5-3 消息模块数据模型设计

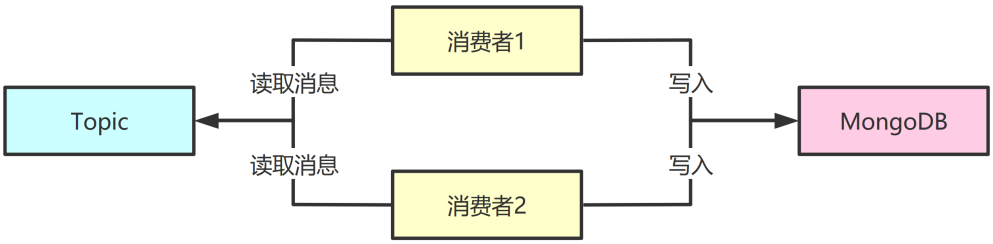
一、创建POJO映射类

MongoDB中没有数据表的概念，而是采用集合（Collection）存储数据，每条数据就是一个文档（Document）。文档结构很好理解，其实就是我们常用的JSON，一个JSON就是一条记录。

1. message集合

集合相当于MySQL中的数据表，但是没有固定的表结构。集合有什么字段，取决于保存在其中的数据。下面这张表格是 `Message` 集合中JSON数据的结构要求。

字段	类型	备注
_id	UUID	自动生成的主键值
uuid	UUID	UUID值，并且设置有唯一性索引，防止消息被重复消费
senderId	Integer	发送者ID，就是用户ID。如果是系统自动发出，这个ID值是0
senderPhoto	String	发送者的头像URL。在消息页面要显示发送人的头像
senderName	String	发送者名称，也就是用户姓名。在消息页面要显示发送人的名字
msg	String	消息正文
sendTime	Date	发送时间



比如说小程序每隔5分钟轮询是否有新的消息，如果积压的消息太多，Java系统没有接收完消息，这时候新的轮询到来，就会产生两个消费者共同接收同一个消息的情况，这会造成数据库中添加了重复的记录，如果每条MQ消息都有唯一的UUID值，第一个消费者把消息保存到数据库，那么第二个消费者就无法再把这条消息保存到数据库，解决了消息的重复消费问题。

创建 `MessageEntity.java` 类，映射 `message` 集合。

```
@Data
@Document(collection = "message")
public class MessageEntity implements Serializable {
    @Id
    private String _id;
    @Indexed(unique = true)
    private String uuid;
    @Indexed
    private Integer senderId;
    private String senderPhoto="https://static-1258386385.cos.ap-beijing.myqcloud.com/img/System.jpg";
    private String senderName;
    @Indexed
    private Date sendTime;
    private String msg;
}
```

2. message_ref集合

虽然 `message` 集合记录的是消息，里面有接受者ID，但是如果是群发消息，那么接受者ID是空值。这时候就需要用上 `message_ref` 集合来记录接收人和已读状态。

字段	类型	备注
_id	UUID	主键
messageId	UUID	message记录的_id
receiverId	String	接收人ID
readFlag	Boolean	是否已读
lastFlag	Boolean	是否为新接收的消息

创建 `MessageRefEntity.java` 类，映射 `message_ref` 集合。

```
@Document(collection = "message_ref")
@Data
public class MessageRefEntity implements Serializable{
    @Id
    private String _id;
    @Indexed
    private String messageId;
    @Indexed
    private Integer receiverId;
    @Indexed
    private Boolean readFlag;
    @Indexed
    private Boolean lastFlag;
}
```

二、MongoDB的联合查询

MongoDB从3.X开始支持集合的连接查询，也就相当于MySQL的表连接。我们先要向MongoDB中添加记录，于是 `message` 和 `message_ref` 两个集合就都创建出来了。

```
db.message.insert( {
    _id: ObjectId("600bea9ab5bafb311f147506"),
    uuid:"bfcb7c47-5886-c528-5127-ce285bc2322a",
    senderId: 0,
    senderPhoto: "https://static-1258386385.cos.ap-beijing.myqcloud.com/img/System.jpg",
    senderName: "Emos系统",
    msg: "HelloWorld",
    sendTime: ISODate("2021-01-23T17:21:30Z")
} );
```

```
db.message_ref.insert( {
    _id: ObjectId("600beaf0d6310000830036f3"),
    messageId: "600bea9ab5bafb311f147506",
    receiverId: 1,
    readFlag: false,
    lastFlag: true
} );
```

执行两个集合的联合查询，根据接收人来查询消息，并且按照消息发送时间降序排列，查询前50条记录

```
db.message.aggregate([
    {
        $set: {
            "id": { $toString: "$_id" }
        }
    },
    {
        $lookup:{
            from:"message_ref",
            localField:"id",
            foreignField:"messageId",
            as:"ref"
        },
    },
    { $match:{"ref.receiverId": 1} },
    { $sort: {sendTime : -1} },
    { $skip: 0 },
    { $limit: 50 }
])
```