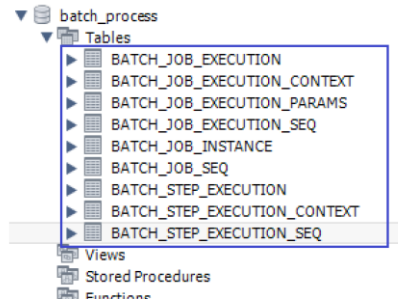




os metadados do Spring Batch

Se eu for no **banco de dados** dedicado ao **Spring Batch** (*mudei para o mysql*) e olhar as tabelas, eu vou encontrar um monte de tabelas referentes ao batch, tabelas essas que podem responder diversas dúvidas:

```
Application.java x pom.xml (lote) x application.properties x
spring.datasource.url=${DB_URL}
spring.datasource.username=${DB_USERNAME}
spring.datasource.password=${DB_PASSWORD}
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.batch.jdbc.initialize-schema=always
```



principais metadados do Spring Batch

Dúvidas do tipo: “Quantas vezes o batch executou com sucesso/fez uma execução lógica”? Essa dúvida pode ser respondida com um select na tabela de **BATCH_JOB_INSTANCE**:

```
1 • SELECT * FROM batch_process.BATCH_JOB_INSTANCE;
```

JOB_INSTANCE_ID	VERSION	JOB_NAME	JOB_KEY
1	0	helloJob	d41d8cd98f00b204e9800998ecf8427e
NULL	NULL	NULL	NULL

```
@EnableBatchProcessing
@Configuration
public class BatchConfig {

    private final JobBuilderFactory jobBuilderFactory;

    private final StepBuilderFactory stepBuilderFactory;

    public BatchConfig(JobBuilderFactory jobBuilderFactory,
                       StepBuilderFactory stepBuilderFactory) {
        this.jobBuilderFactory = jobBuilderFactory;
        this.stepBuilderFactory = stepBuilderFactory;
    }

    @Bean
    public Job helloJob() {
        return jobBuilderFactory
            .get("helloJob")
            .start(printHelloStep())
            .build();
    }
}
```

“Quantas vezes o job executou ao todo, incluindo falhas?” Uma dúvida respondida com um select na tabela **BATCH_JOB_EXECUTION**, aqui tem **informações bem relevantes**, como o status e o tempo de execução:

```
1 • SELECT * FROM batch_process.BATCH_JOB_EXECUTION;
```

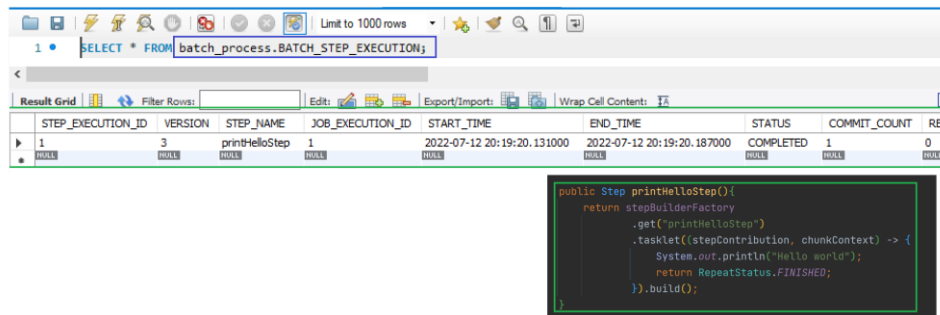
JOB_EXECUTION_ID	VERSION	JOB_INSTANCE_ID	CREATE_TIME	START_TIME	END_TIME	STATUS	EXIT_CODE
1	2	1	2022-07-12 20:19:20.000000	2022-07-12 20:19:20.074000	2022-07-12 20:19:20.214000	COMPLETED	COMPLETE
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

O **Spring Batch** salva algumas informações automaticamente, mas também é possível salvarmos nossas próprias informações relevantes para um job (como uma RN por exemplo). Essas informações fornecem contexto e, portanto, ficariam salvas na tabela **BATCH_JOB_EXECUTION_CONTEXT**:

```
1 • SELECT * FROM batch_process.BATCH_JOB_EXECUTION_CONTEXT;
```

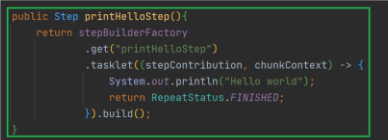
JOB_EXECUTION_ID	SHORT_CONTEXT	SERIALIZED_CONTEXT
1	{@class:"java.util.HashMap"}	NULL
NULL	NULL	NULL

Um **job** é composto de vários steps, portanto é natural que os **Steps** também tenham seus metadados salvos, como por exemplo a execução lógica de um step que fica salvo na tabela **BATCH_STEP_EXECUTION**:



The screenshot shows a database query tool interface. At the top, a SQL query is entered: `SELECT * FROM batch_process.BATCH_STEP_EXECUTION;`. Below the query, a "Result Grid" displays the results of the query. The grid has columns: STEP_EXECUTION_ID, VERSION, STEP_NAME, JOB_EXECUTION_ID, START_TIME, END_TIME, STATUS, COMMIT_COUNT, and REASON_CODE. The first row shows data for a completed step.

STEP_EXECUTION_ID	VERSION	STEP_NAME	JOB_EXECUTION_ID	START_TIME	END_TIME	STATUS	COMMIT_COUNT	REASON_CODE
1	3	printHelloStep	1	2022-07-12 20:19:20.131000	2022-07-12 20:19:20.187000	COMPLETED	1	0



```
public Step printHelloStep(){
    return stepBuilderFactory
        .get("printHelloStep")
        .tasklet((stepContribution, chunkContext) -> {
            System.out.println("Hello world");
            return RepeatStatus.FINISHED;
        }).build();
}
```



qual a pira dos metadados do **Spring Batch**?

Bom, a esse ponto fica lógico que os metadados do Spring batch vão fornecer um controle sobre os jobs e steps, fazendo com que seja possível identificar com facilidade onde ocorreram erros no processo, seja no job ou no step.