搜索中心依赖es，是对es客户端的一层封装，独立于业务应用，对外提供接口，优点：

- 屏蔽复杂的配置，让开发使用更简单，也更利于维护
- 和应用系统解耦，以后es升级只需要动搜索中心
- 让业务应用的jar包体积更小
- 扩展性更好，以后可以提供给前端

设计与实现

搜索中心需要包含两部分基本功能，即**搜索api**和**增量同步数据**。
crm系统的搜索场景基本上都是一个搜索框可以同时对表中一个或多个字段进行搜索，但不同字段的搜索方式有区别，有的需要支持拼音模糊搜索（如产品名称），有的需要全词匹配搜索（如地区编码），有的只需要模糊搜索不需要拼音（如产品编码）；
而增量同步功能包含根据配置创建定时任务，增量的扫描db数据同步到es，包括同步删除的处理。
搜索中心的两大功能基于表配置
（ `ELASTICSEARCH_CFG` ）来实现，如下为表结构
及其字段说明：

表字段	说明
SEARCH_CODE	搜索编码（**全小写**形式，如：td_b_product_group）
SEARCH_NAME	搜索名（描述）
SEARCH_FILED	搜索字段配置，"字段名/搜索类型"
DATA_SOURCE	数据源（执行sql对应的数据源，如：upc）
QUERY_SQL	数据查询SQL（ES中的数据来源）
DOCUMENT_ID	数据导入ES后，作为ES文档id的字段，如、${OFFER_ID}+${OFFER_SUB_ID}（连表查询一对多时要注意是多个表的主键连接才能确定唯一性）
SYNC_CRON	同步数据到ES的频率（cron表达式），示例："* */30 * * * ?"（每30分钟同步一次）
AUTHOR	作者
STATE	状态(U：开启)，如果应用启动后改为E，则会暂停同步任务

1 / 搜索功能相关配置

SEARCH_CODE（搜索编码）

此字段是主键，须保持唯一性，且需要全小写形式，因为该值同时会充当es中的索引名（索引名要求必须全小写），一般建议使用表名如：td*b*product_group

SEARCH_NAME（搜索名）

只是一个描述，无特别作用

SEARCH_FILED（搜索字段）

需要被搜索或作为条件进行过滤的字段都需要配置在该字段中，如：
`FIELD_A/pinyin`
框架设计了三种字段搜索的类型（其实就是ES中字段的分词器类型）：
示例：假设产品搜索场景，需要对产品编码 `OFFER_CODE` 进行模糊搜索，对产品名称 `OFFER_NAME` 进行拼音搜索，同时还有个下拉框可以选择地区 `EPARCHY_CODE` （即精确匹配），那么配置如下：

```
-- 多条搜索字段配置示例:
OFFER_CODE/standard,
OFFER_NAME/pinyin,
EPARCHY_CODE/keyword
```

2 / 增量同步相关配置

DATA_SOURCE（数据源）

执行sql对应的数据源

QUERY_SQL（数据查询sql）

该sql需要注意，必须能查询出每次查询之后发生变化的”新数据”，所以要求表中必须有如 `DONE_DATE` 这种**操作时间含义**的字段，并保证每次新增、修改数据时都会更新此字段，同时搜索中心在sql中提供了两个占位符 `:last_sync_time` 和 `:now` ，分别代表上次同步的时间和当前时间，如此一来，通过操作时间及这两个时间占位符，进行比较就能获取新数据（新增或修改）：

```
select
ID,
CODE,
...
from
...
where
...
and DONE_DATE >= :last_sync_time
and DONE_DATE < :now
-- 注：DONE_DATE < :now 这个条件很关键，可以保证数据不重复导入也不会丢失，记得加上
```

删除数据的同步

上面示例可以同步新增或修改的数据，但在删除场景下，由于数据库中记录都不存在了，所以没法进行同步，解决方案如下：
将业务上的物理删除改成**逻辑删除**，在数据表中增加一个“逻辑删除”标记字段`IS_DELETED`（建议类型为`char(1)`）（使用MybatisPlus的逻辑删除功能简单配置后即可实现删除时将`IS_DELETED`更新成`1`，配置方式请参见官网MybatisPlus逻辑删除），然后在配置该字段时将`IS_DELETED`字段写在select语句中，搜索中心会在执行sql后会根据IS_DELETED是否等于`1`来执行es删除操作。

```
-- 删除数据的同步，查询结果中含有IS_DELETED字段
select
ID,
CODE,
IS_DELETED,
...
from
...
where
...
and DONE_DATE >= :last_sync_time
and DONE_DATE < :now
```

如果表数据改成逻辑删除后数据积压导致数据库压力增大，可以使用定时任务**延迟删除**`IS_DELETED`为1的记录（一定要在同步到es之后才删除）。

DOCUMENT_ID

es中的文档ID生成规则，使用占位符来指定，如`${ID}`、`${CODE}`

`+${TYPE}`，占位符的字段由QUERY_SQL执行结果中取，需要确保生成的id唯一。

SYNC_CRON

数据同步频率，CRON表达式，一共6位，以空格分隔，分别代表“秒分时天月年”，如每30分钟同步一次：

```
* */30 * * * ?
```

3 / API

搜索中心暴漏的API都写在 `SearchController` 类中，
默认context-path是：/bits-搜索中心

```
- 搜索相关
搜索：/search
计数：/count

- 其他方法
- 创建或刷新索引：
/createIndex/{indexName}
/refreshIndex/{indexName}
- 刷新数据（重建索引、全量同步数据）：
/refreshData/{indexName}
- 增量添加数据：
/addData/{indexName}
- 获取所有配置信息：
/getConf
- 获取指定配置信息：
/getConf/{indexName}
```

工程结构

核心类

EsUtils

负责es相关的功能，如搜索，创建删除索引，同步数据等，类结构如下：

SyncDbToEs

负责根据配置中的cron表达式创建定时同步任务：

SearchClient——搜索客户端

SearchClient类不在搜索中心的包中，它需要给业务侧使用，因此写api包中，一共3个搜索方法外加1个计数方法，如下：

由于搜索中心提供的是RESTful API，前端也可以实现js版的SearchClient