do Código, 2014.

Referências

AWS Amazon – O que é RESTful. Disponível em: <https://aws.amazon.com/pt/what-is/restful-api/#:~:text=Representational%20State%20Transfer%20(REST)%20is,complex%20network%20like%20the%20internet.>. Acesso em: maio de 2023.

InfoQ. Disponível em: <https://www.infoq.com/minibooks/emag-03-2010-rest>. Acesso em: maio de 2024.

Michelli Brito. Disponível em: <https://www.youtube.com/@MichelliBrito>. Acesso em: abril de 2024.

Obeautifulcode. Disponível em: <http://blog.obeautifulcode.com/API/Learn-REST-In-18-Slides/>. Acesso em: junho de 2022.

Berkeley. Disponível em: <http://courses.ischool.berkeley.edu/i290-rmm/s12/slides/Lecture3%20REST.pdf>. Acesso em: maio de 2023.

Trybe. Disponível em: <https://blog.betrybe.com/tecnologia/deploy/>. Acesso em: julho de 2024.

Obrigado!

Contato: profmarcel.wagner@fiap.com.br

Cursos:
Tecnologia em Análise e Desenvolvimento de Sistemas (TDS)
Tecnologia em Defesa Cibernética (TDC)
Engenharia de Software (ES)