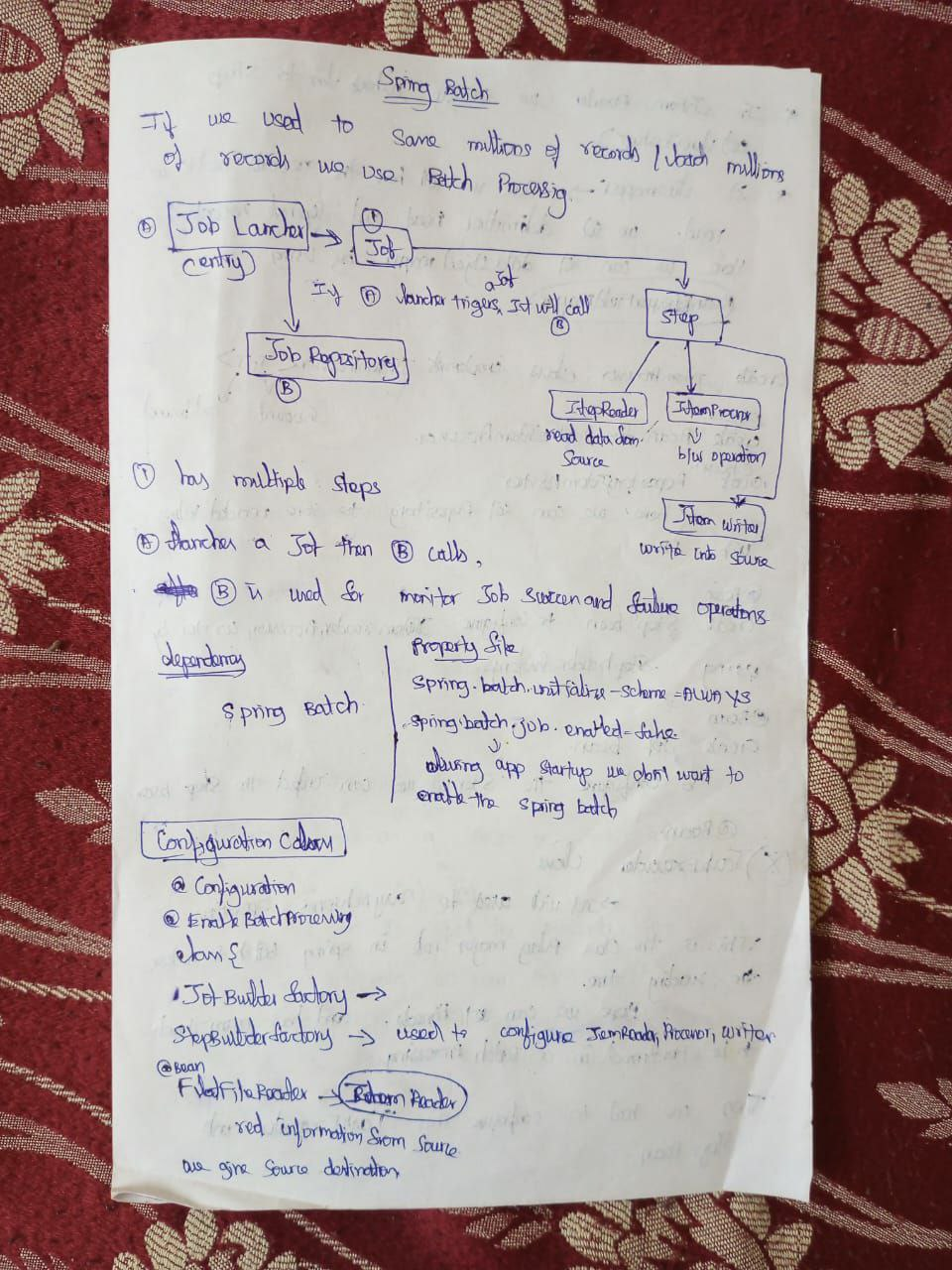
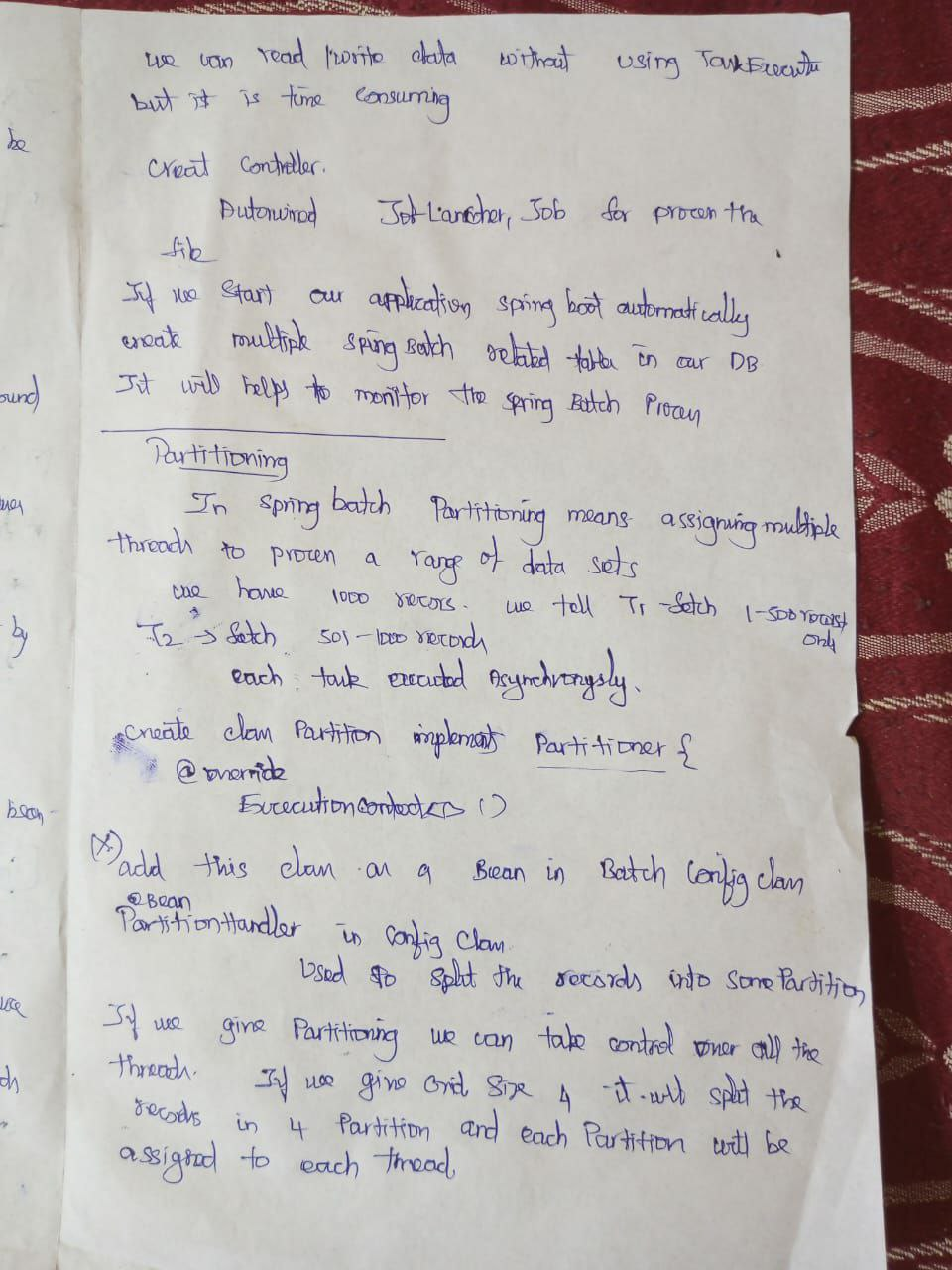
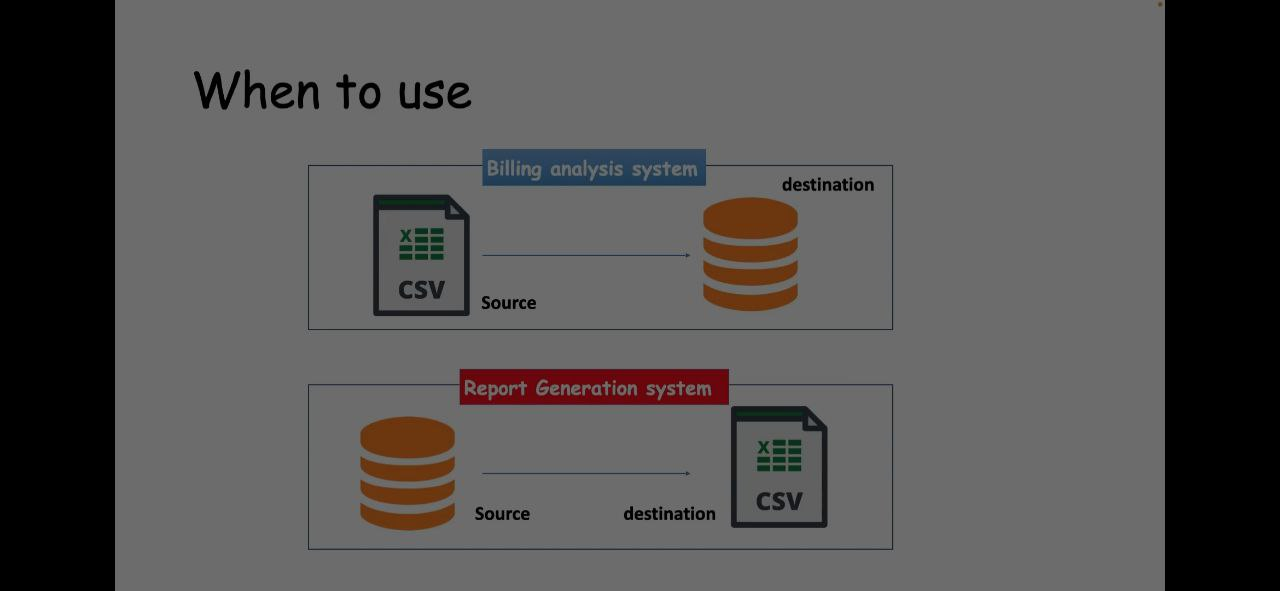
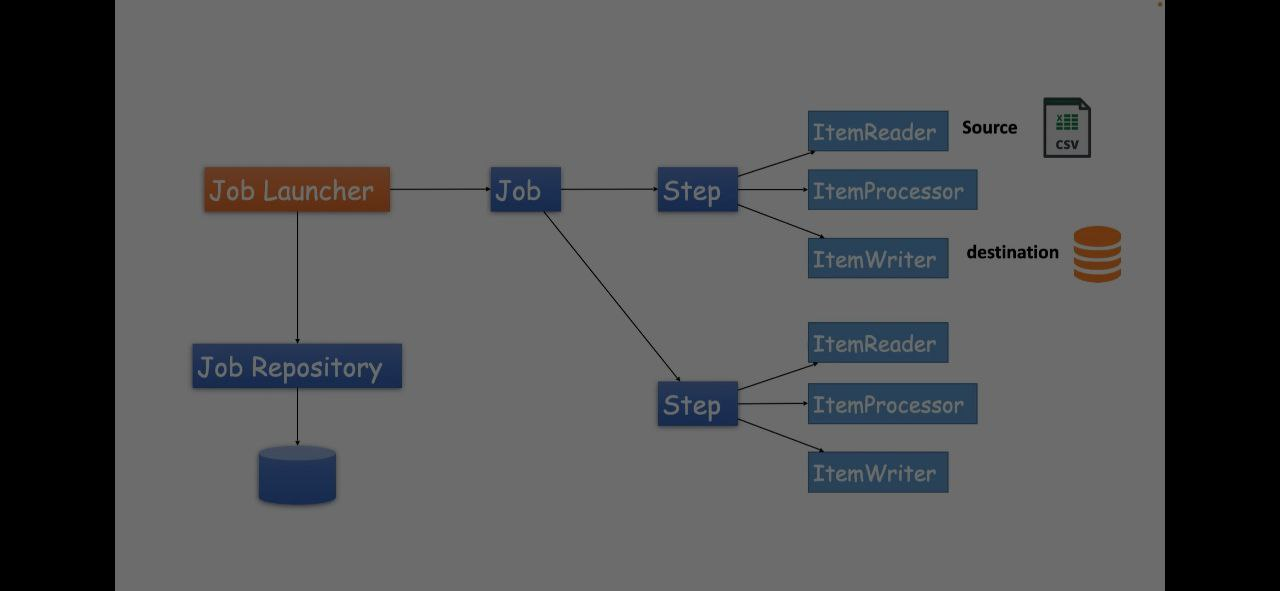
**Spring Batch**

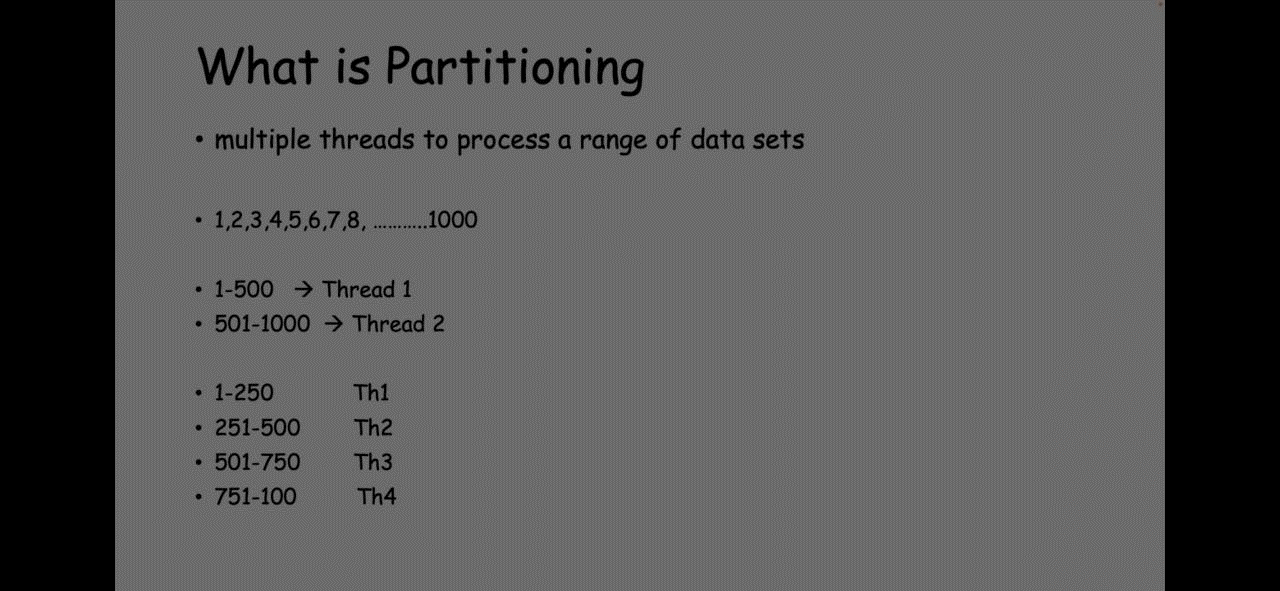


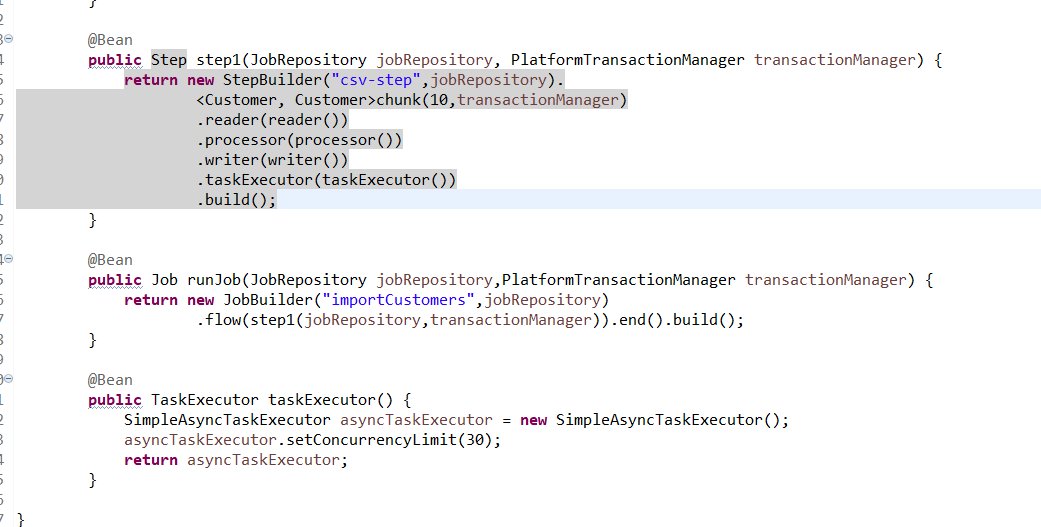










****

When processing 1000 records in a CSV file using the chunk method with ItemReader, ItemProcessor, ItemWriter, and TaskExecutor in Spring Batch, the records are handled as follows:

**1. Chunk-Oriented Processing**

**Chunk Size (10)**:

* You have defined a chunk size of 10 in your configuration.
* This means that Spring Batch will process 10 records at a time in one transaction.

**2. Handling Records with ItemReader, ItemProcessor, ItemWriter, and TaskExecutor**

**Step Execution**:

* **Step Initialization**: The step starts and initializes the FlatFileItemReader, CustomerProcessor, and RepositoryItemWriter.
* **Reading**: The FlatFileItemReader reads 10 records from the CSV file.
* **Processing**: Each of the 10 records is processed by the CustomerProcessor.
* **Writing**: The processed 10 records are written to the repository by the RepositoryItemWriter.
* **Repeat**: The step repeats this process until all 1000 records are processed.

**Detailed Breakdown**

1. **Reading**:
   * The FlatFileItemReader reads records from the CSV file in chunks of 10 records each. For 1000 records, it will read the records in 100 chunks (1000 / 10 = 100).
2. **Processing**:
   * For each chunk of 10 records, the CustomerProcessor processes each Customer object. This processing can include transformations, validations, or any other business logic.
3. **Writing**:
   * After processing, the RepositoryItemWriter writes the chunk of 10 processed records to the repository in a single transaction. This ensures that either all 10 records are written successfully, or none are written (in case of an error).

**Multi-threading with TaskExecutor**

**TaskExecutor (SimpleAsyncTaskExecutor)**:

* The SimpleAsyncTaskExecutor allows for concurrent processing of chunks.
* You have set the concurrency limit to 30, which means up to 30 chunks can be processed simultaneously.

**Handling Records Concurrently**:

* **Concurrent Chunks**: With 30 concurrent threads, up to 30 chunks (each of 10 records) can be processed at the same time.
* **Efficiency**: This greatly improves the efficiency and speed of processing large datasets.

**Example Execution Flow**

1. **Initial 30 Chunks**:
   * The TaskExecutor starts by reading the first 30 chunks (0-9, 10-19, ..., 290-299).
   * Each chunk is processed and written concurrently.
2. **Subsequent Chunks**:
   * As soon as a chunk is completed, the next available chunk is picked up by an available thread.
   * This continues until all chunks are processed.
3. **Completion**:
   * Once all 1000 records are processed in chunks of 10, the step is completed, and the job moves to the next step (if any) or finishes.

**Summary of Flow**

1. **Initialization**:
   * Step and task executor are initialized.
2. **Reading**:
   * Chunks of 10 records are read concurrently up to the concurrency limit (30 chunks at a time).
3. **Processing**:
   * Each chunk of 10 records is processed concurrently by available threads.
4. **Writing**:
   * Processed chunks are written to the repository concurrently.
5. **Iteration**:
   * The next set of chunks is read, processed, and written until all records are handled.
6. **Completion**:
   * The job completes successfully after processing all 1000 records.

**Example Timeline for 1000 Records**

1. **Start**:
   * Threads 1-30 start processing chunks 1-30.
2. **Process**:
   * As threads complete chunks, they pick up the next available chunks (e.g., thread 1 finishes chunk 1, picks up chunk 31).
3. **End**:
   * Threads continue until all 1000 records are processed.

In summary, with a chunk size of 10 and a concurrency limit of 30, the 1000 records in the CSV file will be efficiently read, processed, and written in a highly concurrent manner, significantly speeding up the batch processing job.

Need to study Partitioning for improve more better performance .