Разработка средства генерации SQL-запроса для упрощения задач по тестированию

Андрей Околелов

РСХБ-Интех

HEISENBUG

РСХБ цифра

РоссельхозБанк

Что такое тест?

Найти договор

На основе найденного создать новый

Договор создан

Критерии

Договор

- Дата открытия договора – в 2020 году ЦБ выпустил новое указание №10005
- Тип договора – простой, сложный, красный
- Вид договора – в национальной валюте, в иностранной валюте
- Валюта договора – рубль, юань, доллар, евро
- Дата закрытия

Критерии

Клиент

- Тип клиента – физическое лицо, юридическое лицо, ИП
- Группа клиента – вип, не вип, средний
- Справочник по террористам – включен, не включен
- Справочник особого внимания – включен, не включен
- Действующий – да, нет

Критерии

Счет договора

- Счет 2 порядка
- Валюта – RUB, USD
- Остаток на счете – открыт, закрыт, зарезервирован
- Подразделение счета
- Статус – открыт, закрыт, зарезервирован

```
select dep.*
from IBS.Z#R2_DEPOSIT dep
join IBS.Z#CL_PRIV cl on cl.id = dep.C_CLIENT
join IBS.Z#CLIENT cl_all on cl.id = cl_all.id
join IBS.Z#FT_MONEY cur on dep.C_CURRENCY = cur.ID
join IBS.Z#COM_STATUS_PRD status on dep.C_COM_STATUS = status.id
join IBS.Z#DEPART filial on dep.c_depart = filial.id
join IBS.Z#R2_VID_DEBT accountType on accountType.c_code = 'SUM_DOG'
join IBS.Z#R2_CRED_DEBT balance on balance.C_PRODUCT = dep.ID and balance.C_PN = dep.C_PN and balance.C_VID_DEBT = accountType.id
join IBS.Z#R2_VID_DEPOSIT type1 on dep.C_VID_DEPOSIT=type1.id
where
--паспорт не просрочен
(cl.C_DOC#DATE_END is null or cl.C_DOC#DATE_END > :curr_date)
-- нет исполнительных производств
and not exists (select * from ibs.z#EXECUTORY_PROCES where C_CLIENT = cl.id)
-- не банкрот
and not exists (select * from IBS.Z#CL_CATEGORIES where collection_id = cl_all.c_vids_cl)
-- код подразделения заполнен
and cl.C_DOC#DEPART_CODE is not null
-- из определенного филиала
and cl_all.C_FILIAL = (select id from ibs.Z#BRANCH where C_CODE = :branch)
?condition
-- нет кросс-ссылок
--and (select count(1) from ibs.Z#SHB_XREF_FILIALS tc where tc.collection_id=cl_all.C_SHB_XREF_FILIALS and rownum=1) = 0
FETCH FIRST 1 ROW ONLY
```

Проблемы:

- Монолит, с которым тяжело разбираться
- Монолит перемещается от теста к тесту. При перемещении разработчик может добавить что-то ненужное, или упустить очень важное
- Проблема дублей
- Нет единства в запросах
- Выдача запроса: выдает в основном единичные элементарные данные (ID, CollectionId и т.д.)
- Неудобства при переносе запроса из проекта в SQL develop для тестирования и обратно, принимаем как должное
- Для написания запроса требуется уверенное знание структуры БД

ЦЕЛЬ:

Разработать инструмент получения данных на основе элементов предметной области

Требования:

- FrameWork для работы с текущей БД и на перспективу
- Понятная, удобная выдача, с необходимым набором данных
- Дружественный инструмент для разработчика и тестировщика
- Удобный инструмент разработчика для формирования SQL запросов на основе модели данных
- Удобный инструмент разработчика для формирования SQL запросов на основе элементов предметной области

FrameWork

```java
/**
 * Получить номер счета документа картотеки по id договора РКО
 */
1 usage  Hairov-AF +1 *
public class DocumCardIndex2 extends DbRequest {
    2 usages
    private final String rkoId;

    1 usage  Hairov-AF
    public DocumCardIndex2(String rkoId) { this.rkoId = rkoId; }

    Hairov-AF +1 *
    @Override
    /unchecked/
    protected <Z> Z handling(ResultSet resultSet) throws SQLException {
        String result = "";

        while (resultSet.next()) {
            result = resultSet.getString( columnLabel: "C_MAIN_V_ID");
        }
        log.info("Card-index document account number (like 90902810%): {}", result);
        return (Z) result;
    }

    Мустафин Искандар Ильгамович +1 *
    @Override
    public <Z> Z getResult() throws SQLException, IOException {
        String query =  String.format(
                "SELECT af.C_MAIN_V_ID FROM Z#CARD_INDEX card " +
                        "JOIN Z#AC_FIN af ON card.C_ACC_REF = af.ID " +
                            "WHERE card.COLLECTION_ID = (SELECT C_CARD_IND FROM Z#RKO " +
                                "WHERE ID = '%s')",
                rkoId
        );
        log.info(query);
        return dbWork(query);
    }
}
```

FrameWork

```java
/**
 * Возвращает список Dto_HozOpAcc (Счёта договора)
 *
 * @param builderSql - строитель по параметрам {@link ParametersBuilderHozOpAcc}
 * @param maxResult  - Максимальный размер возвращаемого списка
 * @return List<Dto_HozOpAcc>
 */
1 usage  Родько Роман Сергеевич
private List<Dto_HozOpAcc> getHozOpAccListByParameters(CommonBuilderSql builderSql, int maxResult) {
    return instanceDao.getListEntityByParameters(builderSql, new Dto_HozOpAcc(), maxResult);
}
```

FrameWork

```java
@Data
@Entity
@Table(name = "z#HOZ_OP_ACC", schema="ibs")
public class Dto_HozOpAcc {
    1 usage
    @Id
    private long id;
    no usages
    @Column(name = "collection_id")
    private String collectionId;
    1 usage
    @OneToOne(fetch = FetchType.EAGER)
    @JoinColumn(name = "c_name_account")
    private Dto_TypeAccount linkTypeAccount;
    1 usage
    @OneToOne(fetch = FetchType.EAGER)
    @JoinColumn(name = "c_account_dog#1#2")
    private Dto_AcFin linkAccount;

    okolelov-apa +1 *
    @Override
    public String toString() {
        String tableNameToUpperCase = this.getClass() Class<capture of ? extends Dto_HozOpAcc>
                .getAnnotation(Table.class) Table
                .name() String
                .split( regex: "#")[1].toUpperCase();
        return String.format("%s(id=%d, type:=%s, account:=%s",
                tableNameToUpperCase,
                id,
                linkTypeAccount.getCode(),
                linkAccount.getAccountNumber()
                );
    }
}
```

Дружественность инструмента

Какие вопросы были проработаны

- Особое внимание уделили параметрам для запросов
- Формирование Javadoc
- Названия шагов теста и результат шага
- Название методов и классов

Дружественность инструмента

Название атрибутов класса Dto*

```
@Column(name = "c_name")
private String name;

@Column(name = "c_registr_num")
private Long registrationNumber;

@OneToOne(fetch = FetchType.EAGER)
@JoinColumn(name = "c_country")
private Dto_Country linkCountry;
```

Удобный инструмент для формирования SQL запросов на основе модели данных

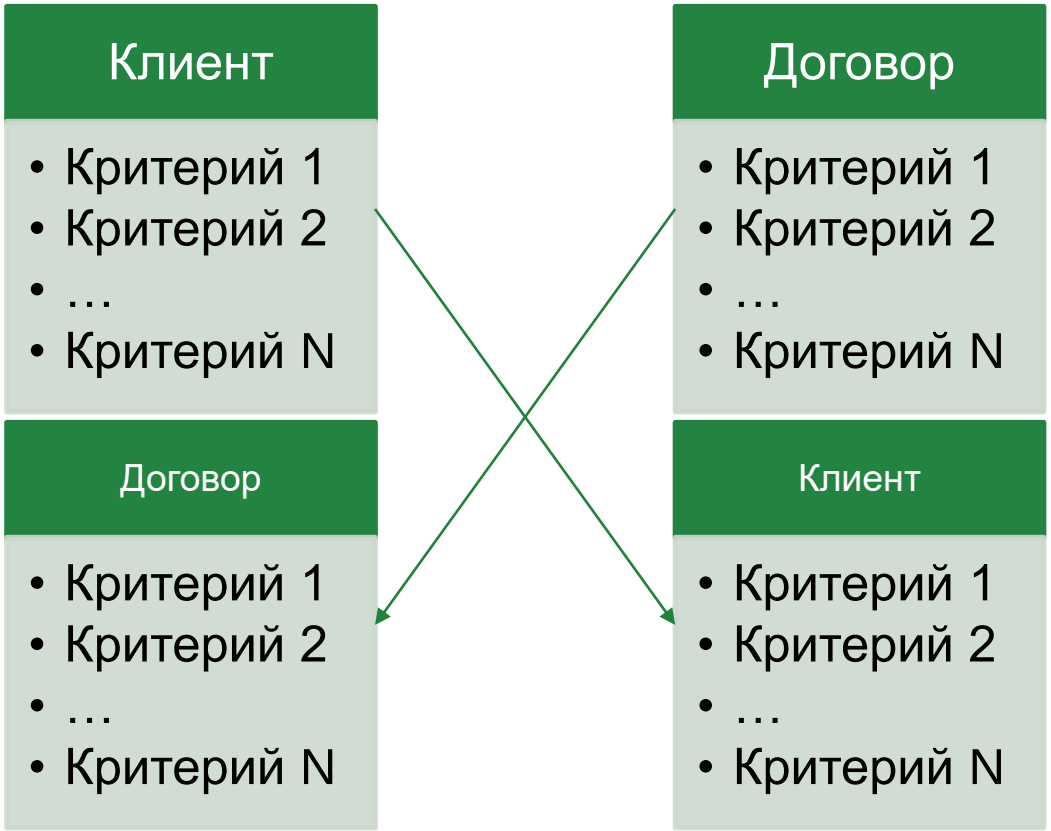
```
String query = String.format("SELECT * FROM Z#DOC_CARD_INDEX WHERE C_VNB_DOC = %s", transferId);

String query =  String.format(
        "SELECT * FROM Z#MAIN_DOCUM partition(Z#MAIN_DOCUM#0) main " +
                "JOIN STATES s ON main.STATE_ID = s.ID AND main.CLASS_ID = s.CLASS_ID " +
                    // Счет по дебету
                    "WHERE main.C_NUM_DT = '%s' " +
                        // Счет по кредиту
                        "AND main.C_NUM_KT = '%s' " +
                        // текущий опер день
                        "AND main.C_DATE_DOC = '%s'",
        dt, kt, opDayDate
);
```

Удобный инструмент для формирования SQL запросов на основе модели данных

```
queryBuilder = new QueryBuilder()
        .select(ColumnCollection.ALL)
        .from(tableRkoLimitPay)
        .join(innerJoin(tableRkoLimits, equal(tableRkoLimitPay.C_KIND_LIMIT, tableRkoLimits.ID)))
        .where(
                equal(tableRkoLimitPay.COLLECTION_ID, targetTable.C_LIMITS_PAY),
                isNull(tableRkoLimitPay.C_DATE_END),
                criteria
        );
```

```
SELECT rko_limit_pay.*
  FROM ibs.z#rko_limit_pay rko_limit_pay
       INNER JOIN ibs.z#rko_limits rko_limits
       ON rko_limit_pay.c_kind_limit = rko_limits.id

WHERE rko_limit_pay.collection_id = rko.c_limits_pay
  AND rko_limit_pay.c_date_end IS NULL
  AND rko_limit_pay.c_date_beg <= to_date('2023-10-09', 'yyyy-mm-dd')
```

Удобный инструмент для формирования SQL запросов на основе элементов предметной области



Удобный инструмент для формирования SQL запросов на основе элементов предметной области

ТК №1000

Клиент ЮЛ — резидент, финансовая организация (признак счета-Финансовые/F_Ф)

Справочники→ Признаки счетов→ Форма предприятия (для ОКОПФ)→ Финансовые → Cntrl+R→ Общероссийский классификатор организационно-правовых форм (ОКОПФ)→ РСХБ. Актуальные коды (просмотр актуальных ОКОПФ для коммерческих организаций)

Расчетно-кассовое обслуживание→ Полный список→ ALT+3:

- Расчетный счет = 40501
- Статус договора = На открытие
- Текущий остаток - больше 0
- Филиал = 000,001,002
- Валюта = 810
- Дата открытия - текущий день

Применить→ скопировать Рег. №. Выбрать договор

Найти Договор, где:

- Клиент ЮЛ:
  - резидент,
  - финансовая организация
- Счет:
  - Счет второго порядка = 40501
  - Текущий остаток — больше 0
- Договор
  - Филиал = 000,001,002
  - Валюта = 810
  - Дата открытия - текущий день
  - Статус договора = На открытие

Удобный инструмент для формирования SQL запросов на основе элементов предметной области

Найти Договор, где:

- Клиент ЮЛ:
  - резидент,
  - финансовая организация
- Счет:
  - Счет второго порядка = 40501
  - Текущий остаток — больше 0
- Договор
  - Филиал = 000,001,002
  - Валюта = 810
  - Дата открытия - текущий день
  - Статус договора = На открытие

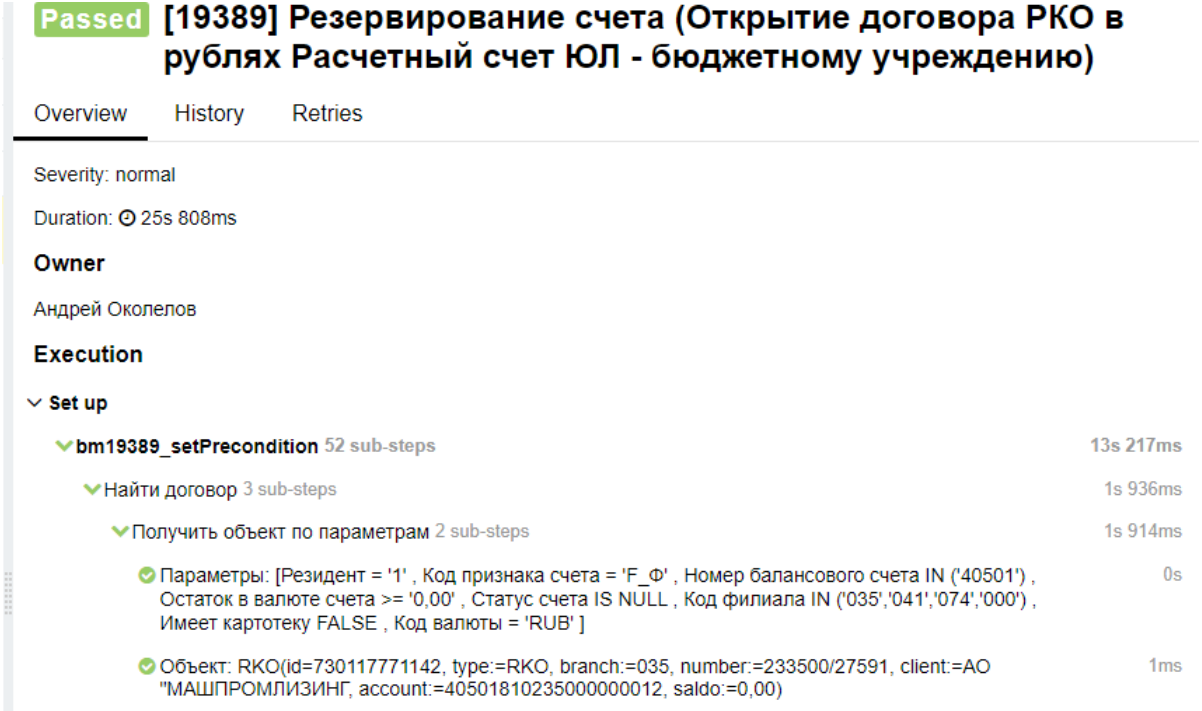
```
ParametersBuilderRko contractParameters = new ParametersBuilderRko()
        .joinClCorp(new ParametersBuilderClCorp()
                .withResident(IsResident.RESIDENT)
                .withPs(Ps.F_K))
        .joinAccount(new ParametersBuilderAccount()
                .withBalanceAccountInTo(balanceAccount)
                .withSaldoGreaterThanOrEqual( balance: 0))
        .withBranches(branches)
        .withCurrency(Currency.RUB)
        .withOpenedDateIsEqualTo(LocalDate.now())
        .withAccountStatus(ComStatusPrd.$TO_OPEN);
```

Удобный инструмент для формирования SQL запросов на основе элементов предметной области

```
tableRkoLimitPay = new TableRkoLimitPay(aliasSuffix);
queryBuilder = new QueryBuilder()
        .select(ColumnCollection.ALL)
        .from(tableRkoLimitPay);

public ParametersBuilderRkoLimitPay withMaxSequenceOfPaymentsEqualTo(int priority) {

    queryBuilder.where(equal(tableRkoLimitPay.C_PRIORITET, priority));

public ParametersBuilderRkoLimitPay withSourceOfOriginEqualTo(RkoLimitOrigin limitOrigin) {

    ParametersBuilderRkoLimitOrigin rkoLimitOrigin = new ParametersBuilderRkoLimitOrigin(aliasSuffix)
            .withCodeEqualTo(limitOrigin);

    queryBuilder
            .join(innerJoin(
                    rkoLimitOrigin.getTableRkoLimitOrigin(),
                    equal(tableRkoLimitPay.C_LIMIT_ORIGIN, rkoLimitOrigin.getTableRkoLimitOrigin().ID)))
            .where(rkoLimitOrigin.getCriteria());
```

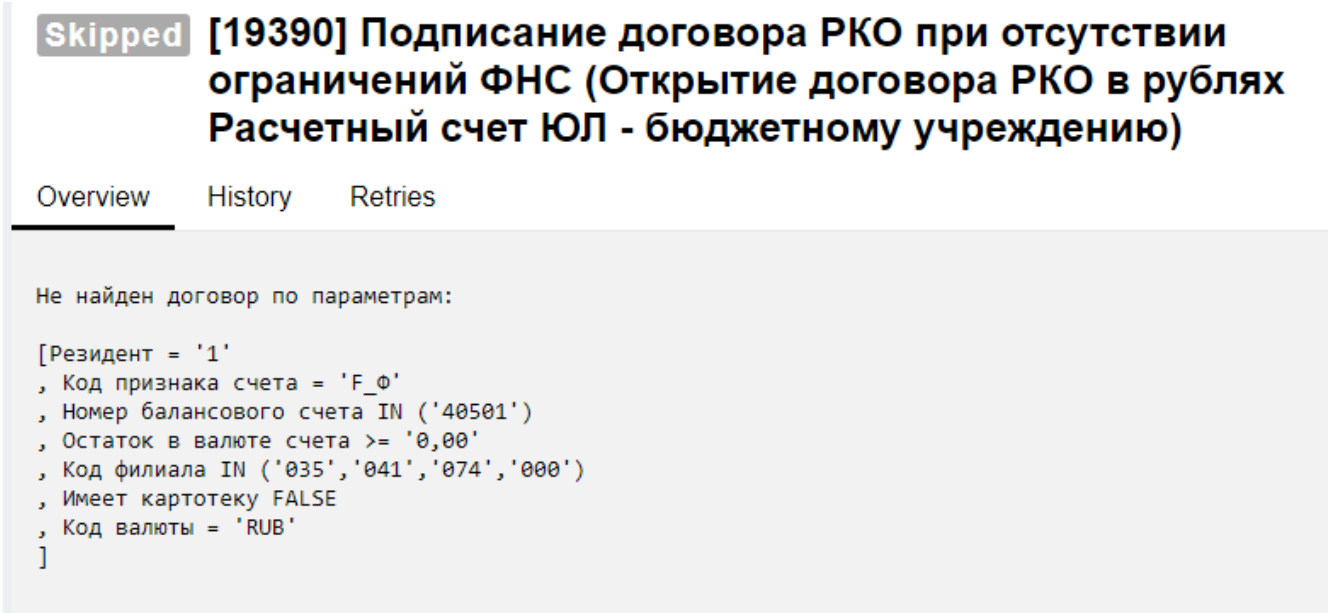
Что мы еще получили:

Приятные бонусы в Allure



Что мы еще получили:

И не очень приятные



Какой эффект получен

- Разработан удобный инструмент поиска данных для разработчика
- Инструмент удобен для команды
- Понимаем, как будем развивать и актуализировать
- Начала решаться проблема дубликатов
- Быстрый вход в проект

НА 20% БОЛЬШЕ ТЕСТОВ*

*мы просто наняли еще одного тестировщика в команду

Проблемы при эксплуатации

Случай 1

Продукт	
Филиал	000

```
queryBuilder = new QueryBuilder()
        .select(ColumnCollection.ALL)
        .from(tableProduct)
        .join(innerJoin(tableBranch, equal(tableProduct.C_FILIAL, tableBranch.ID)))
        .where(equal(tableBranch.C_CODE, branch));
```

Ошибка! Счет принадлежит другому подразделению

Случай 2

Продукт	
Филиал	000
Подразделение	001-001

```
queryBuilder = new QueryBuilder()
        .select(ColumnCollection.ALL)
        .from(tableProduct)
        .join(innerJoin(tableBranch, equal(tableProduct.C_FILIAL, tableBranch.ID)))
        .join(innerJoin(tableDepart, equal(tableProduct.C_DEPART, tableDepart.ID)))
        .where(
                equal(tableBranch.C_CODE, branch),
                equal(tableBranch.C_CODE, substr(tableDepart.C_CODE, startPosition: 1, length: 3))
        );
```

ПОТЕНЦИАЛЬНЫЕ ВОЗМОЖНОСТИ

- Задача 1: Импортозамещение
- Задача 2: Простой доступ к данным заинтересованных лиц