Science Journal Progect

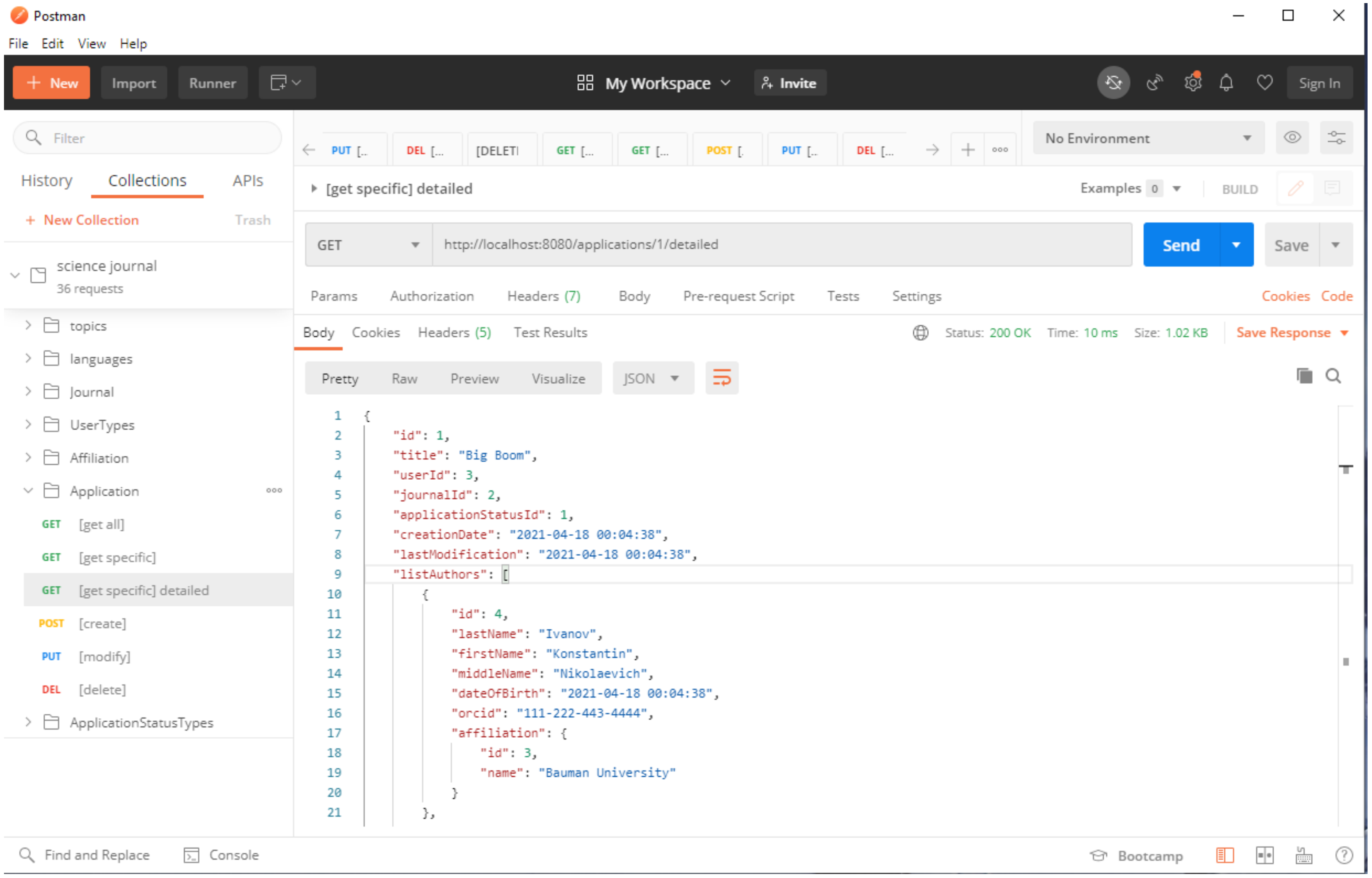
Kuznetsov Alexander

GitHub: https://github.com/AlexanderKuznetsov13/sciencejournal

Used technologies

- Java 11
- Apache Maven (project build)
- Spring Boot 2 (for MVC and dependency injection)
- Spring AOP (use aspects for logging)
- PostgreSQL 13 (data storage)
- Liquibase (database migration & deployment)
- Postman (for testing of REST endpoints)

Postman (for testing of REST endpoints)

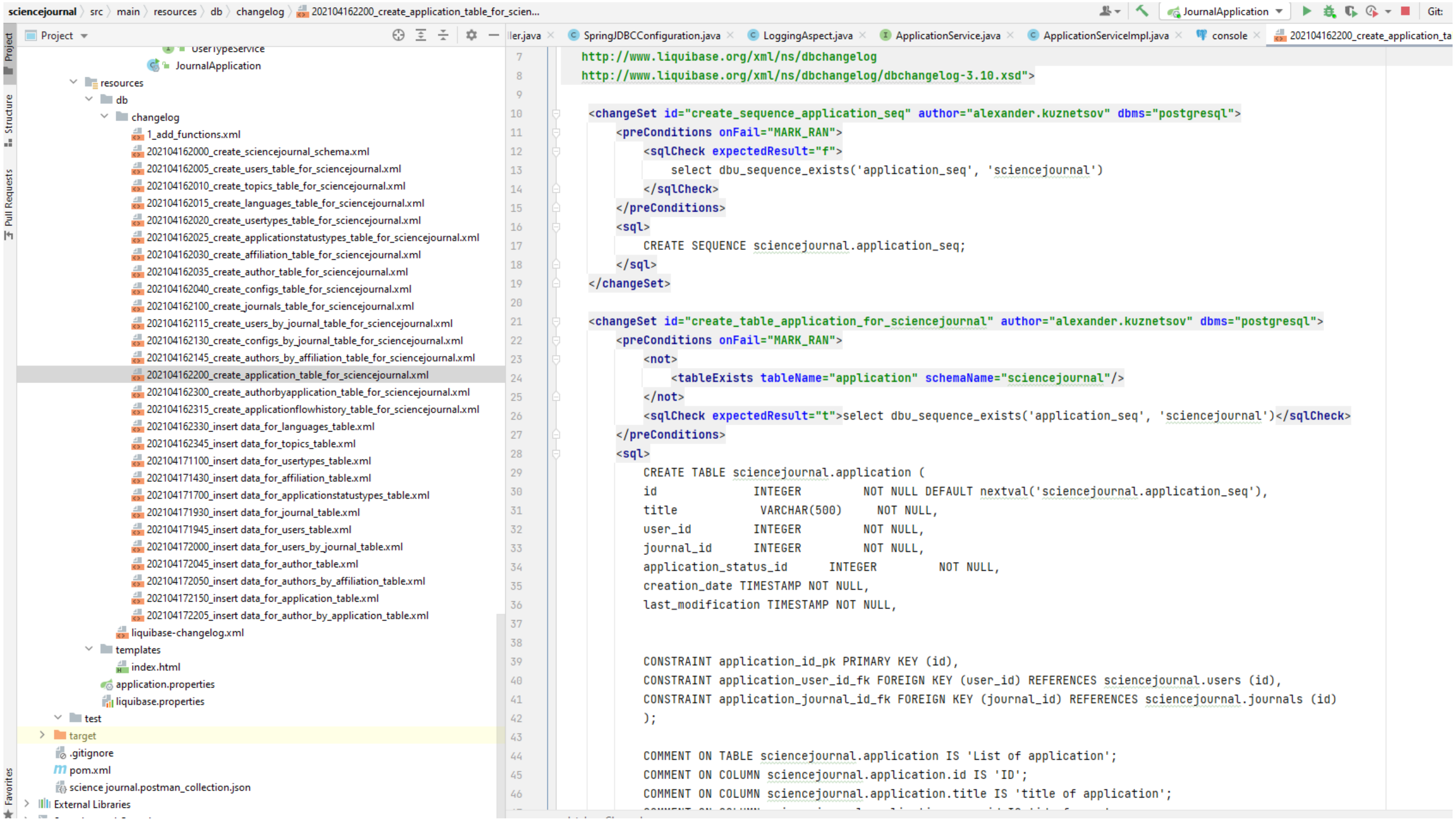


GET http://localhost:8080/applications/1/detailed

Status: 200 OK Time: 10 ms Size: 1.02 KB

```
{
    "id": 1,
    "title": "Big Boom",
    "userId": 3,
    "journalId": 2,
    "applicationStatusId": 1,
    "creationDate": "2021-04-18 00:04:38",
    "lastModification": "2021-04-18 00:04:38",
    "listAuthors": [
        {
            "id": 4,
            "lastName": "Ivanov",
            "firstName": "Konstantin",
            "middleName": "Nikolaevich",
            "dateOfBirth": "2021-04-18 00:04:38",
            "orcid": "111-222-443-4444",
            "affiliation": {
                "id": 3,
                "name": "Bauman University"
            }
        },
```

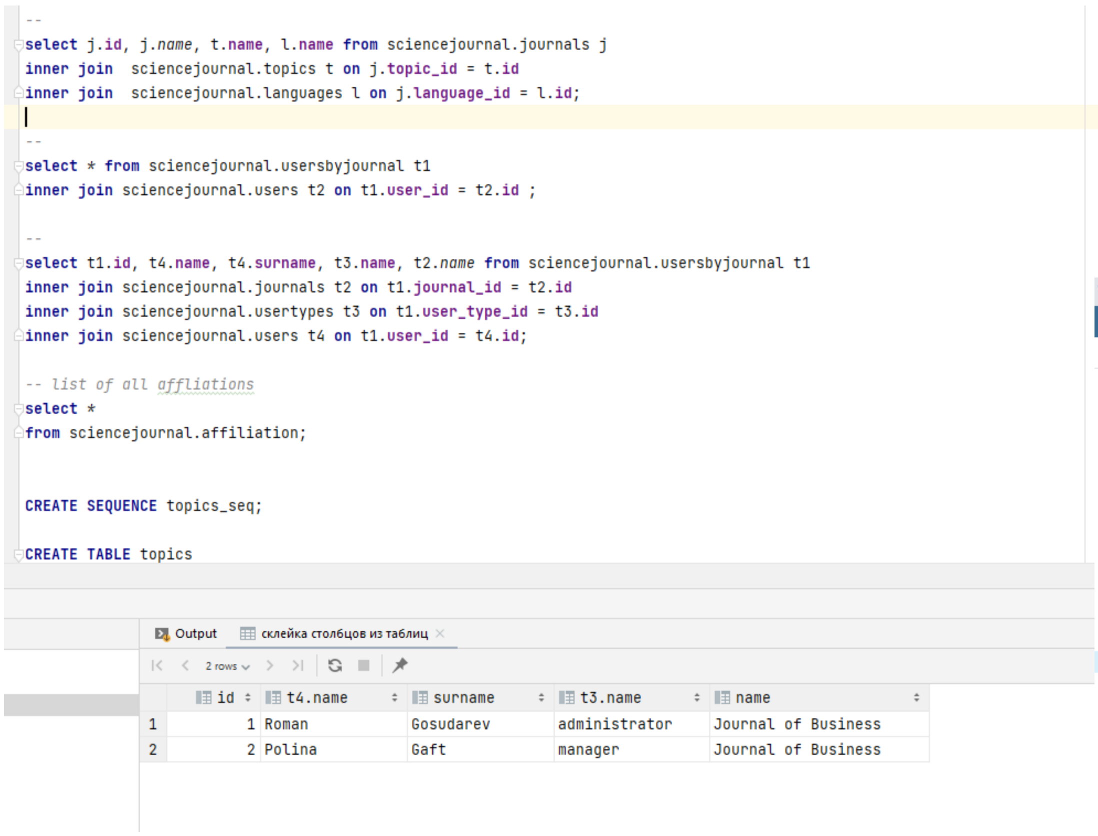
Liquibase


sciencejournal › src › main › resources › db › changelog › 202104162200_create_application_table_for_scien...
JournalApplication
Git:
Project
ler.java
SpringJDBCConfiguration.java
LoggingAspect.java
ApplicationService.java
ApplicationServiceImpl.java
console
202104162200_create_application_ta
Project
Structure
Pull Requests
Favorites

```
        UserTypeService
        JournalApplication
    resources
      db
        changelog
          1_add_functions.xml
          202104162000_create_sciencejournal_schema.xml
          202104162005_create_users_table_for_sciencejournal.xml
          202104162010_create_topics_table_for_sciencejournal.xml
          202104162015_create_languages_table_for_sciencejournal.xml
          202104162020_create_usertypes_table_for_sciencejournal.xml
          202104162025_create_applicationstatustypes_table_for_sciencejournal.xml
          202104162030_create_affiliation_table_for_sciencejournal.xml
          202104162035_create_author_table_for_sciencejournal.xml
          202104162040_create_configs_table_for_sciencejournal.xml
          202104162100_create_journals_table_for_sciencejournal.xml
          202104162115_create_users_by_journal_table_for_sciencejournal.xml
          202104162130_create_configs_by_journal_table_for_sciencejournal.xml
          202104162145_create_authors_by_affiliation_table_for_sciencejournal.xml
          202104162200_create_application_table_for_sciencejournal.xml
          202104162300_create_authorbyapplication_table_for_sciencejournal.xml
          202104162315_create_applicationflowhistory_table_for_sciencejournal.xml
          202104162330_insert data_for_languages_table.xml
          202104162345_insert data_for_topics_table.xml
          202104171100_insert data_for_usertypes_table.xml
          202104171430_insert data_for_affiliation_table.xml
          202104171700_insert data_for_applicationstatustypes_table.xml
          202104171930_insert data_for_journal_table.xml
          202104171945_insert data_for_users_table.xml
          202104172000_insert data_for_users_by_journal_table.xml
          202104172045_insert data_for_author_table.xml
          202104172050_insert data_for_authors_by_affiliation_table.xml
          202104172150_insert data_for_application_table.xml
          202104172205_insert data_for_author_by_application_table.xml
        liquibase-changelog.xml
      templates
        index.html
      application.properties
      liquibase.properties
  test
  target
  .gitignore
  pom.xml
  science journal.postman_collection.json
External Libraries
```

```xml
       http://www.liquibase.org/xml/ns/dbchangelog
       http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.10.xsd">

    <changeSet id="create_sequence_application_seq" author="alexander.kuznetsov" dbms="postgresql">
        <preConditions onFail="MARK_RAN">
            <sqlCheck expectedResult="f">
                select dbu_sequence_exists('application_seq', 'sciencejournal')
            </sqlCheck>
        </preConditions>
        <sql>
            CREATE SEQUENCE sciencejournal.application_seq;
        </sql>
    </changeSet>

    <changeSet id="create_table_application_for_sciencejournal" author="alexander.kuznetsov" dbms="postgresql">
        <preConditions onFail="MARK_RAN">
            <not>
                <tableExists tableName="application" schemaName="sciencejournal"/>
            </not>
            <sqlCheck expectedResult="t">select dbu_sequence_exists('application_seq', 'sciencejournal')</sqlCheck>
        </preConditions>
        <sql>
            CREATE TABLE sciencejournal.application (
            id              INTEGER         NOT NULL DEFAULT nextval('sciencejournal.application_seq'),
            title            VARCHAR(500)     NOT NULL,
            user_id         INTEGER         NOT NULL,
            journal_id      INTEGER         NOT NULL,
            application_status_id      INTEGER         NOT NULL,
            creation_date TIMESTAMP NOT NULL,
            last_modification TIMESTAMP NOT NULL,


            CONSTRAINT application_id_pk PRIMARY KEY (id),
            CONSTRAINT application_user_id_fk FOREIGN KEY (user_id) REFERENCES sciencejournal.users (id),
            CONSTRAINT application_journal_id_fk FOREIGN KEY (journal_id) REFERENCES sciencejournal.journals (id)
            );

            COMMENT ON TABLE sciencejournal.application IS 'List of application';
            COMMENT ON COLUMN sciencejournal.application.id IS 'ID';
            COMMENT ON COLUMN sciencejournal.application.title IS 'title of application';
```

PostgreSQL 13/pgAdmin

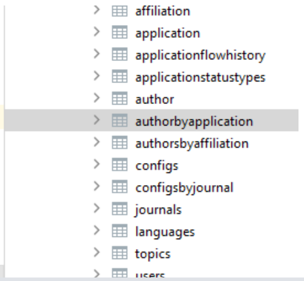


```sql
--
select j.id, j.name, t.name, l.name from sciencejournal.journals j
inner join  sciencejournal.topics t on j.topic_id = t.id
inner join  sciencejournal.languages l on j.language_id = l.id;

--
select * from sciencejournal.usersbyjournal t1
inner join sciencejournal.users t2 on t1.user_id = t2.id ;

--
select t1.id, t4.name, t4.surname, t3.name, t2.name from sciencejournal.usersbyjournal t1
inner join sciencejournal.journals t2 on t1.journal_id = t2.id
inner join sciencejournal.usertypes t3 on t1.user_type_id = t3.id
inner join sciencejournal.users t4 on t1.user_id = t4.id;

-- list of all affliations
select *
from sciencejournal.affiliation;


CREATE SEQUENCE topics_seq;

CREATE TABLE topics
```

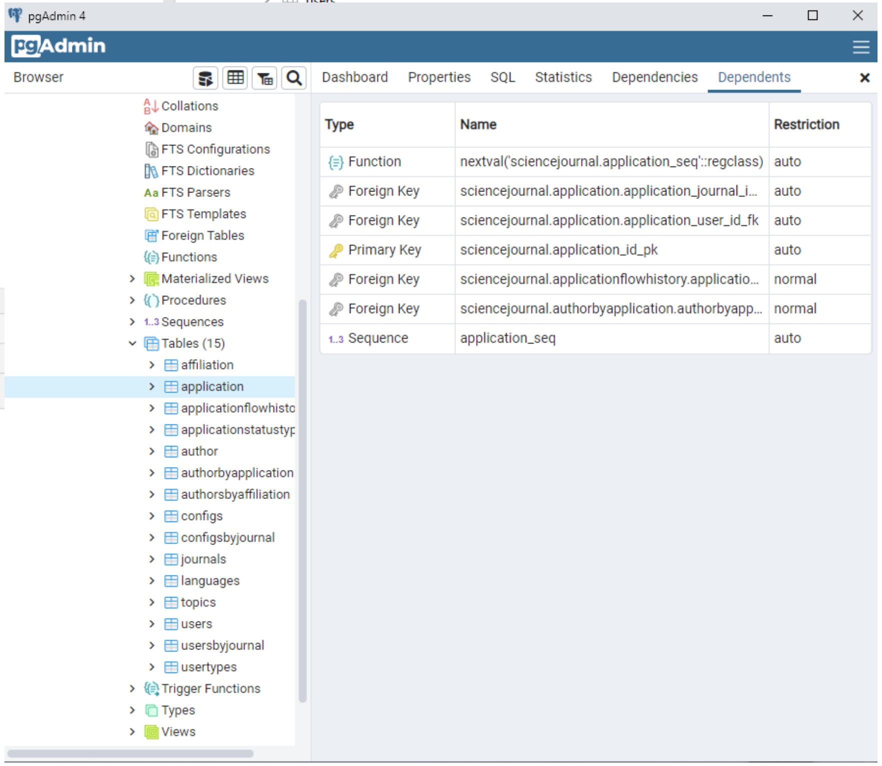
Output   склейка столбцов из таблиц

2 rows

	id	t4.name	surname	t3.name	name
1	1	Roman	Gosudarev	administrator	Journal of Business
2	2	Polina	Gaft	manager	Journal of Business

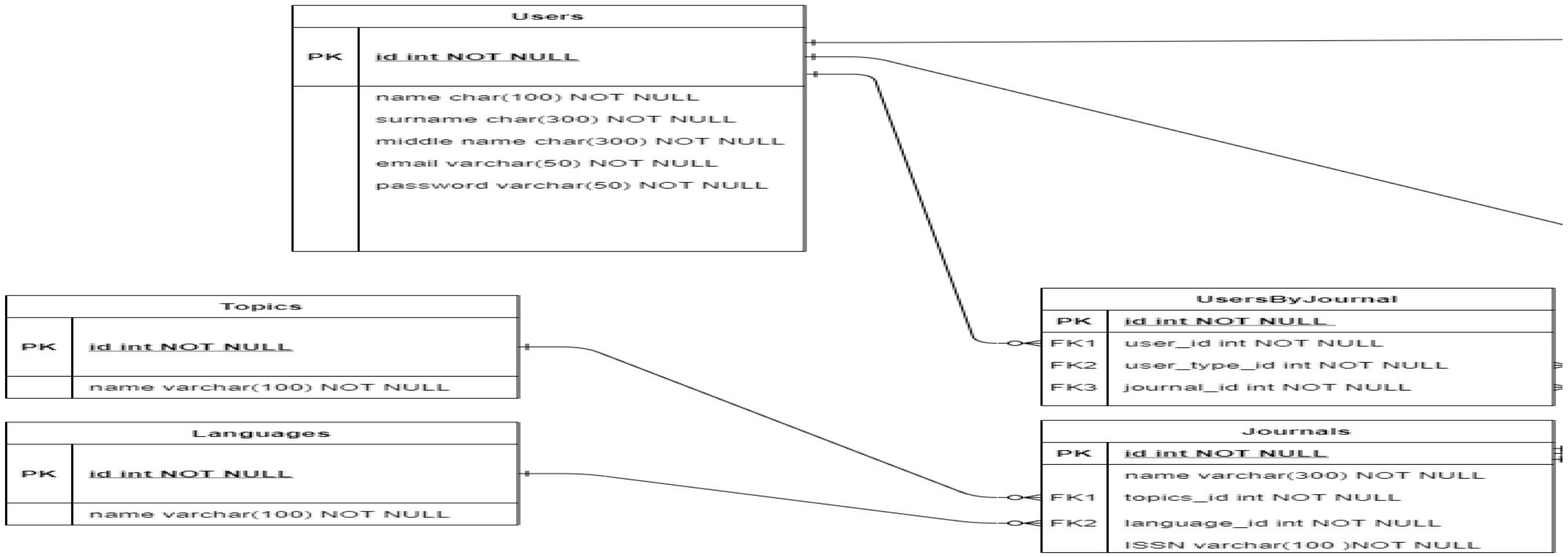


affiliation
application
applicationflowhistory
applicationstatustypes
author
authorbyapplication
authorsbyaffiliation
configs
configsbyjournal
journals
languages
topics
users

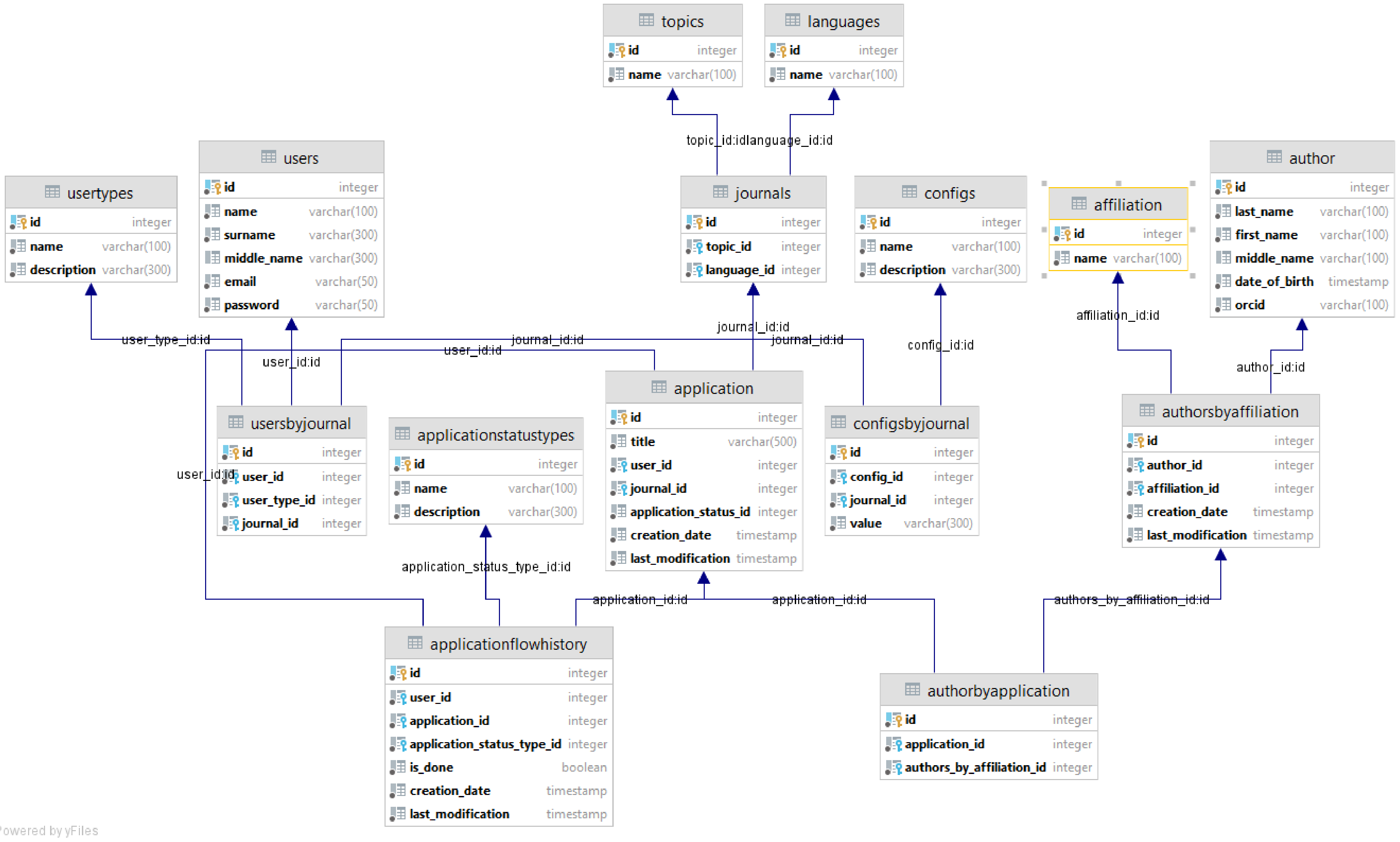


Type	Name	Restriction
Function	nextval('sciencejournal.application_seq'::regclass)	auto
Foreign Key	sciencejournal.application.application_journal_i...	auto
Foreign Key	sciencejournal.application.application_user_id_fk	auto
Primary Key	sciencejournal.application_id_pk	auto
Foreign Key	sciencejournal.applicationflowhistory.applicatio...	normal
Foreign Key	sciencejournal.authorbyapplication.authorbyapp...	normal
Sequence	application_seq	auto

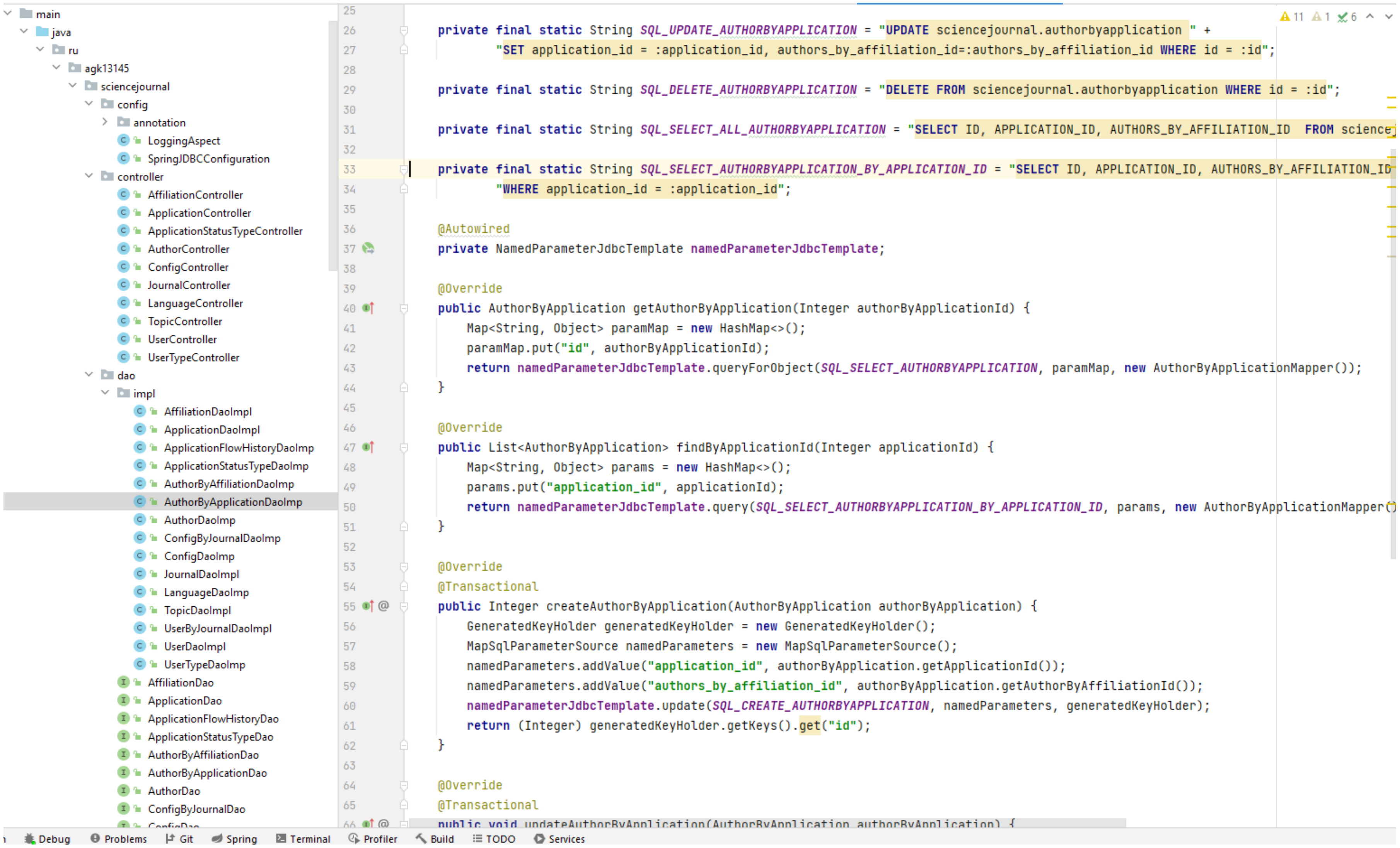
Database diagram (draw.io)



Database diagram (Exported from Intellij IDEA)



DAO for database tables



main
- java
  - ru
    - agk13145
      - sciencejournal
        - config
          - annotation
          - LoggingAspect
          - SpringJDBCConfiguration
        - controller
          - AffiliationController
          - ApplicationController
          - ApplicationStatusTypeController
          - AuthorController
          - ConfigController
          - JournalController
          - LanguageController
          - TopicController
          - UserController
          - UserTypeController
        - dao
          - impl
            - AffiliationDaoImpl
            - ApplicationDaoImpl
            - ApplicationFlowHistoryDaoImp
            - ApplicationStatusTypeDaoImp
            - AuthorByAffiliationDaoImp
            - AuthorByApplicationDaoImp
            - AuthorDaoImp
            - ConfigByJournalDaoImp
            - ConfigDaoImp
            - JournalDaoImpl
            - LanguageDaoImp
            - TopicDaoImpl
            - UserByJournalDaoImpl
            - UserDaoImpl
            - UserTypeDaoImp
          - AffiliationDao
          - ApplicationDao
          - ApplicationFlowHistoryDao
          - ApplicationStatusTypeDao
          - AuthorByAffiliationDao
          - AuthorByApplicationDao
          - AuthorDao
          - ConfigByJournalDao

```java
    private final static String SQL_UPDATE_AUTHORBYAPPLICATION = "UPDATE sciencejournal.authorbyapplication " +
            "SET application_id = :application_id, authors_by_affiliation_id=:authors_by_affiliation_id WHERE id = :id";

    private final static String SQL_DELETE_AUTHORBYAPPLICATION = "DELETE FROM sciencejournal.authorbyapplication WHERE id = :id";

    private final static String SQL_SELECT_ALL_AUTHORBYAPPLICATION = "SELECT ID, APPLICATION_ID, AUTHORS_BY_AFFILIATION_ID  FROM science

    private final static String SQL_SELECT_AUTHORBYAPPLICATION_BY_APPLICATION_ID = "SELECT ID, APPLICATION_ID, AUTHORS_BY_AFFILIATION_ID
            "WHERE application_id = :application_id";

    @Autowired
    private NamedParameterJdbcTemplate namedParameterJdbcTemplate;

    @Override
    public AuthorByApplication getAuthorByApplication(Integer authorByApplicationId) {
        Map<String, Object> paramMap = new HashMap<>();
        paramMap.put("id", authorByApplicationId);
        return namedParameterJdbcTemplate.queryForObject(SQL_SELECT_AUTHORBYAPPLICATION, paramMap, new AuthorByApplicationMapper());
    }

    @Override
    public List<AuthorByApplication> findByApplicationId(Integer applicationId) {
        Map<String, Object> params = new HashMap<>();
        params.put("application_id", applicationId);
        return namedParameterJdbcTemplate.query(SQL_SELECT_AUTHORBYAPPLICATION_BY_APPLICATION_ID, params, new AuthorByApplicationMapper()
    }

    @Override
    @Transactional
    public Integer createAuthorByApplication(AuthorByApplication authorByApplication) {
        GeneratedKeyHolder generatedKeyHolder = new GeneratedKeyHolder();
        MapSqlParameterSource namedParameters = new MapSqlParameterSource();
        namedParameters.addValue("application_id", authorByApplication.getApplicationId());
        namedParameters.addValue("authors_by_affiliation_id", authorByApplication.getAuthorByAffiliationId());
        namedParameterJdbcTemplate.update(SQL_CREATE_AUTHORBYAPPLICATION, namedParameters, generatedKeyHolder);
        return (Integer) generatedKeyHolder.getKeys().get("id");
    }

    @Override
    @Transactional
    public void updateAuthorByApplication(AuthorByApplication authorByApplication) {
```

Debug Problems Git Spring Terminal Profiler Build TODO Services

Live presentation

FUTURE

- create Review microservice + Notification microservice
- Add queue (RabbitMQ, Apache Kafka)
- Add Docker support (container runtime)
- Add tests

The End

Thanks for listening