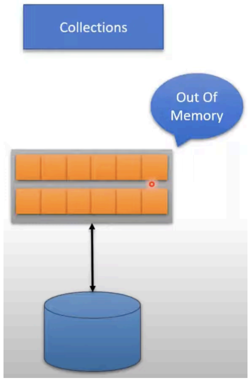
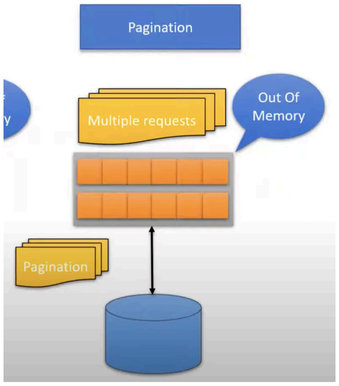
1) What are Streams in Mule 4 Runtime?

- **Collections:**
  When we query a database using JDBC, the result is stored in heap memory. If the data is small, it can fit into collections like arrays or ArrayLists without any issue. However, if the data is very large and there isn't enough space in heap memory, it could cause an OutOfMemoryError.

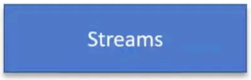
  Using collections to load all the data at once is not a good approach when dealing with large datasets. If the data is small, this method works fine, but for huge amounts of data, it can lead to problems like memory overload.



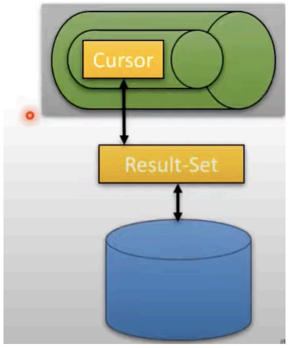
- **Pagination:**
  In this method, we display records 1 to 10 on page 1, records 11 to 20 on page 2, and so on. This allows us to retrieve data one page at a time. However, if many requests come in to access page 1, those pages will be loaded into memory, which could cause an out of memory error.



- **Streams:**
  When we query a database, we get a result set. This result set includes a cursor, which is like a pointer to a record in the database. The cursor is wrapped in an object called a stream, which works like an iterator. Instead of getting all the data at once, we can get one row at a time. After processing one row, we can ask for the next row, and so on. This way, the data comes to us only when we need it, instead of all at once.



Streams



2) What are different typesStreams in Mule 4 Runtime?

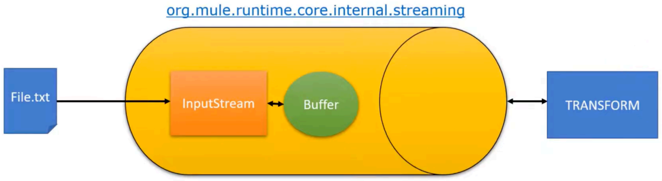
- **Non Repeatable Stream:**
  In this, you will use the stream of data, and it won't be available to the next processor. It's memory-efficient because it reads the data and then removes it.

- **Repeatable in MemoryStream:**
  When we read data from a file, it's stored in memory (heap memory) of the application. This method is useful when the file size is small.

  The process starts by reading the file into an InputStream object. This InputStream has a buffer, which temporarily holds the data. The InputStream is then wrapped in a stream to make the data available for the next step in the process.

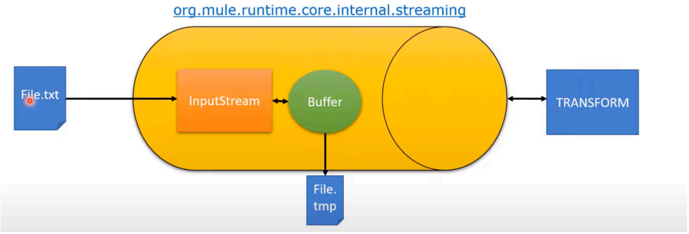
  The buffer has three important settings:

  - **Initial buffer size:** This is the starting amount of data the buffer can hold.
  - **Buffer size increment:** This determines how much the buffer size will grow the next time we need to load more data.
  - **Max buffer size:** This is the maximum amount of data the buffer can hold.
  - **Buffer unit**: This defines the unit of measurement for the buffer size, such as bytes or kilobytes.



- **Repeatable file Store Stream:**
  Data is first loaded into a buffer. When the buffer is full, the data is saved to a temporary file. Then, the buffer loads new data and stores it in the temp file, and this process continues. This is like creating a copy of the incoming data.

  When we need to process the data, the buffer loads it from the temp file and sends it to the next step in the process.



3) **What is Response Streaming Mode?**

Response Streaming Mode controls how the server sends data in response to a request:

- **AUTO (default):** The server will use Content-Length (size of the data) if it knows the size of the response. If the size is unknown, it will send the data in chunks (using Transfer-Encoding: chunked).
- **ALWAYS:** The server will always send the response in chunks, no matter if it knows the size of the data or not.
- **NEVER:** The server will always use Content-Length, even if it has to first process the data to figure out its size. It won't send data in chunks.

4) What is Reject Invalid Transfer Encoding Headers?

Transfer-Encoding headers are used in HTTP to specify how data is transferred between the client and the server. Valid options for these headers include chunked, deflate, compress, identity, and gzip (they're not case-sensitive, so "Chunked" and "chunked" are the same) Enabling the "Reject invalid transfer encoding" setting in the HTTP Listener ensures only valid headers are accepted, enhancing security and preventing misuse

5) What are valid transfer encoding headers?

- **Chunked**: Sends data in small pieces instead of all at once.
- **Deflate**: Compresses data using a standard algorithm to save bandwidth.
- **Compress**: An older, less common method for data compression.
- **Identity**: Sends data in its original, uncompressed form.
- **Gzip**: A popular and efficient method for compressing data to make transfers faster and smaller.

6) What is Cross-Origin Resource Sharing (CORS) ?

Cross-Origin Resource Sharing (CORS) is a way to let a website tell the browser if it’s okay to share its data with another website. It uses extra HTTP headers for this.

If two websites are from the same origin (same domain, protocol, and port), they can share data without any issues. But if the websites are from different origins, they need to follow the CORS rules to allow sharing.

The CORS mechanism was introduced to allow controlled communication between websites with:

Different domains (e.g., example.com and test.com),
Different subdomains (e.g., app.example.com and api.example.com),
Different ports (e.g., example.com:8080 and example.com:3000), or
Different protocols (e.g., http and https)

7) What is Redelivery policy in the HTTP Listener?

A redelivery policy in Mule controls how many times Mule retries a message if it fails to process.

When a message comes in, Mule tries to process it.
If it fails, Mule retries based on the policy's rules.
If it still fails after the allowed retries, Mule stops trying and throws an error.
This prevents wasting resources on messages that can't be processed.

- **Max Redelivery Count:** How many times Mule will try to process a message again if it fails. Default is 5 tries.
- **Use Secure Hash:** Whether Mule will use a secure method to track messages. Default is True.
- **Message Digest Algorithm:** The method used to create a unique ID for each message. Default is SHA-256.
- **ID Expression:** Custom rules to track redelivered messages (only if Use Secure Hash is off).
- **Object Store:** Where Mule keeps track of how many times a message has been retried.

8) Define HTTP Polling Source?

The HTTP Polling Source is like a timer that keeps checking a website or API at regular intervals. It is the combination of a http request and a scheduler and this is used as the event source.

9) What is HTTP Basic Security Filter?

The HTTP Basic Security Filter ensures that only users with the correct username and password can use the API.

10) What is a Load Static Resource?

HTTP Connector's Load Static Resource allows us to load a specific file like a webpage or script, from the server.

11) What is HTTP Request Connector?

HTTP Request Connector is used to interact with external services over the internet.

12) What is Listener in the HTTP Connector?

Listener in the HTTP Connector is like a receiver that listens for incoming requests from other systems or users.