

DB -> Kafka -> ElasticSearch (Kafka, ES)

DB -> Kafka -> ElasticSearch Kafka + Bulk

DB -> Kafka -> ElasticSearch

1. **DB -> Kafka -> ElasticSearch** DB -> Kafka -> ElasticSearch
2. **Kafka -> bucket** Kafka -> bucket
3. **ElasticSearch -> Bulk API** ElasticSearch -> Bulk API

DB -> Kafka -> ElasticSearch

1. **Kafka -> Bucket** Kafka -> Bucket
2. **ElasticSearch -> Bulk** ElasticSearch -> Bulk
3. **DB -> Kafka -> ElasticSearch** DB -> Kafka -> ElasticSearch

Kafka Bucket

```
qryCnt := models.QryCount(y.Db, cntStr)
fmt.Println("qryCnt:", qryCnt)

bucketCnt := 0
if qryCnt == 0 {
    bucketCnt = 0
} else {
    bucketCnt = 1 + (int(qryCnt) / p.Limit)
}

fmt.Println("bucketCnt:", bucketCnt)
for i := 0; i < bucketCnt; i++ {
    // for i := 0; i < 3; i++ {
    p.Offset = p.Limit * i
    sqlStr, _, _ := models_join.ListType1PlainQryStrGet(y, lt, p, q, "/list")
```

```

    if err := t.ProduceaBucketToKafka(y, "kkk", headers, sqlStr, topic); err != nil {
        return errors.New(e.PageQryErr("vrt452", ": "+err.Error()))
    }
}

```

Bucket

ElasticSearch Bulk API

```

for _, row := range v.Page {
    row.ClientCode = clientCode

    fmt.Printf("BuyerId: %d, LastSorderDate: %s, ClientCode : %s\n", row.Id, row.LastSorderDate, row.ClientCode)

    row.LastVistDate = row.LastSorderDate

    // Meta
    meta := fmt.Sprintf(` { "index": { "_index": "%s", "_id": "%d" } }%s`, esIndex, row.Id, "\n")

    bulkRequest.WriteString(meta)

    //
    data, _ := json.Marshal(row)

    bulkRequest.WriteString(string(data) + "\n")

    fmt.Println(string(data)) //

    fmt.Printf("Prepared document for BuyerId: %d\n", row.Id)
}

// Bulk
res, err := es.Bulk(bytes.NewReader(bulkRequest.Bytes()))

if err != nil {
    return fmt.Errorf("bulk request failed: %w", err)
}

defer res.Body.Close()

// Elasticsearch
if res.IsError() {
    log.Printf("Bulk indexing failed: %s", res.String())

    return fmt.Errorf("bulk indexing failed")
}

```

Bulk API

Revision #1

Created 7 February 2025 03:02:29 by ☐

Updated 7 February 2025 03:06:08 by ☐