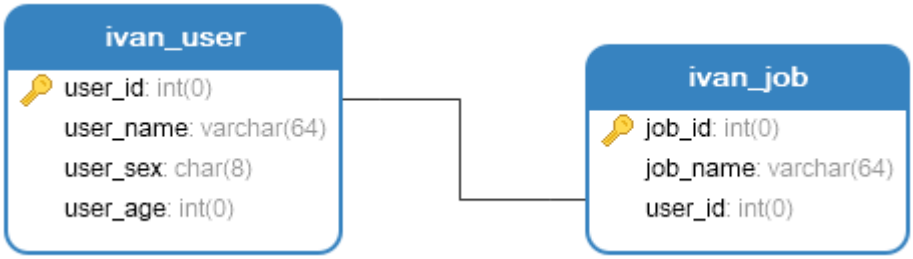
自定义多表关联分页查询

准备表和数据

```
-- ----------------------------
-- Table structure for ivan_user
-- ----------------------------
DROP TABLE IF EXISTS `ivan_user`;
CREATE TABLE `ivan_user`  (
  `user_id` int(0) NOT NULL AUTO_INCREMENT,
  `user_name` varchar(64) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL
DEFAULT NULL,
  `user_sex` char(8) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL
DEFAULT NULL,
  `user_age` int(0) NULL DEFAULT NULL,
  PRIMARY KEY (`user_id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 7 CHARACTER SET = utf8mb4 COLLATE =
utf8mb4_0900_ai_ci ROW_FORMAT = Dynamic;

-- ----------------------------
-- Records of ivan_user
-- ----------------------------
INSERT INTO `ivan_user` VALUES (1, 'Tom', '男', 18);
INSERT INTO `ivan_user` VALUES (2, 'Bob', '男', 22);
INSERT INTO `ivan_user` VALUES (3, 'Mary', '女', 17);
INSERT INTO `ivan_user` VALUES (4, 'Jim', '男', 18);
INSERT INTO `ivan_user` VALUES (5, 'Dave', '男', 22);
INSERT INTO `ivan_user` VALUES (6, 'Anly', '女', 17);

DROP TABLE IF EXISTS `ivan_job`;
CREATE TABLE `ivan_job`  (
  `job_id` int(0) NOT NULL AUTO_INCREMENT,
  `job_name` varchar(64) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL
DEFAULT NULL,
  `user_id` int(0) NULL DEFAULT NULL,
  PRIMARY KEY (`job_id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 7 CHARACTER SET = utf8mb4 COLLATE =
utf8mb4_0900_ai_ci ROW_FORMAT = Dynamic;

-- ----------------------------
-- Records of ivan_job
-- ----------------------------
INSERT INTO `ivan_job` VALUES (1, '教师', 1);
INSERT INTO `ivan_job` VALUES (2, '教师', 2);
INSERT INTO `ivan_job` VALUES (3, '教师', 3);
INSERT INTO `ivan_job` VALUES (4, '警察', 1);
INSERT INTO `ivan_job` VALUES (5, '警察', 2);
INSERT INTO `ivan_job` VALUES (6, '公务员', 1);
```

关系如下：



一对多，`1`个用户对应`n`个job

配置分页拦截器

```
@EnableTransactionManagement
@Configuration
public class MybatisPlusConfig {

    /**
     * 分页插件
     */
    @Bean
    public PaginationInterceptor paginationInterceptor() {
        return new PaginationInterceptor();
    }
}
```

准备业务逻辑:

1. Entity

```
@TableName(value ="ivan_user")
@Data
public class User implements Serializable {
    @TableId(type = IdType.AUTO)
    private Integer userId;

    private String userName;

    private String userSex;

    private Integer userAge;

    @TableField(exist = false)
    private static final long serialVersionUID = 1L;
}
```

```
@TableName(value ="ivan_job")
@Data
public class Job implements Serializable {
    @TableId(type = IdType.AUTO)
    private Integer jobId;

    private String jobName;

    private Integer userId;

    @TableField(exist = false)
    private static final long serialVersionUID = 1L;
}
```

```
@Data
public class UserVo {
    private Integer userId;

    private String userName;

    private String userSex;

    private Integer userAge;

    private List<Job> jobList;
}
```

2. Controller

```
@RestController
public class UserController {
    @Autowired
    private UserService userService;

    @GetMapping("users")
    public R<IPage<UserVo>> getAllUsers() {
        IPage<UserVo> userPage = userService.getUsersAndJobs();
        return R.ok(userPage);
    }
}
```

3. Service

```
public interface UserService extends IService<User> {
    IPage<UserVo> getUsersAndJobs();
}
```

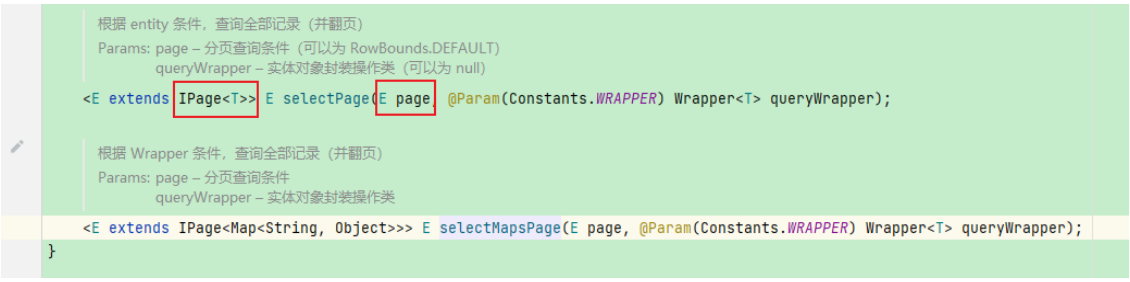
4. ServiceImpl

```
@Service
public class UserServiceImpl extends ServiceImpl<UserMapper, User>
implements UserService {

    @Autowired
```

```
    private UserMapper userMapper;

    @Override
    public IPage<UserVo> getUsersAndJobs() {
        QueryWrapper<UserVo> wrapper = new QueryWrapper<>();
        Page<UserVo> page = new Page<>(1,5);
        wrapper.in("user_id", 1,2,3,4,5,6);
        IPage<UserVo> userPage = userMapper.getUsersAndJobs(page, wrapper);
        return userPage;
    }
}
```

5. mapper

先观察一下MyBatisPlus现有的单表分页查询方法：



然后模仿写出自己的多表分页查询方法

```
@Repository
public interface UserMapper extends BaseMapper<User> {
    IPage<UserVo> getAllUsers(Page<UserVo> page, @Param(Constants.WRAPPER)
QueryWrapper<UserVo> wrapper);
}
```

> 注意：方法中第一个形参必须是Page，才能有分页效果；返回值可以是IPage，也可以是任意合理类型。

6. mapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
        PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.ivan.user.mapper.UserMapper">
    <resultMap id="UserVoResult" type="com.ivan.user.vo.UserVo">
        <id property="userId" column="user_id"/>
        <result property="userName" column="user_name"/>
        <result property="userAge" column="user_age"/>
        <result property="userSex" column="user_sex"/>
        <collection property="jobList" ofType="com.ivan.user.entity.Job">
            <id property="jobId" column="job_id"/>
            <result property="userId" column="user_id"/>
            <result property="jobName" column="job_name"/>
        </collection>
    </resultMap>
    <select id="getUsersAndJobs" resultMap="UserVoResult">
        SELECT *
        FROM (
                 SELECT u.user_id   AS user_id,
```

```
                        u.user_name AS user_name,
                        u.user_sex  AS user_sex,
                        u.user_age  AS user_age,
                        j.job_id    AS job_id,
                        j.job_name  AS job_name
                FROM `ivan_user` u
                         JOIN `ivan_job` j ON u.user_id = j.user_id
            ) x ${ew.customSqlSegment}
    </select>
</mapper>
```

测试

http://localhost:8080/users

控制台打印出来的SQL：

```
==>  Preparing: SELECT * FROM ( SELECT u.user_id AS user_id, u.user_name AS
user_name, u.user_sex AS user_sex, u.user_age AS user_age, j.job_id AS job_id,
j.job_name AS job_name FROM `ivan_user` u JOIN `ivan_job` j ON u.user_id =
j.user_id ) x WHERE (user_id IN (?,?,?,?,?,?)) LIMIT ?,?
==> Parameters: 1(Integer), 2(Integer), 3(Integer), 4(Integer), 5(Integer),
6(Integer), 0(Long), 5(Long)
<==    Columns: user_id, user_name, user_sex, user_age, job_id, job_name
<==        Row: 1, Tomm, 男, 18, 1, 教师
<==        Row: 2, Bob, 男, 22, 2, 教师
<==        Row: 3, Mary, 女, 17, 3, 教师
<==        Row: 1, Tomm, 男, 18, 4, 警察
<==        Row: 2, Bob, 男, 22, 5, 警察
<==      Total: 5
```

debug后观察，已经查询到了多表分页后的数据：