订单获取数据库方式

以下是关于该代码中表格信息获取方式以及调用途径的完整描述：

表格信息获取方式

1. **通过接口请求获取数据**：

```
{
    "success_code": 200,
    "data": [
        {
            "order_id": "123456",
            "user_name": "张三",
            "name": "电影A",
            "cinema_name": "XX影院",
            "ticket_num": 2,
            "ticket_total_price": 50.0,
            "phone_code": "1234567890",
            "order_date": "2025-01-01 10:00:00"
        },
        // 更多订单数据对象...
    ],
    "total": 100
}
```

   - 代码中定义了 `loadCurrentPageOrder` 方法来负责从后端获取订单数据，并将获取到的数据填充到表格中进行展示。在这个方法内部，使用了 `await` 关键字以异步的方式调用 `getCurrentPageOrder` 函数（该函数是从 `../../../api/index` 文件中导入的，意味着它是与后端 API 进行交互的接口调用函数），向服务器发送请求并等待服务器返回包含订单数据的响应结果。
   - 传递给 `getCurrentPageOrder` 函数的参数包括当前页码（`currentPage`）、每页显示的数量（`pageSize`）以及搜索输入内容（`input`，实际使用的是 `searchInput` 变量的值，它在点击搜索按钮后会被赋值为用户输入的内容或者在分页切换等操作时保持之前的搜索内容）。

- 服务器接收到请求后，根据这些参数查询相应的订单数据，并将包含订单数据以及总记录数等信息的 JSON 格式数据返回给前端。例如，返回的数据结构可能类似如下形式（示例）：
- 前端接收到返回的 JSON 数据后，会进行判断，如果 `success_code` 的值等于 `200`，表示请求成功获取到了有效的订单数据，此时就会将返回数据中的 `data` 字段（即包含多个订单信息的数组）赋值给 `tableData` 变量，这个 `tableData` 变量正是绑定到 `el-table` 组件的 `:data` 属性上的，从而实现将后端获取到的订单数据展示在表格中。同时，也会将返回数据中的 `total` 字段的值赋值给 `total` 变量，用于分页组件显示总记录数以及计算总页数等功能。

2. **根据不同操作更新表格数据**：
   - **组件创建时加载初始数据**：在组件的 `created` 生命周期钩子函数中，会调用 `loadCurrentPageOrder` 方法，并传入初始的当前页码（设置为 `1`）、每页显示数量（设置为 `8`）以及空字符串（因为刚进入页面还没有进行搜索操作），来获取并加载第一页的订单数据作为初始展示内容填充到表格中。
   - **搜索操作更新数据**：当用户在页面的搜索框中输入内容并点击搜索按钮时，会触发 `search` 方法。在这个方法中，首先将用户输入框中输入的内容（通过 `v-model` 绑定在 `input` 变量上）赋值给 `searchInput` 变量，然后调用 `loadCurrentPageOrder` 方法，传入页码为 `1`（表示从第一页开始重新搜索查找符合条件的数据）、每页显示数量 `8` 以及新的搜索输入内容（`searchInput`），向后端发起新的请求，获取符合搜索条件的订单数据，进而更新表格展示的内容，实现根据用户搜索内容动态展示相应订单数据的功能。
   - **分页操作更新数据**：在分页组件中，当用户点击上一页、下一页或者直接选择某个页码进行切换时，会触发 `currentChange` 方法。该方法首先更新 `currentPage` 变量为当前切换后的页码值，然后调用 `loadCurrentPageOrder` 方法，传入更新后的当前页码、每页显示数量（固定为 `8`）以及之前保存的搜索输入内容（`searchInput`），重新向后端请求对应页面的订单数据，以此来更新表格展示的内容，确保表格能根据分页操作正确显示不同页面的订单信息。

调用途径

1. **函数调用关系**：
   - 整个获取表格信息的核心函数调用链路起始于组件的 `created` 钩子函数、`search` 方法以及 `currentChange` 方法，它们都会调用 `loadCurrentPageOrder` 方法来发起获取订单数据的请求。
   - `loadCurrentPageOrder` 方法内部再去调用 `getCurrentPageOrder` 函数，这个函数的实际调用是指向 `../../../api/index` 文件中定义的与后端交互的接口函数，通过传递合

适的参数（当前页码、每页显示数量、搜索内容）来向服务器请求获取对应的订单数据，从而实现了从后端获取数据并填充到表格的完整调用途径。

2. **模块导入依赖**：
    - 从代码结构来看，通过 `import {getCurrentPageOrder, deleteOrder} from '../../../api/index'` 语句，将 `getCurrentPageOrder` 函数引入到当前组件的脚本中，使得在组件内部能够调用这个函数与后端 API 进行通信，获取订单数据。这体现了代码在模块层面的依赖关系以及调用途径的构建，通过正确导入相应的 API 函数，才能在组件的方法中顺利发起请求获取表格所需的订单数据。

综上所述，表格信息通过向后端 API 发起请求的方式获取，依靠 `loadCurrentPageOrder` 方法调用 `getCurrentPageOrder` 函数这条调用途径，结合不同的操作场景（组件创建、搜索、分页）来动态更新表格展示的订单数据内容。