

iFlyCode 用户手册

目 录

1. iFlyCode 简介	1
2. iFlyCode 安装指南	1
2.1. 安装 JetBrains 系列 IDE	1
2.2. 安装 iFlyCode 插件	1
2.3. 注册讯飞开放平台账号	2
2.4. 账号登录	2
3. iFlyCode 使用指南	3
3.1. 快捷键和设置	3
3.2. 代码生成	5
3.3. 代码补齐	5
3.4. 代码解释	6
3.5. 代码纠错	6
3.6. 单元测试	7
3.7. 文档注释	7
3.8. 知识问答	7
3.9. 代码检查	7
3.10. 代码调试	8
3.11. SQL 生成/优化	9

1. iFlyCode 简介

iFlyCode 是一款智能编码助手插件，可以在程序员编程过程中沉浸式交互生成代码建议，助力程序员提升编码效率和企业敏捷开发。

iFlyCode 已适配多种主流编辑器，当前服务处于邀测阶段，欢迎您点击链接 <https://iflycode.xfyun.cn>，注册并申请 iFlyCode 试用。

2. iFlyCode 安装指南

2.1. 安装 JetBrains 系列 IDE

2.2. 安装 iFlyCode 插件

方法一：从插件市场直接安装

以 IDEA 为例：打开 IDEA，使用快捷键 `Ctrl + Alt + S` 或点击“File”-“Settings”-“Plugins”，搜索 iFlyCode 进行安装。

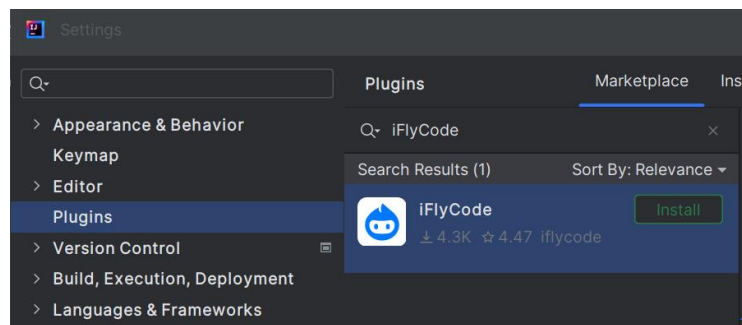


图 1 插件市场安装

方法二：从本地安装包安装插件

以 IDEA 为例：打开 IDEA，使用快捷键 `Ctrl + Alt + S` 或点击“File”-“Settings”-“Plugins”，点击右上角设置图标，选择“Install Plugin from Disk”。

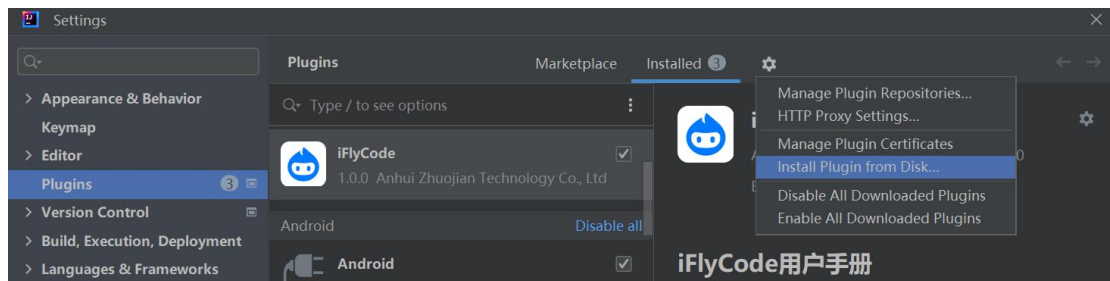


图 2 本地安装包安装

2.3. 注册讯飞开放平台账号

点击 <https://www.xfyun.cn/>，注册讯飞开放平台账号。

2.4. 账号登录

1) 点击插件登录。

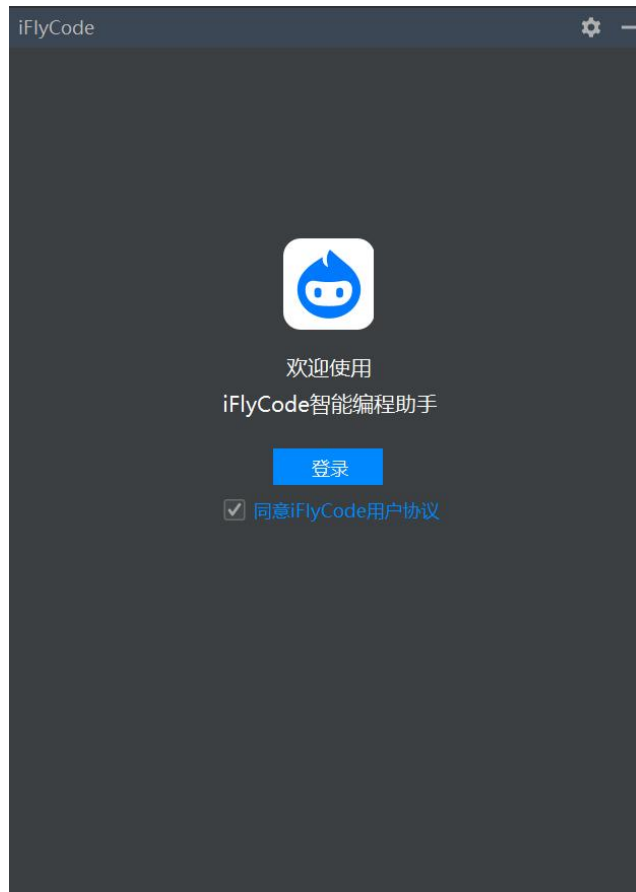


图 3 插件登录

2) 点击讯飞开放平台账号登录，若无账号请先进行注册。



图 4 点击讯飞开放平台账号登录

- 3) 登录成功，返回 iFlyCode 插件即可开启智能编程之旅。若无权限，请申请试用。

3. iFlyCode 使用指南

3.1. 快捷键和设置

日常使用中，iFlyCode 支持沉浸式生成/补齐代码，您只需进行正常编程工作，回车、空格等按键将自动生成代码建议，您可以使用 Tab 键采纳建议、Esc 拒绝建议或直接继续编程忽略建议。您可以选中代码后，通过右键或对话框上方选择代码解释/纠错/单元测试等功能、选中代码直接提问，或直接向 iFlyCode 提问技术问题。

iFlyCode 快捷键列表：

- Tab 采纳建议
- Esc 拒绝建议
- Alt+\ 主动触发建议

此外，iFlyCode 还提供多种自定义设置，

打开 iFlyCode 对话框，点击上方设置按钮，进入自定义设置页面。



图 5 iFlyCode 自定义设置按钮

iFlyCode 支持自定义设置，您可以根据使用习惯，自定义 iFlyCode 触发及交互：

- 自定义是否开启停顿触发及停顿触发的时间
- 自定义是否开启代码建议（若不开启则通过“Alt+\”主动触发代码建议）
- 自定义是否优先生成代码块（若不开启，则优先生成单行代码）
- 自定义聊天框发送消息按钮设置（Enter 发送问题或 Shift+Enter 发送问题；当设置 Shift+Enter 发送时，此时再点击 Enter 时就会在对话问题框内换行）

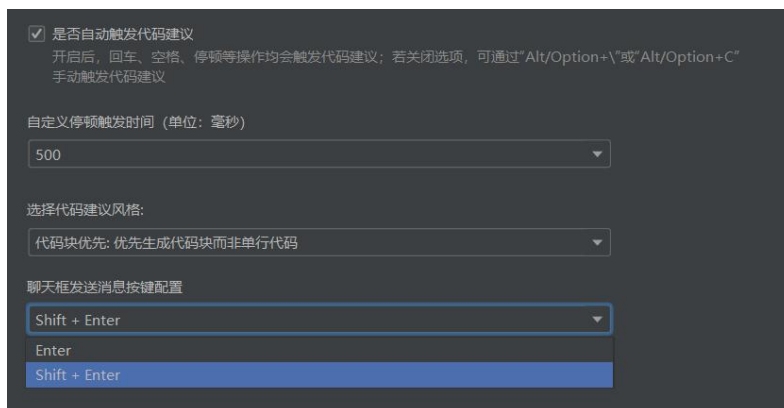
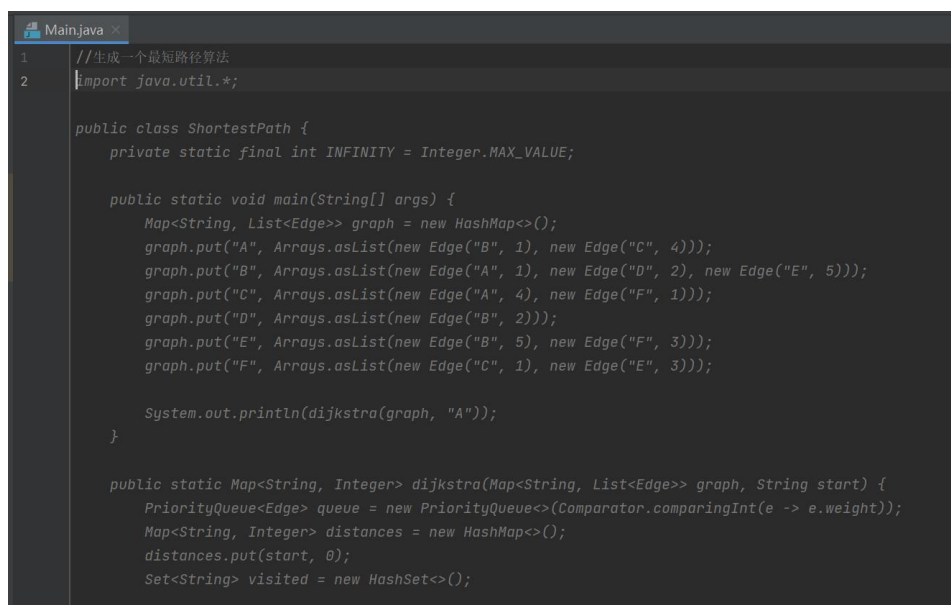


图 6 自定义设置页面

3.2. 代码生成

iFlyCode 支持在编辑器内根据注释、函数名生成代码，以注释生成代码为例，编写完成注释后，回车即触发代码建议，使用 Tab 键快捷采纳建议、Esc 拒绝建议或直接继续编程忽略建议。



```
1 //生成一个最短路径算法
2 import java.util.*;

public class ShortestPath {
    private static final int INFINITY = Integer.MAX_VALUE;

    public static void main(String[] args) {
        Map<String, List<Edge>> graph = new HashMap<>();
        graph.put("A", Arrays.asList(new Edge("B", 1), new Edge("C", 4)));
        graph.put("B", Arrays.asList(new Edge("A", 1), new Edge("D", 2), new Edge("E", 5)));
        graph.put("C", Arrays.asList(new Edge("A", 4), new Edge("F", 1)));
        graph.put("D", Arrays.asList(new Edge("B", 2)));
        graph.put("E", Arrays.asList(new Edge("B", 5), new Edge("F", 3)));
        graph.put("F", Arrays.asList(new Edge("C", 1), new Edge("E", 3)));

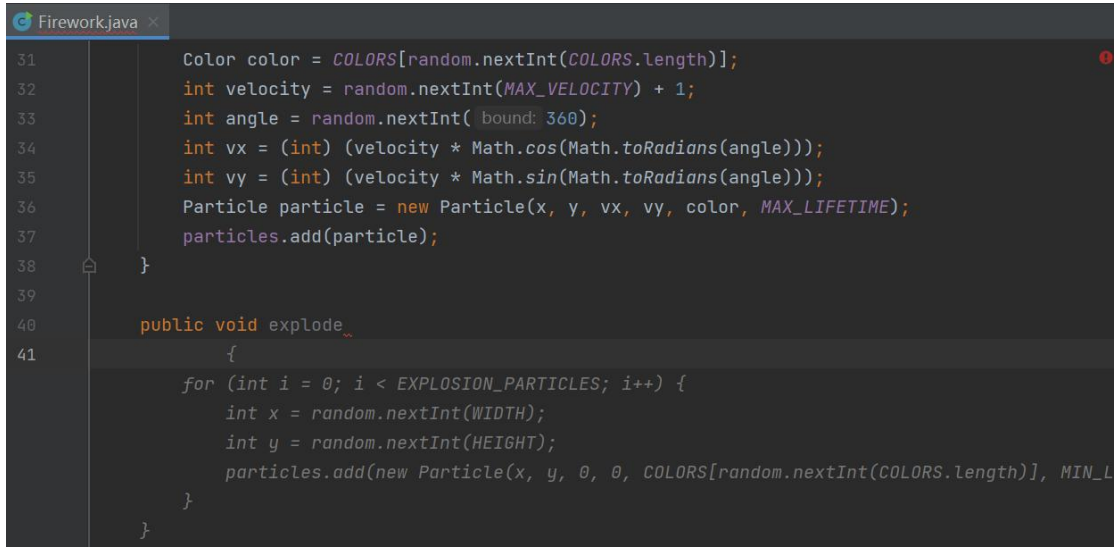
        System.out.println(dijkstra(graph, "A"));
    }

    public static Map<String, Integer> dijkstra(Map<String, List<Edge>> graph, String start) {
        PriorityQueue<Edge> queue = new PriorityQueue<>(Comparator.comparingInt(e -> e.weight));
        Map<String, Integer> distances = new HashMap<>();
        distances.put(start, 0);
        Set<String> visited = new HashSet<>();
    }
}
```

图 7 根据注释生成代码

3.3. 代码补齐

iFlyCode 支持在编辑器内，通过方法名、上下文等信息触发代码补齐，以上下文信息为例，在已有上下文的代码中，回车、空格均会自动触发代码建议，使用 Tab 键快捷采纳建议、Esc 拒绝建议或直接继续编程忽略建议。



```
31     Color color = COLORS[random.nextInt(COLORS.length)];
32     int velocity = random.nextInt(MAX_VELOCITY) + 1;
33     int angle = random.nextInt( bound: 360);
34     int vx = (int) (velocity * Math.cos(Math.toRadians(angle)));
35     int vy = (int) (velocity * Math.sin(Math.toRadians(angle)));
36     Particle particle = new Particle(x, y, vx, vy, color, MAX_LIFETIME);
37     particles.add(particle);
38 }
39
40 public void explode_
41 {
    for (int i = 0; i < EXPLOSION_PARTICLES; i++) {
        int x = random.nextInt(WIDTH);
        int y = random.nextInt(HEIGHT);
        particles.add(new Particle(x, y, 0, 0, COLORS[random.nextInt(COLORS.length)], MIN_L
    }
}
```

图 8 根据上下文生成代码

3.4. 代码解释

iFlyCode 支持对选中代码进行代码解释，包括该段代码的作用和含义、代码中的类和方法的意义等。选中代码后，右键选择“iFlyCode: 代码解释”使用代码解释功能。

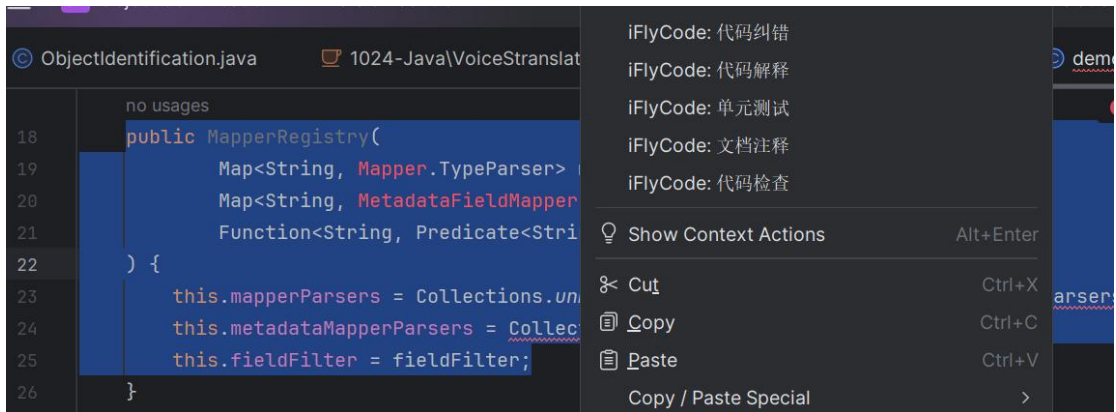


图 9 右键触发代码解释/代码纠错/单元测试/文档注释/代码检查

3.5. 代码纠错

iFlyCode 支持对选中代码进行代码纠错，包括该段代码中的拼写、语法和逻辑错误。选中代码后，右键选择“iFlyCode: 代码纠错”使用代码纠错功能（如图 9）。

3.6. 单元测试

iFlyCode 支持对选中代码生成单元测试用例。选中代码后，右键选择“iFlyCode: 单元测试”，使用单元测试功能（如图 9）。

3.7. 文档注释

iFlyCode 支持对选中的函数生成文档注释。选中代码后，右键选择“iFlyCode: 文档注释”使用文档注释功能（如图 9）。

3.8. 知识问答

iFlyCode 支持对选中代码进行提问或直接进行技术问题提问。

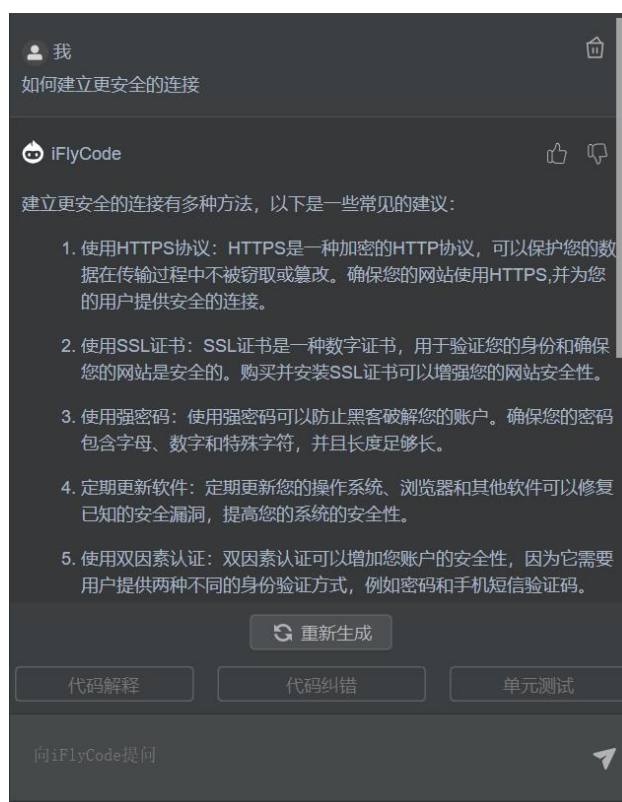


图 10 技术问答

3.9. 代码检查

右键选择“iFlyCode: 代码检查”，使用代码检查功能（如图 9）。检查后，iFlyCode 面板 Check 页面将给出检查结果。



图 11 代码检查结果

点击对应结果，定位到错误行。点击检查按钮进行检查修复。

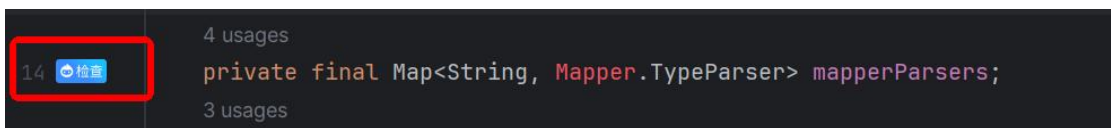


图 12 代码检查修复

3.10. 代码调试

代码运行报错，iFlyCode 提供代码调试功能，点击代码调试按钮即可对运行报错进行修复。

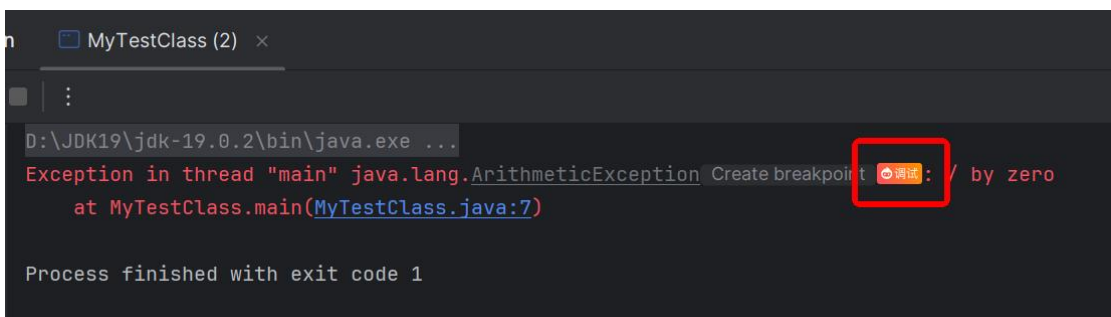


图 13 代码调试



图 14 代码调试修复

3.11. SQL 生成/优化

iFlyCode 支持配置 MySQL 数据源，结合数据库结构进行 SQL 生成和 SQL 优化。

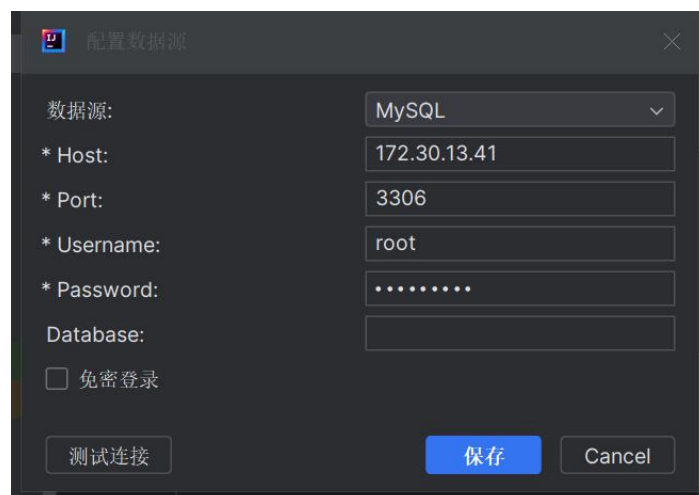


图 15 配置 MySQL 数据源

配置完成后，选择数据库和数据表，输入框处选择 SQL 生成或者 SQL 优化。

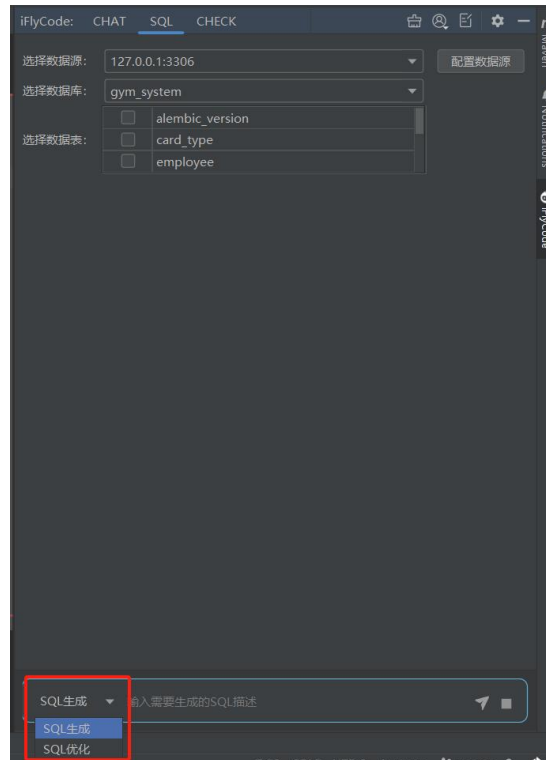


图 16 选择数据表，使用 SQL 生成和优化功能

对话式交互形式，自由提问 SQL 生成和优化需求。

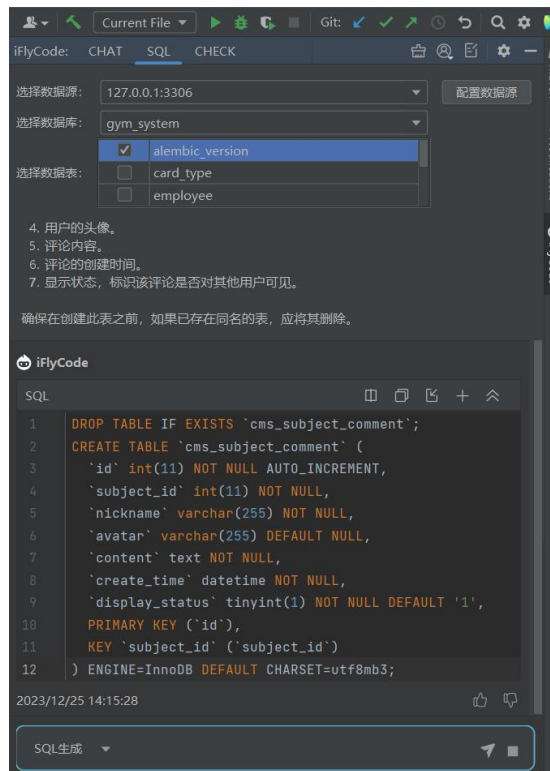


图 17 问答方式使用 SQL 生成

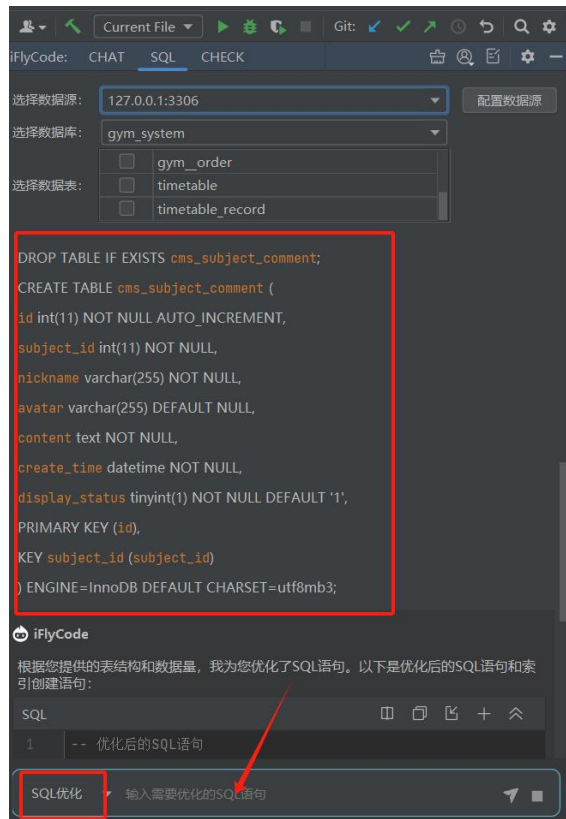


图 18 问答方式使用 SQL 优化