Java Stream: streams are wrappers around a data source, allowing us to operate with that data source and making bulk processing convenient and fast. The reason to use Stream is to make it easy to process large amount of data.

public void whenFilterEmployees\_thenGetFilteredStream() {

Integer[] empIds = { 1, 2, 3, 4 };

List<Employee> employees = Stream.of(empIds)

.map(employeeRepository::findById)

.filter(e -> e != null)

.filter(e -> e.getSalary() > 200000)

.collect(Collectors.toList());

assertEquals(Arrays.asList(arrayOfEmps[2]), employees);

}

This example shows that wrap all the employee id into a stream and then map all these employee ids into a list of employee objects. During this process, filter function is used twice to remove the null Employee objects and remove the employee objects whose salary is lower than 200000. Then convert the stream into a list of Employee. This example shows that Stream is a wrapper around a data source. It is really convenient for us to operate on all the data by using Stream.