AnimalController.java

package com.srt.controller;

import com.github.pagehelper.PageHelper;

import com.github.pagehelper.PageInfo;

import com.srt.pojo.\*;

import com.srt.service.AnimalsService;

import net.sf.json.JSONArray;

import net.sf.json.JSONObject;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

@Controller

@RequestMapping("Animals")

public class AnimalsController {

@Autowired

private AnimalsService animalsService;

// @RequestMapping("getInfo")

// @ResponseBody

// public List<Animals> getInfo(){

// return animalsService.getInfo();

// }

@RequestMapping("selectInfo")

@ResponseBody

public PageInfo selectInfo(@RequestBody Map map){

// String Species = map.get("Species").toString();

// return animalsService.selectInfo(Species);

PageInfo pageinfo = new PageInfo();

String Species = map.get("Species").toString();

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

PageHelper.startPage(pn, pageSize);

List<Animals> PageInfo = animalsService.selectInfo(Species);

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(PageInfo,5);

return page;

}

// @RequestMapping("selectInfoy")

// @ResponseBody

// public PageInfo selectInfoy(@RequestBody Map map){

// PageInfo pageinfo = new PageInfo();

// String Genus = map.get("Genus").toString();

// int pn=Integer.parseInt(map.get("pn").toString());

// int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

// PageHelper.startPage(pn, pageSize);

// List<Animals> PageInfo = animalsService.selectInfoy(Genus);

// // 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// // 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

// PageInfo page = new PageInfo(PageInfo,5);

// return page;

//

// }

// @RequestMapping("selectInfoType")

// @ResponseBody

// public PageInfo selectInfoType(@RequestBody Map map){

// PageInfo pageinfo = new PageInfo();

// String FirstClo = map.get("FirstClo").toString();

// int pn=Integer.parseInt(map.get("pn").toString());

// int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

// PageHelper.startPage(pn, pageSize);

// List<Animals> PageInfo = animalsService.selectInfoType(FirstClo);

// // 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// // 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

// PageInfo page = new PageInfo(PageInfo,5);

// return page;

//

// }

// public List<Animals> selectInfoy(@RequestBody Map map){

// String Genus = map.get("Genus").toString();

//

// return animalsService.selectInfoy(Genus);

// }

@RequestMapping("getPageInfo")

@ResponseBody

public PageInfo getPageInfo(@RequestBody Map map){

PageInfo pageinfo = new PageInfo();

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

PageHelper.startPage(pn, pageSize);

List<Animals> PageInfo = animalsService.getPageInfo();

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(PageInfo,5);

return page;

/\* int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

PageInfo pageInfo = new PageInfo();

List<Animals> animals = animalsService.getPageInfo((pn - 1) \* pageSize,pageSize);

long counts = animalsService.getCounts() / 2;

int pages = (int) Math.round(counts / pageSize + 0.5);

pageInfo.setList(animals);

pageInfo.setTotal(counts);

pageInfo.setPages(pages);

pageInfo.setSize(pages);

pageInfo.setPageNum(pn);

pageInfo.setPageSize(pageSize);

pageInfo.setStartRow(0);

pageInfo.setEndRow(animals.size() > 0 ? animals.size() - 1 : 0);

pageInfo.setNavigatePages(5);

pageInfo.setHasPreviousPage(pn != 1);

pageInfo.setHasNextPage(pn != pages);

pageInfo.setIsFirstPage(pn == 1);

pageInfo.setIsLastPage(pn == pages);

calNav(pageInfo);

pageInfo.setNavigateFirstPage(1);

pageInfo.setNavigateLastPage(pages);

return pageInfo;\*/

}

@RequestMapping("selectChangeInfo")

@ResponseBody

public JSONObject selectChangeInfo(@RequestBody Map map){

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

String species=map.get("Species").toString();

String genus=map.get("Genus").toString();

String bioprojecttype=map.get("Bioprojecttype").toString();

String gang=map.get("gang").toString();

String mu=map.get("mu").toString();

String ke=map.get("ke").toString();

String shu=map.get("shu").toString();

String zhong=map.get("zhong").toString();

String sampleSite0 = map.get("sampleSite0").toString();

String sampleSite1 = map.get("sampleSite1").toString();

// List<Animals> ExhibitionInfo0 = animalsService.selectChangeInfo(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

// List<String> outline = map.get("Bioprojecttype").toString();

PageHelper.startPage(pn, pageSize);

List<Animals> ExhibitionInfo = animalsService.selectChangeInfo(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(ExhibitionInfo,5);

JSONObject obj = new JSONObject();

// List<String> speciesList = new ArrayList<>();

// List<String> sampleSiteList = new ArrayList<>();

// for (int i = 0; i < ExhibitionInfo0.size(); i++) {

// if(!speciesList.contains(ExhibitionInfo0.get(i).getSpecies()))

// speciesList.add(ExhibitionInfo0.get(i).getSpecies());

// if(!sampleSiteList.contains(ExhibitionInfo0.get(i).getStudyType()))

// sampleSiteList.add(ExhibitionInfo0.get(i).getStudyType());

// }

obj.put("page",page);

JSONObject searchOptions = animalsService.getSearchOptions(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

obj.put("objs",searchOptions);

// obj.put("speciesList",speciesList);

// obj.put("sampleSiteList",sampleSiteList);

return obj;

/\* List<Animals> ExhibitionInfo = animalsService.selectChangeInfo(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,(pn - 1) \* pageSize,pageSize);

PageInfo pageInfo = new PageInfo();

long counts = animalsService.getCounts() / 2;

int pages = (int) Math.round(counts / pageSize + 0.5);

pageInfo.setList(ExhibitionInfo);

pageInfo.setTotal(counts);

pageInfo.setPages(pages);

pageInfo.setSize(pages);

pageInfo.setPageNum(pn);

pageInfo.setPageSize(pageSize);

pageInfo.setStartRow(0);

pageInfo.setEndRow(ExhibitionInfo.size() > 0 ? ExhibitionInfo.size() - 1 : 0);

pageInfo.setNavigatePages(5);

pageInfo.setHasPreviousPage(pn != 1);

pageInfo.setHasNextPage(pn != pages);

pageInfo.setIsFirstPage(pn == 1);

pageInfo.setIsLastPage(pn == pages);

calNav(pageInfo);

pageInfo.setNavigateFirstPage(1);

pageInfo.setNavigateLastPage(pages);

return pageInfo;\*/

}

@RequestMapping("selectChangeInfoOrder")

@ResponseBody

public JSONObject selectChangeInfoOrder(@RequestBody Map map){

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

String species=map.get("Species").toString();

String genus=map.get("Genus").toString();

String bioprojecttype=map.get("Bioprojecttype").toString();

String gang=map.get("gang").toString();

String mu=map.get("mu").toString();

String ke=map.get("ke").toString();

String shu=map.get("shu").toString();

String zhong=map.get("zhong").toString();

String sampleSite0 = map.get("sampleSite0").toString();

String sampleSite1 = map.get("sampleSite1").toString();

String orderField = map.get("orderField").toString();

String order = map.get("order").toString();

// List<String> outline = map.get("Bioprojecttype").toString();

PageHelper.startPage(pn, pageSize);

List<Animals> ExhibitionInfo = animalsService.selectChangeInfoOrder(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1,orderField,order);

PageInfo page = new PageInfo(ExhibitionInfo,5);

JSONObject obj = new JSONObject();

obj.put("page",page);

return obj;

}

@RequestMapping("selectAll")

@ResponseBody

public List<Animals> selectAll(@RequestBody Map map){

String species=map.get("Species").toString();

String genus=map.get("Genus").toString();

String bioprojecttype=map.get("Bioprojecttype").toString();

String gang=map.get("gang").toString();

String mu=map.get("mu").toString();

String ke=map.get("ke").toString();

String shu=map.get("shu").toString();

String zhong=map.get("zhong").toString();

String sampleSite0 = map.get("sampleSite0").toString();

String sampleSite1 = map.get("sampleSite1").toString();

List<Animals> ExhibitionInfo0 = animalsService.selectChangeInfo(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

return ExhibitionInfo0;

}

@RequestMapping("getAllInfo")

@ResponseBody

public List<Animals> getInfo(){

return animalsService.getInfo();

}

@RequestMapping("getGroup")

@ResponseBody

public List<Animals> getGroup(){

return animalsService.getGroup();

}

@RequestMapping("getGroupTypeOne")

@ResponseBody

public List<Animals> getGroupType(){

return animalsService.getGroupTypeOne();

}

@RequestMapping("getGroupTypeTwo")

@ResponseBody

public List<Animals> getGroupTypeTwo(String FirstClo){

return animalsService.getGroupTypeTwo(FirstClo);

}

@RequestMapping("getBioProjectType")

@ResponseBody

public List<Bioproject> getBioProjectType(){

return animalsService.getBioProjectType();

}

@RequestMapping("getOutline")

@ResponseBody

public List<Gmksz> getOutline(){

return animalsService.getOutline();

}

@RequestMapping("getOrders")

@ResponseBody

public List<Animals> getOrders(String gang){

return animalsService.getOrders(gang);

}

public void calNav(PageInfo pageInfo){

int i;

if (pageInfo.getPages() <= pageInfo.getNavigatePages()) {

pageInfo.setNavigatepageNums(new int[pageInfo.getPages()]);

for(i = 0; i < pageInfo.getPages(); ++i) {

pageInfo.getNavigatepageNums()[i] = i + 1;

}

} else {

pageInfo.setNavigatepageNums(new int[pageInfo.getNavigatePages()]);

i = pageInfo.getPageNum() - pageInfo.getNavigatePages() / 2;

int endNum = pageInfo.getPageNum() + pageInfo.getNavigatePages() / 2;

if (i < 1) {

i = 1;

for(i = 0; i < pageInfo.getNavigatePages(); ++i) {

pageInfo.getNavigatepageNums()[i] = i++;

}

} else if (endNum > pageInfo.getPages()) {

endNum = pageInfo.getPages();

for(i = pageInfo.getNavigatePages() - 1; i >= 0; --i) {

pageInfo.getNavigatepageNums()[i] = endNum--;

}

} else {

for(i = 0; i < pageInfo.getNavigatePages(); ++i) {

pageInfo.getNavigatepageNums()[i] = i++;

}

}

}

}

@RequestMapping("getTree")

@ResponseBody

public List<Gmksz> getTree(){

return animalsService.getTree();

}

}

SearchController.java

package com.srt.controller;

import com.github.pagehelper.PageHelper;

import com.github.pagehelper.PageInfo;

import com.srt.pojo.Animals;

import com.srt.pojo.Bioproject;

import com.srt.service.SearchService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseBody;

import java.util.List;

import java.util.Map;

@Controller

@RequestMapping("Exhibition")

public class SearchController {

@Autowired

private SearchService searchService;

/\*\*

\* 展示数据

\* @param map

\* @return

\*/

@RequestMapping("getExhibitionInfo")

@ResponseBody

public PageInfo getExhibitionInfo(@RequestBody Map map){

PageInfo pageinfo = new PageInfo();

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

PageHelper.startPage(pn, pageSize);

List<Animals> ExhibitionInfo = searchService.getExhibitionInfo();

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(ExhibitionInfo,5);

return page;

}

/\*\*

\* 搜索信息

\* @param map

\* @return

\*/

@RequestMapping("selectInfo")

@ResponseBody

public PageInfo selectInfo(@RequestBody Map map){

PageInfo pageinfo = new PageInfo();

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

String species=map.get("species").toString();

String sampleSite1=map.get("sampleSite1").toString();

// String sampleSite2=map.get("sampleSite2").toString();

String instrument=map.get("instrument").toString();

String libraryStrategy=map.get("libraryStrategy").toString();

String newStudyType=map.get("newStudyType").toString();

String newDna=map.get("newDna").toString();

String newSex=map.get("newSex").toString();

String startDate=map.get("startDate").toString();

String endDate=map.get("endDate").toString();

String startMonth=map.get("startMonth").toString();

String endMonth=map.get("endMonth").toString();

String orderField = map.get("orderField").toString();

String order = map.get("order").toString();

PageHelper.startPage(pn, pageSize);

List<Animals> ExhibitionInfo;

if(order.equals(""))

ExhibitionInfo = searchService.selectInfo(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth);

else

ExhibitionInfo = searchService.selectInfoOrder(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth,orderField,order);

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(ExhibitionInfo,5);

return page;

}

@RequestMapping("selectChangeInfo")

@ResponseBody

public PageInfo selectChangeInfo(@RequestBody Map map){

PageInfo pageinfo = new PageInfo();

int pn=Integer.parseInt(map.get("pn").toString());

int pageSize=Integer.parseInt(map.get("pageSize").toString()) ;

String species=map.get("species").toString();

String sampleSite1=map.get("sampleSite1").toString();

// String sampleSite2=map.get("sampleSite2").toString();

String instrument=map.get("instrument").toString();

String libraryStrategy=map.get("libraryStrategy").toString();

String newStudyType=map.get("newStudyType").toString();

String newDna=map.get("newDna").toString();

String newSex=map.get("newSex").toString();

String startDate=map.get("startDate").toString();

String endDate=map.get("endDate").toString();

String startMonth=map.get("startMonth").toString();

String endMonth=map.get("endMonth").toString();

String orderField = map.get("orderField").toString();

String order = map.get("order").toString();

PageHelper.startPage(pn, pageSize);

List<Animals> ExhibitionInfo;

if(order.equals(""))

ExhibitionInfo = searchService.selectChangeInfo(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth);

else

ExhibitionInfo = searchService.selectChangeInfoOrder(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth,orderField,order);

// 使用pageInfo包装查询后的结果，只需要将pageInfo交给页面就行了。

// 封装了详细的分页信息,包括有我们查询出来的数据，传入连续显示的页数

PageInfo page = new PageInfo(ExhibitionInfo,5);

return page;

}

// @RequestMapping("getSkip")

// @ResponseBody

// public List<Animals> selectWebById(@RequestBody Map map){

// String id = map.get("id").toString();

// return searchService.selectWebById(id);

// }

/\*\*

\* 拉取下载所有数据

\* @return

\*/

@RequestMapping("getAll")

@ResponseBody

public List<Animals> getAll(@RequestBody Map map){

String species=map.get("species").toString();

String sampleSite1=map.get("sampleSite1").toString();

// String sampleSite2=map.get("sampleSite2").toString();

String instrument=map.get("instrument").toString();

String libraryStrategy=map.get("libraryStrategy").toString();

String bioprojecttype=map.get("bioprojecttype").toString();

List<Animals> Info = searchService.getAll(species,sampleSite1,instrument,libraryStrategy,bioprojecttype);

return Info;

}

@RequestMapping("getAllInfo")

@ResponseBody

public List<Animals> getInfo(){

return searchService.getInfo();

}

// 六个搜索框内容

@RequestMapping("getSpecies")

@ResponseBody

public List<Animals> getSpecies(){

return searchService.getSpecies();

}

@RequestMapping("getSampleSite1")

@ResponseBody

public List<Animals> getSampleSite1(){

return searchService.getSampleSite1();

}

@RequestMapping("getSampleSite2")

@ResponseBody

public List<Animals> getSampleSite2(){

return searchService.getSampleSite2();

}

@RequestMapping("getInstrument")

@ResponseBody

public List<Animals> getInstrument(){

return searchService.getInstrument();

}

@RequestMapping("getLibraryStrategy")

@ResponseBody

public List<Animals> getLibraryStrategy(){

return searchService.getLibraryStrategy();

}

@RequestMapping("getBioProjectType")

@ResponseBody

public List<Bioproject> getBioProjectType(){

return searchService.getBioProjectType();

}

//

// @RequestMapping("getSpecies")

// @ResponseBody

// public List<Animals> getSpecies(){

// return searchService.getSpecies();

// }

}

Animal.java

package com.srt.pojo;

public class Animals {

private String experimentAccession;

private String experimentTitle;

private String organismName;

private String instrument;

private String submitter;

private String studyAccession;

private String studyTitle;

private String sampleAccession;

private String sampleTitle;

private String totalSize;

private String totalRuns;

private String totalSpots;

private String totalBases;

private String sampleName;

private String libraryStrategy;

private String librarySource;

private String librarySelection;

private String bioprojectAccession;

private String biosampleAccession;

private String sampleSite;

private String sex;

private String age;

private String creationDate;

private String collectionDate;

private String geographicLocation;

private String latitude;

private String longitude;

private String pubmedId;

private String conditions;

private String pheotype;

private String breed;

private String sourceDatabase;

private String bioprojectDescription;

private String sampleNumber;

private String lianjieb;

private String outline;

private String orders;

private String family;

private String genus;

private String species;

private String lianjiep;

private String lianjies;

private String lianjiee;

private String bioprojectType;

private String sampleType;

private String studyType;

private String DNAExtractionMethod;

private String lianjienew;

private String extra1;

private String extra2;

// private String firstClo;

// private String secondClo;

// public String getFirstClo() {

// return firstClo;

// }

//

// public void setFirstClo(String firstClo) {

// this.firstClo = firstClo;

// }

//

// public String getSecondClo() {

// return secondClo;

// }

//

// public void setSecondClo(String secondClo) {

// this.secondClo = secondClo;

// }

public String getExperimentAccession() {

return experimentAccession;

}

public void setExperimentAccession(String experimentAccession) {

this.experimentAccession = experimentAccession == null ? null : experimentAccession.trim();

}

public String getExperimentTitle() {

return experimentTitle;

}

public void setExperimentTitle(String experimentTitle) {

this.experimentTitle = experimentTitle == null ? null : experimentTitle.trim();

}

public String getOrganismName() {

return organismName;

}

public void setOrganismName(String organismName) {

this.organismName = organismName == null ? null : organismName.trim();

}

public String getInstrument() {

return instrument;

}

public void setInstrument(String instrument) {

this.instrument = instrument == null ? null : instrument.trim();

}

public String getSubmitter() {

return submitter;

}

public void setSubmitter(String submitter) {

this.submitter = submitter == null ? null : submitter.trim();

}

public String getStudyAccession() {

return studyAccession;

}

public void setStudyAccession(String studyAccession) {

this.studyAccession = studyAccession == null ? null : studyAccession.trim();

}

public String getStudyTitle() {

return studyTitle;

}

public void setStudyTitle(String studyTitle) {

this.studyTitle = studyTitle == null ? null : studyTitle.trim();

}

public String getSampleAccession() {

return sampleAccession;

}

public void setSampleAccession(String sampleAccession) {

this.sampleAccession = sampleAccession == null ? null : sampleAccession.trim();

}

public String getSampleTitle() {

return sampleTitle;

}

public void setSampleTitle(String sampleTitle) {

this.sampleTitle = sampleTitle == null ? null : sampleTitle.trim();

}

public String getTotalSize() {

return totalSize;

}

public void setTotalSize(String totalSize) {

this.totalSize = totalSize == null ? null : totalSize.trim();

}

public String getTotalRuns() {

return totalRuns;

}

public void setTotalRuns(String totalRuns) {

this.totalRuns = totalRuns == null ? null : totalRuns.trim();

}

public String getTotalSpots() {

return totalSpots;

}

public void setTotalSpots(String totalSpots) {

this.totalSpots = totalSpots == null ? null : totalSpots.trim();

}

public String getTotalBases() {

return totalBases;

}

public void setTotalBases(String totalBases) {

this.totalBases = totalBases == null ? null : totalBases.trim();

}

public String getSampleName() {

return sampleName;

}

public void setSampleName(String sampleName) {

this.sampleName = sampleName == null ? null : sampleName.trim();

}

public String getLibraryStrategy() {

return libraryStrategy;

}

public void setLibraryStrategy(String libraryStrategy) {

this.libraryStrategy = libraryStrategy == null ? null : libraryStrategy.trim();

}

public String getLibrarySource() {

return librarySource;

}

public void setLibrarySource(String librarySource) {

this.librarySource = librarySource == null ? null : librarySource.trim();

}

public String getLibrarySelection() {

return librarySelection;

}

public void setLibrarySelection(String librarySelection) {

this.librarySelection = librarySelection == null ? null : librarySelection.trim();

}

public String getBioprojectAccession() {

return bioprojectAccession;

}

public void setBioprojectAccession(String bioprojectAccession) {

this.bioprojectAccession = bioprojectAccession == null ? null : bioprojectAccession.trim();

}

public String getBiosampleAccession() {

return biosampleAccession;

}

public void setBiosampleAccession(String biosampleAccession) {

this.biosampleAccession = biosampleAccession == null ? null : biosampleAccession.trim();

}

public String getSampleSite() {

return sampleSite;

}

public void setSampleSite(String sampleSite) {

this.sampleSite = sampleSite == null ? null : sampleSite.trim();

}

public String getSex() {

return sex;

}

public void setSex(String sex) {

this.sex = sex == null ? null : sex.trim();

}

public String getAge() {

return age;

}

public void setAge(String age) {

this.age = age == null ? null : age.trim();

}

public String getCreationDate() {

return creationDate;

}

public void setCreationDate(String creationDate) {

this.creationDate = creationDate == null ? null : creationDate.trim();

}

public String getCollectionDate() {

return collectionDate;

}

public void setCollectionDate(String collectionDate) {

this.collectionDate = collectionDate == null ? null : collectionDate.trim();

}

public String getGeographicLocation() {

return geographicLocation;

}

public void setGeographicLocation(String geographicLocation) {

this.geographicLocation = geographicLocation == null ? null : geographicLocation.trim();

}

public String getLatitude() {

return latitude;

}

public void setLatitude(String latitude) {

this.latitude = latitude == null ? null : latitude.trim();

}

public String getLongitude() {

return longitude;

}

public void setLongitude(String longitude) {

this.longitude = longitude == null ? null : longitude.trim();

}

public String getPubmedId() {

return pubmedId;

}

public void setPubmedId(String pubmedId) {

this.pubmedId = pubmedId == null ? null : pubmedId.trim();

}

public String getConditions() {

return conditions;

}

public void setConditions(String conditions) {

this.conditions = conditions == null ? null : conditions.trim();

}

public String getPheotype() {

return pheotype;

}

public void setPheotype(String pheotype) {

this.pheotype = pheotype == null ? null : pheotype.trim();

}

public String getBreed() {

return breed;

}

public void setBreed(String breed) {

this.breed = breed == null ? null : breed.trim();

}

public String getSourceDatabase() {

return sourceDatabase;

}

public void setSourceDatabase(String sourceDatabase) {

this.sourceDatabase = sourceDatabase == null ? null : sourceDatabase.trim();

}

public String getBioprojectDescription() {

return bioprojectDescription;

}

public void setBioprojectDescription(String bioprojectDescription) {

this.bioprojectDescription = bioprojectDescription == null ? null : bioprojectDescription.trim();

}

public String getSampleNumber() {

return sampleNumber;

}

public void setSampleNumber(String sampleNumber) {

this.sampleNumber = sampleNumber == null ? null : sampleNumber.trim();

}

public String getLianjieb() {

return lianjieb;

}

public void setLianjieb(String lianjieb) {

this.lianjieb = lianjieb == null ? null : lianjieb.trim();

}

public String getOutline() {

return outline;

}

public void setOutline(String outline) {

this.outline = outline == null ? null : outline.trim();

}

public String getOrders() {

return orders;

}

public void setOrders(String orders) {

this.orders = orders == null ? null : orders.trim();

}

public String getFamily() {

return family;

}

public void setFamily(String family) {

this.family = family == null ? null : family.trim();

}

public String getGenus() {

return genus;

}

public void setGenus(String genus) {

this.genus = genus == null ? null : genus.trim();

}

public String getSpecies() {

return species;

}

public void setSpecies(String species) {

this.species = species == null ? null : species.trim();

}

public String getLianjiep() {

return lianjiep;

}

public void setLianjiep(String lianjiep) {

this.lianjiep = lianjiep == null ? null : lianjiep.trim();

}

public String getLianjies() {

return lianjies;

}

public void setLianjies(String lianjies) {

this.lianjies = lianjies == null ? null : lianjies.trim();

}

public String getLianjiee() {

return lianjiee;

}

public void setLianjiee(String lianjiee) {

this.lianjiee = lianjiee == null ? null : lianjiee.trim();

}

public String getBioprojectType() {

return bioprojectType;

}

public void setBioprojectType(String bioprojectType) {

this.bioprojectType = bioprojectType == null ? null : bioprojectType.trim();

}

public String getSampleType() {

return sampleType;

}

public void setSampleType(String sampleType) {

this.sampleType = sampleType == null ? null : sampleType.trim();

}

public String getStudyType() {

return studyType;

}

public void setStudyType(String studyType) {

this.studyType = studyType;

}

public String getDNAExtractionMethod() {

return DNAExtractionMethod;

}

public void setDNAExtractionMethod(String DNAExtractionMethod) {

this.DNAExtractionMethod = DNAExtractionMethod;

}

public String getLianjienew() {

return lianjienew;

}

public void setLianjienew(String lianjienew) {

this.lianjienew = lianjienew;

}

public String getExtra1() {

return extra1;

}

public void setExtra1(String extra1) {

this.extra1 = extra1;

}

public String getExtra2() {

return extra2;

}

public void setExtra2(String extra2) {

this.extra2 = extra2;

}

}

AnimalService.java

package com.srt.service;

import com.srt.pojo.\*;

import net.sf.json.JSONArray;

import net.sf.json.JSONObject;

import java.util.List;

public interface AnimalsService {

List<Animals> getInfo();

List<Animals> getGroup();

List<Animals> selectChangeInfo(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1);

List<Animals> getGroupTypeOne();

List<Animals> getGroupTypeTwo(String FirstClo);

List<Animals> selectInfo(String Species);

List<Animals> selectInfoy(String Genus);

List<Animals> getPageInfo();

List<Animals> selectInfoType(String FirstClo);

List<Gmksz> getOutline();

List<Bioproject> getBioProjectType();

List<Animals> getOrders(String gang);

List<Animals> selectAll(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong);

List<Animals> getPageInfo(int i, int pageSize);

long getCounts();

List<Animals> selectChangeInfo(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, int i, int pageSize);

List<Gmksz> getTree();

List<Animals> selectChangeInfoOrder(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1, String orderField, String order);

JSONObject getSearchOptions(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1);

}

AnimalServiceImpl.java

package com.srt.service.impl;

import com.srt.mapper.AnimalsMapper;

import com.srt.mapper.BioprojectMapper;

import com.srt.mapper.GmkszMapper;

import com.srt.pojo.\*;

import com.srt.service.AnimalsService;

import net.sf.json.JSON;

import net.sf.json.JSONArray;

import net.sf.json.JSONObject;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class AnimalsServiceImpl implements AnimalsService {

@Autowired

private AnimalsMapper animalsMapper;

@Autowired

private BioprojectMapper bioprojectMapper;

@Autowired

private GmkszMapper gmkszMapper;

public List<Animals> getGroup(){

List<Animals> result = animalsMapper.selectByGenusl();

return result;

}

public List<Animals> getGroupTypeOne(){

List<Animals> result = animalsMapper.selectByBioprojectTypeOne();

return result;

}

public List<Animals> getGroupTypeTwo(String FirstClo){

List<Animals> result = animalsMapper.selectByBioprojectTypeTwo(FirstClo);

return result;

}

public List<Animals> selectInfo(String Species) {

List<Animals> result = animalsMapper.selectBySpecies(Species);

return result;

}

public List<Animals> getInfo(){

List<Animals> Info = animalsMapper.selectByExample(null);

return Info;

}

public List<Animals> selectInfoy(String Genus) {

List<Animals> result = animalsMapper.selectByGenus(Genus);

return result;

}

public List<Animals> selectInfoType(String FirstClo) {

List<Animals> result = animalsMapper.selectByType(FirstClo);

return result;

}

public List<Animals> getPageInfo(){

List<Animals> result = animalsMapper.selectExample(null);

return result;

}

// public List<Animals> getOutline(){

// List<Animals> result = animalsMapper.selectByG(null);

// return result;

// }

public List<Gmksz> getOutline(){

List<Gmksz> result = gmkszMapper.selectByExample(null);

return result;

}

public List<Bioproject> getBioProjectType(){

List<Bioproject> result = bioprojectMapper.selectByExample(null);

return result;

}

public List<Animals> getOrders(String gang){

List<Animals> result = animalsMapper.selectByM(gang);

return result;

}

public List<Animals> selectChangeInfo(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1) {

List<Animals> result = animalsMapper.selectOnChangeInfo(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

return result;

}

public List<Animals> selectAll(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong){

List<Animals> result = animalsMapper.selectAll(species,genus,bioprojecttype,gang,mu,ke,shu,zhong);

return result;

}

@Override

public List<Animals> getPageInfo(int i, int pageSize) {

return animalsMapper.selectParts(i,pageSize);

}

@Override

public long getCounts() {

return animalsMapper.countByView();

}

@Override

public List<Animals> selectChangeInfo(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, int i, int pageSize) {

List<Animals> result = animalsMapper.selectOnChangeInfoParts(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,i,pageSize);

return result;

}

@Override

public List<Gmksz> getTree() {

List<Gmksz> gmkszs = gmkszMapper.selectByParentId(0);

for(int i = 0;i < gmkszs.size();i++){

buildTree(gmkszs.get(i));

}

// List<Bioproject> parents = bioprojectMapper.selectByParentId(0);

// for(int i = 0;i < parents.size();i++){

// buildTree(parents.get(i));

// }

return gmkszs;

}

@Override

public List<Animals> selectChangeInfoOrder(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1, String orderField, String order) {

String cause = getMapping(orderField).equals("") ? orderField : getMapping(orderField);

if(order.equals("descend"))

cause += " DESC";

else if(order.equals("ascend"))

cause += " ASC";

List<Animals> result = animalsMapper.selectOnChangeInfoOrder(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1,getMapping(orderField).equals("") ? orderField : getMapping(orderField),cause);

return result;

}

@Override

public JSONObject getSearchOptions(String species, String genus, String bioprojecttype, String gang, String mu, String ke, String shu, String zhong, String sampleSite0, String sampleSite1) {

JSONObject tmp = new JSONObject();

List<String> speciesOpt = animalsMapper.selectOptionsForSpecies(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

List<SampleSiteTmp> sampleSiteTmpList = animalsMapper.selectOptionsForSampleSite(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

JSONArray sampleSiteOpt = new JSONArray();

String mark0 = "";

for(int i = 0;i < sampleSiteTmpList.size();){

if(i == 0)

mark0 = sampleSiteTmpList.get(0).getSampleSite0();

JSONObject father = new JSONObject();

JSONArray children = new JSONArray();

int j = i;

for(;j < sampleSiteTmpList.size() && sampleSiteTmpList.get(j).getSampleSite0().equals(mark0);j++){

JSONObject childrenObj = new JSONObject();

childrenObj.put("label", sampleSiteTmpList.get(j).getSampleSite1());

childrenObj.put("value", sampleSiteTmpList.get(j).getSampleSite1());

children.add(childrenObj);

}

father.put("label", mark0);

father.put("value", mark0);

father.put("children", children);

sampleSiteOpt.add(father);

if(j < sampleSiteTmpList.size())

mark0 = sampleSiteTmpList.get(j).getSampleSite0();

i = j;

}

List<StudyTypeTmp> studyTypeTmpList = animalsMapper.selectOptionsForStudyType(species,genus,bioprojecttype,gang,mu,ke,shu,zhong,sampleSite0,sampleSite1);

JSONArray studyTypeOpt = new JSONArray();

for(int i = 0;i < studyTypeTmpList.size();){

if(i == 0)

mark0 = studyTypeTmpList.get(0).getLevel1();

JSONObject father = new JSONObject();

JSONArray children = new JSONArray();

int j = i;

String mark1 = "";

for(;j < studyTypeTmpList.size() && studyTypeTmpList.get(j).getLevel1().equals(mark0);){

if(j == 0)

mark1 = studyTypeTmpList.get(0).getLevel2();

JSONObject father1 = new JSONObject();

JSONArray children1 = new JSONArray();

int k = j;

for(;k < studyTypeTmpList.size() && studyTypeTmpList.get(k).getLevel2().equals(mark1);k++){

JSONObject childrenObj = new JSONObject();

childrenObj.put("label", studyTypeTmpList.get(k).getLevel3());

childrenObj.put("value", studyTypeTmpList.get(k).getLevel3());

children1.add(childrenObj);

}

father1.put("label", mark1);

father1.put("value", mark1);

father1.put("children", children1);

children.add(father1);

if(k < studyTypeTmpList.size())

mark1 = studyTypeTmpList.get(k).getLevel2();

j = k;

}

father.put("label", mark0);

father.put("value", mark0);

father.put("children", children);

studyTypeOpt.add(father);

if(j < studyTypeTmpList.size())

mark0 = studyTypeTmpList.get(j).getLevel1();

i = j;

}

tmp.put("speciesOpt", speciesOpt);

tmp.put("sampleSiteOpt", sampleSiteOpt);

tmp.put("studyTypeOpt", studyTypeOpt);

return tmp;

}

public void buildTree(Bioproject bioproject){

List<Bioproject> tmp = bioprojectMapper.selectByParentId(bioproject.getId());

if(tmp.size() > 0){

bioproject.setSons(tmp);

for(int i = 0;i < tmp.size();i++){

buildTree(tmp.get(i));

}

}

}

public void buildTree(Gmksz bioproject){

List<Gmksz> tmp = gmkszMapper.selectByParentId(bioproject.getId());

if(tmp.size() > 0){

bioproject.setSons(tmp);

for(int i = 0;i < tmp.size();i++){

buildTree(tmp.get(i));

}

}

}

private String getMapping(String source){

String str = "";

switch (source){

case "bioprojectAccession":

str = "Bioproject\_Accession";

break;

case "bioprojectDescription":

str = "Bioproject\_Description";

break;

case "bioprojectType":

str = "Bioproject\_Type";

break;

case "biosampleAccession":

str = "Biosample\_Accession";

break;

case "collectionDate":

str = "collection\_Date";

break;

case "creationDate":

str = "creation\_Date";

break;

case "experimentAccession":

str = "experiment\_Accession";

break;

case "experimentTitle":

str = "experiment\_Title";

break;

case "geographicLocation":

str = "geographic\_Location";

break;

case "librarySelection":

str = "library\_Selection";

break;

case "librarySource":

str = "library\_Source";

break;

case "libraryStrategy":

str = "library\_Strategy";

break;

case "organismName":

str = "organism\_Name";

break;

case "pubmedId":

str = "pubmed\_Id";

break;

case "sampleAccession":

str = "sample\_Accession";

break;

case "sampleName":

str = "sample\_Name";

break;

case "sampleNumber":

str = "sample\_Number";

break;

case "sampleSite":

str = "sample\_Site";

break;

case "sampleTitle":

str = "sample\_Title";

break;

case "sampleType":

str = "sample\_Type";

break;

case "sourceDatabase":

str = "source\_Database";

break;

case "studyAccession":

str = "study\_Accession";

break;

case "studyTitle":

str = "study\_Title";

break;

case "totalBases":

str = "total\_Bases";

break;

case "totalRuns":

str = "total\_Runs";

break;

case "totalSize":

str = "total\_Size(Mb)";

break;

case "totalSpots":

str = "total\_Spots";

break;

case "dnaextractionMethod":

str = "DNA\_Extraction\_Method";

break;

case "studyType":

str = "study\_Type";

break;

}

return str;

}

// @Override

// public List<Gmksz> getTree() {

// List<Gmksz> parents = gmkszMapper.selectByParentId(0);

// for(int i = 0;i < parents.size();i++){

// if(gmkszMapper.selectType1(parents.get(i).getGmksz()) > 0)

// parents.get(i).setSlots("{ icon: 'genus' }");

// else if(gmkszMapper.selectType2(parents.get(i).getGmksz()) > 0)

// parents.get(i).setSlots("{ icon: 'family' }");

// else if(gmkszMapper.selectType3(parents.get(i).getGmksz()) > 0)

// parents.get(i).setSlots("{ icon: 'order' }");

// else if(gmkszMapper.selectType4(parents.get(i).getGmksz()) > 0)

// parents.get(i).setSlots("{ icon: 'class' }");

// else

// parents.get(i).setSlots("{ icon: 'species' }");

// buildTree(parents.get(i));

// }

// return parents;

// }

//

// public void buildTree(Gmksz gmksz){

// List<Gmksz> tmp = gmkszMapper.selectByParentId(gmksz.getId());

// if(tmp.size() > 0){

// gmksz.setSons(tmp);

// for(int i = 0;i < tmp.size();i++){

// if(gmkszMapper.selectType1(tmp.get(i).getGmksz()) > 0)

// tmp.get(i).setSlots("{ icon: 'genus' }");

// else if(gmkszMapper.selectType2(tmp.get(i).getGmksz()) > 0)

// tmp.get(i).setSlots("{ icon: 'family' }");

// else if(gmkszMapper.selectType3(tmp.get(i).getGmksz()) > 0)

// tmp.get(i).setSlots("{ icon: 'order' }");

// else if(gmkszMapper.selectType4(tmp.get(i).getGmksz()) > 0)

// tmp.get(i).setSlots("{ icon: 'class' }");

// else

// tmp.get(i).setSlots("{ icon: 'species' }");

// buildTree(tmp.get(i));

// }

// }

// }

}

SearchService.java

package com.srt.service;

import com.srt.pojo.Animals;

import com.srt.pojo.Bioproject;

import java.util.List;

public interface SearchService {

List<Animals> getExhibitionInfo();

List<Animals> selectInfo(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth);

List<Animals> selectChangeInfo(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth);

// List<Animals> selectInfo(Animals animals);

// List<Animals> selectWebById(String id);

List<Animals> getAll(String species, String sampleSite1, String instrument, String libraryStrategy, String bioprojecttype);

List<Animals> getInfo();

List<Animals> getSpecies();

List<Animals> getSampleSite1();

List<Animals> getSampleSite2();

List<Animals> getInstrument();

List<Animals> getLibraryStrategy();

List<Bioproject> getBioProjectType();

List<Animals> selectInfoOrder(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth, String orderField, String order);

List<Animals> selectChangeInfoOrder(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth, String orderField, String order);

}

SeachServiceImpl.java

package com.srt.service.impl;

import com.srt.mapper.AnimalsMapper;

import com.srt.mapper.BioprojectMapper;

import com.srt.pojo.Animals;

import com.srt.pojo.AnimalsExample;

import com.srt.pojo.Bioproject;

import com.srt.service.SearchService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class SearchServiceImpl implements SearchService {

@Autowired

private AnimalsMapper animalsMapper;

@Autowired

private BioprojectMapper bioprojectMapper;

public List<Animals> getExhibitionInfo() {

List<Animals> result = animalsMapper.selectByExample(null);

return result;

}

public List<Animals> selectInfo(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth) {

List<Animals> result = animalsMapper.selectInfo(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth);

return result;

}

public List<Animals> selectChangeInfo(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth) {

List<Animals> result = animalsMapper.selectChangeInfo(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth);

return result;

}

// public List<Animals> selectInfo(Animals animals) {

// List<Animals> result = animalsMapper.selectInfo(animals);

// return result;

// }

// public List<Animals> selectWebById(String id) {

// List<Animals> str = animalsMapper.selectWebById(id);

// return str;

// }

public List<Animals> getAll(String species,String sampleSite1,String instrument,String libraryStrategy,String bioprojecttype) {

List<Animals> All = animalsMapper.selectAllInfo(species,sampleSite1,instrument,libraryStrategy,bioprojecttype);

return All;

}

public List<Animals> getInfo(){

List<Animals> Info = animalsMapper.selectByExample(null);

return Info;

}

public List<Animals> getSpecies() {

List<Animals> species = animalsMapper.selectSpecies(null);

return species;

}

public List<Animals> getSampleSite1() {

List<Animals> sampleSite1 = animalsMapper.getSampleSite1(null);

return sampleSite1;

}

public List<Animals> getSampleSite2() {

List<Animals> sampleSite2 = animalsMapper.getSampleSite2(null);

return sampleSite2;

}

public List<Animals> getInstrument() {

List<Animals> instrument = animalsMapper.getInstrument(null);

return instrument;

}

public List<Animals> getLibraryStrategy() {

List<Animals> libraryStrategy = animalsMapper.getLibraryStrategy(null);

return libraryStrategy;

}

public List<Bioproject> getBioProjectType(){

List<Bioproject> result = bioprojectMapper.selectByExample(null);

return result;

}

@Override

public List<Animals> selectInfoOrder(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth, String orderField, String order) {

AnimalsExample example = new AnimalsExample();

String cause = getMapping(orderField).equals("") ? orderField : getMapping(orderField);

if(order.equals("descend"))

cause += " DESC";

else if(order.equals("ascend"))

cause += " ASC";

example.setOrderByClause(cause);

List<Animals> Info = animalsMapper.selectByExample(example);

return Info;

}

@Override

public List<Animals> selectChangeInfoOrder(String species, String sampleSite1, String instrument, String libraryStrategy, String newStudyType, String newDna, String newSex, String startDate, String endDate, String startMonth, String endMonth, String orderField, String order) {

String cause = getMapping(orderField).equals("") ? orderField : getMapping(orderField);

if(order.equals("descend"))

cause += " DESC";

else if(order.equals("ascend"))

cause += " ASC";

List<Animals> result = animalsMapper.selectChangeInfo1(species,sampleSite1,instrument,libraryStrategy,newStudyType,newDna,newSex,startDate,endDate,startMonth,endMonth,getMapping(orderField).equals("") ? orderField : getMapping(orderField),cause);

return result;

}

public String getMapping(String source){

String str = "";

switch (source){

case "bioprojectAccession":

str = "Bioproject\_Accession";

break;

case "bioprojectDescription":

str = "Bioproject\_Description";

break;

case "bioprojectType":

str = "Bioproject\_Type";

break;

case "biosampleAccession":

str = "Biosample\_Accession";

break;

case "collectionDate":

str = "collection\_Date";

break;

case "creationDate":

str = "creation\_Date";

break;

case "experimentAccession":

str = "experiment\_Accession";

break;

case "experimentTitle":

str = "experiment\_Title";

break;

case "geographicLocation":

str = "geographic\_Location";

break;

case "librarySelection":

str = "library\_Selection";

break;

case "librarySource":

str = "library\_Source";

break;

case "libraryStrategy":

str = "library\_Strategy";

break;

case "organismName":

str = "organism\_Name";

break;

case "pubmedId":

str = "pubmed\_Id";

break;

case "sampleAccession":

str = "sample\_Accession";

break;

case "sampleName":

str = "sample\_Name";

break;

case "sampleNumber":

str = "sample\_Number";

break;

case "sampleSite":

str = "sample\_Site";

break;

case "sampleTitle":

str = "sample\_Title";

break;

case "sampleType":

str = "sample\_Type";

break;

case "sourceDatabase":

str = "source\_Database";

break;

case "studyAccession":

str = "study\_Accession";

break;

case "studyTitle":

str = "study\_Title";

break;

case "totalBases":

str = "total\_Bases";

break;

case "totalRuns":

str = "total\_Runs";

break;

case "totalSize":

str = "total\_Size(Mb)";

break;

case "totalSpots":

str = "total\_Spots";

break;

case "dnaextractionMethod":

str = "DNA\_Extraction\_Method";

break;

case "studyType":

str = "study\_Type";

break;

}

return str;

}

}

AnimalMapper.java

package com.srt.mapper;

import com.srt.pojo.Animals;

import com.srt.pojo.AnimalsExample;

import java.util.List;

import com.srt.pojo.SampleSiteTmp;

import com.srt.pojo.StudyTypeTmp;

import org.apache.ibatis.annotations.Param;

public interface AnimalsMapper {

long countByExample(AnimalsExample example);

int deleteByExample(AnimalsExample example);

int deleteByPrimaryKey(String experimentAccession);

int insert(Animals record);

int insertSelective(Animals record);

List<Animals> selectByExample(AnimalsExample example);

Animals selectByPrimaryKey(String experimentAccession);

int updateByExampleSelective(@Param("record") Animals record, @Param("example") AnimalsExample example);

int updateByExample(@Param("record") Animals record, @Param("example") AnimalsExample example);

int updateByPrimaryKeySelective(Animals record);

int updateByPrimaryKey(Animals record);

// 梁皓伟

// List<Animals> selectChangeInfo(@Param("species") String species,@Param("sampleSite1") String sampleSite1,@Param("instrument") String instrument,@Param("libraryStrategy") String libraryStrategy,@Param("bioprojecttype") String bioprojecttype);

List<Animals> selectAllInfo(@Param("species") String species,@Param("sampleSite1") String sampleSite1,@Param("instrument") String instrument,@Param("libraryStrategy") String libraryStrategy,@Param("bioprojecttype") String bioprojecttype);

// List<Animals> selectInfo(Animals record);

// List<Animals> selectWebById(String id);

List<Animals> selectSpecies(AnimalsExample example);

List<Animals> getSampleSite1(AnimalsExample example);

List<Animals> getSampleSite2(AnimalsExample example);

List<Animals> getInstrument(AnimalsExample example);

List<Animals> getLibraryStrategy(AnimalsExample example);

// 王奎翰

List<Animals> selectExample(AnimalsExample example);

List<Animals> selectBySpecies(String Species);

List<Animals> selectByGenus(String Genus);

List<Animals> selectByType(String FirstClo);

List<Animals> selectByGenusl();

List<Animals> selectByBioprojectTypeOne();

List<Animals> selectByBioprojectTypeTwo(String firstClo);

List<Animals> selectOnChangeInfo(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong,@Param("sampleSite0") String sampleSite0,@Param("sampleSite1") String sampleSite1);

List<Animals> selectByG(AnimalsExample example);

List<Animals> selectByM(String gang);

// List<Animals> selectAll(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong);

// List<Animals> getOutline();

List<Animals> selectAll(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong);

List<Animals> selectParts(@Param("i") int i, @Param("pageSize") int pageSize);

long countByView();

List<Animals> selectOnChangeInfoParts(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong, @Param("i") int i, @Param("pageSize") int pageSize);

List<Animals> selectInfo(@Param("species") String species, @Param("sampleSite1") String sampleSite1, @Param("instrument") String instrument, @Param("libraryStrategy") String libraryStrategy, @Param("newStudyType") String newStudyType, @Param("newDna") String newDna, @Param("newSex") String newSex, @Param("startDate") String startDate, @Param("endDate") String endDate, @Param("startMonth") String startMonth, @Param("endMonth") String endMonth);

List<Animals> selectChangeInfo1(@Param("species") String species, @Param("sampleSite1") String sampleSite1, @Param("instrument") String instrument, @Param("libraryStrategy") String libraryStrategy, @Param("newStudyType") String newStudyType, @Param("newDna") String newDna, @Param("newSex") String newSex, @Param("startDate") String startDate, @Param("endDate") String endDate, @Param("startMonth") String startMonth, @Param("endMonth") String endMonth, @Param("param") String param, @Param("cause") String cause);

List<Animals> selectChangeInfo(@Param("species") String species, @Param("sampleSite1") String sampleSite1, @Param("instrument") String instrument, @Param("libraryStrategy") String libraryStrategy, @Param("newStudyType") String newStudyType, @Param("newDna") String newDna, @Param("newSex") String newSex, @Param("startDate") String startDate, @Param("endDate") String endDate, @Param("startMonth") String startMonth, @Param("endMonth") String endMonth);

List<Animals> selectOnChangeInfoOrder(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong, @Param("sampleSite0") String sampleSite0, @Param("sampleSite1") String sampleSite1, @Param("param") String s, @Param("cause") String cause);

List<String> selectOptionsForSpecies(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong,@Param("sampleSite0") String sampleSite0,@Param("sampleSite1") String sampleSite1);

List<SampleSiteTmp> selectOptionsForSampleSite(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong,@Param("sampleSite0") String sampleSite0,@Param("sampleSite1") String sampleSite1);

List<StudyTypeTmp> selectOptionsForStudyType(@Param("species") String species,@Param("genus") String genus,@Param("bioprojecttype") String bioprojecttype,@Param("gang") String gang,@Param("mu") String mu,@Param("ke") String ke,@Param("shu") String shu,@Param("zhong") String zhong,@Param("sampleSite0") String sampleSite0,@Param("sampleSite1") String sampleSite1);

}

AnimalMapper.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.srt.mapper.AnimalsMapper">

<resultMap id="BaseResultMap" type="com.srt.pojo.Animals">

<id column="Experiment\_Accession" jdbcType="VARCHAR" property="experimentAccession" />

<result column="Experiment\_Title" jdbcType="VARCHAR" property="experimentTitle" />

<result column="Organism\_Name" jdbcType="VARCHAR" property="organismName" />

<result column="Instrument" jdbcType="VARCHAR" property="instrument" />

<result column="Submitter" jdbcType="VARCHAR" property="submitter" />

<result column="Study\_Accession" jdbcType="VARCHAR" property="studyAccession" />

<result column="Study\_Title" jdbcType="VARCHAR" property="studyTitle" />

<result column="Sample\_Accession" jdbcType="VARCHAR" property="sampleAccession" />

<result column="Sample\_Title" jdbcType="VARCHAR" property="sampleTitle" />

<result column="Total\_Size" jdbcType="VARCHAR" property="totalSize" />

<result column="Total\_RUNs" jdbcType="VARCHAR" property="totalRuns" />

<result column="Total\_Spots" jdbcType="VARCHAR" property="totalSpots" />

<result column="Total\_Bases" jdbcType="VARCHAR" property="totalBases" />

<result column="Sample\_Name" jdbcType="VARCHAR" property="sampleName" />

<result column="Library\_Strategy" jdbcType="VARCHAR" property="libraryStrategy" />

<result column="Library\_Source" jdbcType="VARCHAR" property="librarySource" />

<result column="Library\_Selection" jdbcType="VARCHAR" property="librarySelection" />

<result column="Bioproject\_Accession" jdbcType="VARCHAR" property="bioprojectAccession" />

<result column="BioSample\_Accession" jdbcType="VARCHAR" property="biosampleAccession" />

<result column="Sample\_site" jdbcType="VARCHAR" property="sampleSite" />

<result column="Sex" jdbcType="VARCHAR" property="sex" />

<result column="Age" jdbcType="VARCHAR" property="age" />

<result column="Creation\_date" jdbcType="VARCHAR" property="creationDate" />

<result column="Collection\_date" jdbcType="VARCHAR" property="collectionDate" />

<result column="Geographic\_location" jdbcType="VARCHAR" property="geographicLocation" />

<result column="Latitude" jdbcType="VARCHAR" property="latitude" />

<result column="Longitude" jdbcType="VARCHAR" property="longitude" />

<result column="PubMed\_ID" jdbcType="VARCHAR" property="pubmedId" />

<result column="Conditions" jdbcType="VARCHAR" property="conditions" />

<result column="Pheotype" jdbcType="VARCHAR" property="pheotype" />

<result column="Breed" jdbcType="VARCHAR" property="breed" />

<result column="Source\_database" jdbcType="VARCHAR" property="sourceDatabase" />

<result column="Bioproject\_description" jdbcType="VARCHAR" property="bioprojectDescription" />

<result column="Sample\_number" jdbcType="VARCHAR" property="sampleNumber" />

<result column="LianjieB" jdbcType="VARCHAR" property="lianjieb" />

<result column="Outline" jdbcType="VARCHAR" property="outline" />

<result column="Orders" jdbcType="VARCHAR" property="orders" />

<result column="Family" jdbcType="VARCHAR" property="family" />

<result column="Genus" jdbcType="VARCHAR" property="genus" />

<result column="Species" jdbcType="VARCHAR" property="species" />

<result column="LianjieP" jdbcType="VARCHAR" property="lianjiep" />

<result column="LianjieS" jdbcType="VARCHAR" property="lianjies" />

<result column="LianjieE" jdbcType="VARCHAR" property="lianjiee" />

<result column="Bioproject\_type" jdbcType="VARCHAR" property="bioprojectType" />

<result column="Sample\_type" jdbcType="VARCHAR" property="sampleType" />

<result column="Study\_type" jdbcType="VARCHAR" property="studyType" />

<result column="DNA\_extraction\_method" jdbcType="VARCHAR" property="DNAExtractionMethod" />

<result column="LianjieNew" jdbcType="VARCHAR" property="lianjienew" />

<result column="extra1" jdbcType="VARCHAR" property="extra1" />

<result column="extra2" jdbcType="VARCHAR" property="extra2" />

<!-- <result column="First\_clo" jdbcType="VARCHAR" property="firstClo" />-->

<!-- <result column="Second\_clo" jdbcType="VARCHAR" property="secondClo" />-->

</resultMap>

<resultMap id="BaseResultMapP" type="com.srt.pojo.Animals">

<result column="Bioproject\_Accession" jdbcType="VARCHAR" property="bioprojectAccession" />

<result column="Study\_Title" jdbcType="VARCHAR" property="studyTitle" />

<result column="Sample\_number" jdbcType="VARCHAR" property="sampleNumber" />

<result column="Library\_Strategy" jdbcType="VARCHAR" property="libraryStrategy" />

<result column="Bioproject\_description" jdbcType="VARCHAR" property="bioprojectDescription" />

<result column="Bioproject\_type" jdbcType="VARCHAR" property="bioprojectType" />

</resultMap>

<resultMap id="SampleSiteMap" type="com.srt.pojo.SampleSiteTmp">

<result column="Sample\_site0" jdbcType="VARCHAR" property="sampleSite0" />

<result column="Sample\_site1" jdbcType="VARCHAR" property="sampleSite1" />

</resultMap>

<resultMap id="StudyTypeMap" type="com.srt.pojo.StudyTypeTmp">

<result column="level1" jdbcType="VARCHAR" property="level1" />

<result column="level2" jdbcType="VARCHAR" property="level2" />

<result column="level3" jdbcType="VARCHAR" property="level3" />

</resultMap>

<sql id="Example\_Where\_Clause">

<where>

<foreach collection="oredCriteria" item="criteria" separator="or">

<if test="criteria.valid">

<trim prefix="(" prefixOverrides="and" suffix=")">

<foreach collection="criteria.criteria" item="criterion">

<choose>

<when test="criterion.noValue">

and ${criterion.condition}

</when>

<when test="criterion.singleValue">

and ${criterion.condition} #{criterion.value}

</when>

<when test="criterion.betweenValue">

and ${criterion.condition} #{criterion.value} and #{criterion.secondValue}

</when>

<when test="criterion.listValue">

and ${criterion.condition}

<foreach close=")" collection="criterion.value" item="listItem" open="(" separator=",">

#{listItem}

</foreach>

</when>

</choose>

</foreach>

</trim>

</if>

</foreach>

</where>

</sql>

<sql id="Update\_By\_Example\_Where\_Clause">

<where>

<foreach collection="example.oredCriteria" item="criteria" separator="or">

<if test="criteria.valid">

<trim prefix="(" prefixOverrides="and" suffix=")">

<foreach collection="criteria.criteria" item="criterion">

<choose>

<when test="criterion.noValue">

and ${criterion.condition}

</when>

<when test="criterion.singleValue">

and ${criterion.condition} #{criterion.value}

</when>

<when test="criterion.betweenValue">

and ${criterion.condition} #{criterion.value} and #{criterion.secondValue}

</when>

<when test="criterion.listValue">

and ${criterion.condition}

<foreach close=")" collection="criterion.value" item="listItem" open="(" separator=",">

#{listItem}

</foreach>

</when>

</choose>

</foreach>

</trim>

</if>

</foreach>

</where>

</sql>

<sql id="Base\_Column\_List">

Experiment\_Accession, Experiment\_Title, Organism\_Name, Instrument, Submitter, Study\_Accession,

Study\_Title, Sample\_Accession, Sample\_Title, Total\_Size, Total\_RUNs, Total\_Spots,

Total\_Bases, Sample\_Name, Library\_Strategy, Library\_Source, Library\_Selection, Bioproject\_Accession,

BioSample\_Accession, Sample\_site, Sex, Age, Creation\_date, Collection\_date,

Geographic\_location, Latitude, Longitude, PubMed\_ID, Conditions, Pheotype, Breed,

Source\_database, Bioproject\_description, Sample\_number, LianjieB, Outline, Orders,

Family, Genus, Species, LianjieP, LianjieS, LianjieE, Bioproject\_type, Sample\_type,

Study\_type, DNA\_extraction\_method, LianjieNew, extra1, extra2

</sql>

<!--lhw-->

<select id="selectSpecies" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Species

from animals GROUP BY Species ORDER BY Species

</select>

<select id="getSampleSite1" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Sample\_site

from animals GROUP BY Sample\_site ORDER BY Sample\_site

</select>

<select id="getSampleSite2" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Sample\_site2

from animals GROUP BY Sample\_site2 ORDER BY Sample\_site2

</select>

<select id="getInstrument" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Instrument

from animals GROUP BY Instrument ORDER BY Instrument

</select>

<select id="getLibraryStrategy" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Library\_Strategy

from animals GROUP BY Library\_Strategy ORDER BY Library\_Strategy

</select>

<select id="selectWebById" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select

lianjie

from

animals

where project\_id = #{id,jdbcType=VARCHAR}

</select>

<select id="selectInfo" parameterType="java.lang.String" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="sampleSite1 != null and sampleSite1 != ''">

AND Sample\_site = #{sampleSite1}

</if>

<if test="instrument != null and instrument != ''">

AND Instrument = #{instrument}

</if>

<if test="libraryStrategy != null and libraryStrategy != ''">

AND Library\_Strategy = #{libraryStrategy}

</if>

<if test="newStudyType != null and newStudyType != ''">

AND Bioproject\_type = #{newStudyType}

</if>

<if test="newDna != null and newDna != ''">

AND DNA\_extraction\_method = #{newDna}

</if>

<if test="newSex != null and newSex != ''">

AND Sex = #{newSex}

</if>

<if test="startDate != null and startDate != ''">

AND Creation\_date &gt;= #{startDate}

</if>

<if test="endDate != null and endDate != ''">

AND Creation\_date &lt;= #{endDate}

</if>

<if test="startMonth != null and startMonth != ''">

AND Age &gt;= #{startMonth}

</if>

<if test="endMonth != null and endMonth != ''">

AND Age &lt;= #{endMonth}

</if>

</where>

</select>

<select id="selectChangeInfo" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="sampleSite1 != null and sampleSite1 != ''">

AND Sample\_site = #{sampleSite1}

</if>

<if test="instrument != null and instrument != ''">

AND Instrument = #{instrument}

</if>

<if test="libraryStrategy != null and libraryStrategy != ''">

AND Library\_Strategy = #{libraryStrategy}

</if>

<if test="newStudyType != null and newStudyType != ''">

AND Bioproject\_type = #{newStudyType}

</if>

<if test="newDna != null and newDna != ''">

AND DNA\_extraction\_method = #{newDna}

</if>

<if test="newSex != null and newSex != ''">

AND Sex = #{newSex}

</if>

<if test="startDate != null and startDate != ''">

AND Creation\_date &gt;= #{startDate}

</if>

<if test="endDate != null and endDate != ''">

AND Creation\_date &lt;= #{endDate}

</if>

<if test="startMonth != null and startMonth != ''">

AND Age &gt;= #{startMonth}

</if>

<if test="endMonth != null and endMonth != ''">

AND Age &lt;= #{endMonth}

</if>

</where>

</select>

<select id="selectChangeInfo1" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="sampleSite1 != null and sampleSite1 != ''">

AND Sample\_site = #{sampleSite1}

</if>

<if test="instrument != null and instrument != ''">

AND Instrument = #{instrument}

</if>

<if test="libraryStrategy != null and libraryStrategy != ''">

AND Library\_Strategy = #{libraryStrategy}

</if>

<if test="newStudyType != null and newStudyType != ''">

AND Bioproject\_type = #{newStudyType}

</if>

<if test="newDna != null and newDna != ''">

AND DNA\_extraction\_method = #{newDna}

</if>

<if test="newSex != null and newSex != ''">

AND Sex = #{newSex}

</if>

<if test="startDate != null and startDate != ''">

AND Creation\_date &gt;= #{startDate}

</if>

<if test="endDate != null and endDate != ''">

AND Creation\_date &lt;= #{endDate}

</if>

<if test="startMonth != null and startMonth != ''">

AND Age &gt;= #{startMonth}

</if>

<if test="endMonth != null and endMonth != ''">

AND Age &lt;= #{endMonth}

</if>

<if test="param != null and param != ''">

AND ${param} IS NOT NULL AND ${param} &lt;&gt; '' AND ${param} &lt;&gt; 'NA'

</if>

</where>

<if test="cause != null and cause != ''">

ORDER BY ${cause}

</if>

</select>

<select id="selectAllInfo" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="sampleSite1 != null and sampleSite1 != ''">

AND Sample\_site = #{sampleSite1}

</if>

<if test="instrument != null and instrument != ''">

AND Instrument = #{instrument}

</if>

<if test="libraryStrategy != null and libraryStrategy != ''">

AND Library\_Strategy = #{libraryStrategy}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

</where>

</select>

<!--wkh-->

<select id="selectAll" resultType="com.srt.pojo.Animals" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

</where>

GROUP BY Bioproject\_Accession

order by Bioproject\_Accession

</select>

<select id="selectExample" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Experiment\_Accession,Bioproject\_Accession,Study\_Title,Bioproject\_description,Bioproject\_type,Sample\_number,Library\_Strategy,LianjieS,LianjieB

from animals GROUP BY Bioproject\_Accession ORDER BY Bioproject\_Accession

</select>

<select id="selectParts" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Experiment\_Accession,Bioproject\_Accession,Study\_Title,Bioproject\_description,Bioproject\_type,Sample\_number,Library\_Strategy,LianjieS,LianjieB

from animals GROUP BY Bioproject\_Accession ORDER BY Bioproject\_Accession limit ${i},${pageSize}

</select>

<select id="selectByG" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select distinct Outline

from animals where Outline is not null order by Outline

</select>

<select id="selectByM" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select distinct Orders

from animals

where Outline = #{Outline} and Orders is not null order by Orders

</select>

<select id="selectBySpecies" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Bioproject\_Accession,Study\_Title,Bioproject\_description,Bioproject\_type,Sample\_number,Library\_Strategy

from animals

where Species = #{Species}

GROUP BY Bioproject\_Accession

</select>

<select id="selectByGenus" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Bioproject\_Accession,Study\_Title,Bioproject\_description,Bioproject\_type,Sample\_number,Library\_Strategy

from animals

where Genus = #{Genus}

GROUP BY Bioproject\_Accession

order by Bioproject\_Accession

</select>

<select id="selectByGenusl" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select distinct trim(ifNull(Genus , 'Others')) as Genus from animals order by Genus

</select>

<select id="selectByBioprojectTypeOne" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select SUBSTRING\_INDEX(Bioproject\_type,'-',1) as First\_clo from animals group by First\_clo order by First\_clo

</select>

<select id="selectByBioprojectTypeTwo" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select SUBSTRING\_INDEX((SUBSTRING\_INDEX(Bioproject\_type,'-',2)),'-',-1) as Second\_clo from animals

where SUBSTRING\_INDEX(Bioproject\_type,'-',1) = #{FirstClo}

group by Second\_clo order by Second\_clo

</select>

<select id="selectByType" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select Bioproject\_Accession,Study\_Title,Bioproject\_description,Bioproject\_type,Sample\_number,Library\_Strategy

from animals

where SUBSTRING\_INDEX(Bioproject\_type,'-',1) = #{FirstClo}

GROUP BY Bioproject\_Accession

</select>

<select id="getOutline" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select DISTINCT

Outline,Orders,Family,Genus,Species

from animals

where Outline is not null

</select>

<select id="selectOnChangeInfoOrder" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMapP">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species == 'Wild animals'.toString()">

AND Species &lt;&gt; 'Bovine' AND Species &lt;&gt; 'Horse' AND Species &lt;&gt; 'Pig' AND Species &lt;&gt; 'Sheep'

</if>

<if test="species == 'Domestic animals'.toString()">

AND Species = 'Bovine' OR Species = 'Horse' OR Species = 'Pig' OR Species = 'Sheep'

</if>

<if test="species != null and species != '' and species != 'Wild animals'.toString() and species != 'Domestic animals'.toString()">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

<if test="sampleSite0 != null and sampleSite0 != '' and sampleSite0 != 'undefined'">

AND Study\_type = #{sampleSite0}

</if>

<if test="sampleSite1 != null and sampleSite1 != '' and sampleSite1 != 'undefined'">

AND Sample\_site = #{sampleSite1}

</if>

<if test="param != null and param != ''">

AND ${param} IS NOT NULL AND ${param} &lt;&gt; '' AND ${param} &lt;&gt; 'NA'

</if>

</where>

GROUP BY Sample\_Accession

<if test="cause != null and cause != ''">

ORDER BY ${cause}

</if>

</select>

<select id="selectOnChangeInfo" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMapP">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species == 'Wild animals'.toString()">

AND Species &lt;&gt; 'Bovine' AND Species &lt;&gt; 'Horse' AND Species &lt;&gt; 'Pig' AND Species &lt;&gt; 'Sheep'

</if>

<if test="species == 'Domestic animals'.toString()">

AND Species = 'Bovine' OR Species = 'Horse' OR Species = 'Pig' OR Species = 'Sheep'

</if>

<if test="species != null and species != '' and species != 'Wild animals'.toString() and species != 'Domestic animals'.toString()">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

<if test="sampleSite0 != null and sampleSite0 != '' and sampleSite0 != 'undefined'">

AND Study\_type = #{sampleSite0}

</if>

<if test="sampleSite1 != null and sampleSite1 != '' and sampleSite1 != 'undefined'">

AND Sample\_site = #{sampleSite1}

</if>

</where>

GROUP BY Sample\_Accession

order by Sample\_Accession

</select>

<select id="selectOptionsForSpecies" parameterType="com.srt.pojo.AnimalsExample" resultType="java.lang.String">

select distinct species

from animals

<where>

<if test="species == 'Wild animals'.toString()">

AND Species &lt;&gt; 'Bovine' AND Species &lt;&gt; 'Horse' AND Species &lt;&gt; 'Pig' AND Species &lt;&gt; 'Sheep'

</if>

<if test="species == 'Domestic animals'.toString()">

AND Species = 'Bovine' OR Species = 'Horse' OR Species = 'Pig' OR Species = 'Sheep'

</if>

<if test="species != null and species != '' and species != 'Wild animals'.toString() and species != 'Domestic animals'.toString()">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

<if test="sampleSite0 != null and sampleSite0 != '' and sampleSite0 != 'undefined'">

AND Study\_type = #{sampleSite0}

</if>

<if test="sampleSite1 != null and sampleSite1 != '' and sampleSite1 != 'undefined'">

AND Sample\_site = #{sampleSite1}

</if>

</where>

order by species

</select>

<select id="selectOptionsForSampleSite" parameterType="com.srt.pojo.AnimalsExample" resultMap="SampleSiteMap">

SELECT DISTINCT IFNULL(Study\_type,'NA') AS Sample\_site0,Sample\_site AS Sample\_site1

FROM animals

<where>

<if test="species == 'Wild animals'.toString()">

AND Species &lt;&gt; 'Bovine' AND Species &lt;&gt; 'Horse' AND Species &lt;&gt; 'Pig' AND Species &lt;&gt; 'Sheep'

</if>

<if test="species == 'Domestic animals'.toString()">

AND Species = 'Bovine' OR Species = 'Horse' OR Species = 'Pig' OR Species = 'Sheep'

</if>

<if test="species != null and species != '' and species != 'Wild animals'.toString() and species != 'Domestic animals'.toString()">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

<if test="sampleSite0 != null and sampleSite0 != '' and sampleSite0 != 'undefined'">

AND Study\_type = #{sampleSite0}

</if>

<if test="sampleSite1 != null and sampleSite1 != '' and sampleSite1 != 'undefined'">

AND Sample\_site = #{sampleSite1}

</if>

</where>

ORDER BY Sample\_site0,Sample\_site1

</select>

<select id="selectOptionsForStudyType" parameterType="com.srt.pojo.AnimalsExample" resultMap="StudyTypeMap">

SELECT DISTINCT

SUBSTRING\_INDEX(Bioproject\_type,'-',1) as level1,

SUBSTRING\_INDEX(SUBSTRING\_INDEX(Bioproject\_type,'-',2),'-',-1) as level2,

SUBSTRING\_INDEX(SUBSTRING\_INDEX(Bioproject\_type,'-',3),'-',-1) as level3

FROM animals

<where>

<if test="species == 'Wild animals'.toString()">

AND Species &lt;&gt; 'Bovine' AND Species &lt;&gt; 'Horse' AND Species &lt;&gt; 'Pig' AND Species &lt;&gt; 'Sheep'

</if>

<if test="species == 'Domestic animals'.toString()">

AND Species = 'Bovine' OR Species = 'Horse' OR Species = 'Pig' OR Species = 'Sheep'

</if>

<if test="species != null and species != '' and species != 'Wild animals'.toString() and species != 'Domestic animals'.toString()">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

<if test="sampleSite0 != null and sampleSite0 != '' and sampleSite0 != 'undefined'">

AND Study\_type = #{sampleSite0}

</if>

<if test="sampleSite1 != null and sampleSite1 != '' and sampleSite1 != 'undefined'">

AND Sample\_site = #{sampleSite1}

</if>

</where>

ORDER BY level1,level2,level3

</select>

<select id="selectOnChangeInfoParts" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMapP">

select

<include refid="Base\_Column\_List" />

from animals

<where>

<if test="species != null and species != ''">

AND Species = #{species}

</if>

<if test="genus != null and genus != ''">

AND Genus = #{genus}

</if>

<if test="bioprojecttype != null and bioprojecttype != ''">

AND Bioproject\_type like CONCAT('%', #{bioprojecttype}, '%')

</if>

<if test="gang != null and gang != '' and gang != 'undefined'">

AND Outline = #{gang}

</if>

<if test="mu != null and mu != '' and mu != 'undefined'">

AND Orders = #{mu}

</if>

<if test="ke != null and ke != '' and ke != 'undefined'">

AND Family = #{ke}

</if>

<if test="shu != null and shu != '' and shu != 'undefined'">

AND Genus = #{shu}

</if>

<if test="zhong != null and zhong != '' and zhong != 'undefined'">

AND Species = #{zhong}

</if>

</where>

GROUP BY Bioproject\_Accession

order by Bioproject\_Accession

limit ${i},${pageSize}

</select>

<select id="selectByExample" parameterType="com.srt.pojo.AnimalsExample" resultMap="BaseResultMap">

select

<if test="distinct">

distinct

</if>

<include refid="Base\_Column\_List" />

from animals

<if test="\_parameter != null">

<include refid="Example\_Where\_Clause" />

</if>

<if test="orderByClause != null">

order by ${orderByClause}

</if>

</select>

<select id="selectByPrimaryKey" parameterType="java.lang.String" resultMap="BaseResultMap">

select

<include refid="Base\_Column\_List" />

from animals

where Experiment\_Accession = #{experimentAccession,jdbcType=VARCHAR}

</select>

<delete id="deleteByPrimaryKey" parameterType="java.lang.String">

delete from animals

where Experiment\_Accession = #{experimentAccession,jdbcType=VARCHAR}

</delete>

<delete id="deleteByExample" parameterType="com.srt.pojo.AnimalsExample">

delete from animals

<if test="\_parameter != null">

<include refid="Example\_Where\_Clause" />

</if>

</delete>

<insert id="insert" parameterType="com.srt.pojo.Animals">

insert into animals (Experiment\_Accession, Experiment\_Title,

Organism\_Name, Instrument, Submitter,

Study\_Accession, Study\_Title, Sample\_Accession,

Sample\_Title, Total\_Size, Total\_RUNs,

Total\_Spots, Total\_Bases, Sample\_Name,

Library\_Strategy, Library\_Source, Library\_Selection,

Bioproject\_Accession, BioSample\_Accession,

Sample\_site1, Sample\_site2, Sex,

Age, Creation\_date, Collection\_date,

Geographic\_location, Latitude, Longitude,

PubMed\_ID, Conditions, Pheotype,

Breed, Source\_database, Bioproject\_description,

Sample\_number, LianjieB, Outline,

Orders, Family, Genus,

Species, LianjieP, LianjieS,

LianjieE, Bioproject\_type, Sample\_type

)

values (#{experimentAccession,jdbcType=VARCHAR}, #{experimentTitle,jdbcType=VARCHAR},

#{organismName,jdbcType=VARCHAR}, #{instrument,jdbcType=VARCHAR}, #{submitter,jdbcType=VARCHAR},

#{studyAccession,jdbcType=VARCHAR}, #{studyTitle,jdbcType=VARCHAR}, #{sampleAccession,jdbcType=VARCHAR},

#{sampleTitle,jdbcType=VARCHAR}, #{totalSize,jdbcType=VARCHAR}, #{totalRuns,jdbcType=VARCHAR},

#{totalSpots,jdbcType=VARCHAR}, #{totalBases,jdbcType=VARCHAR}, #{sampleName,jdbcType=VARCHAR},

#{libraryStrategy,jdbcType=VARCHAR}, #{librarySource,jdbcType=VARCHAR}, #{librarySelection,jdbcType=VARCHAR},

#{bioprojectAccession,jdbcType=VARCHAR}, #{biosampleAccession,jdbcType=VARCHAR},

#{sampleSite1,jdbcType=VARCHAR}, #{sampleSite2,jdbcType=VARCHAR}, #{sex,jdbcType=VARCHAR},

#{age,jdbcType=VARCHAR}, #{creationDate,jdbcType=VARCHAR}, #{collectionDate,jdbcType=VARCHAR},

#{geographicLocation,jdbcType=VARCHAR}, #{latitude,jdbcType=VARCHAR}, #{longitude,jdbcType=VARCHAR},

#{pubmedId,jdbcType=VARCHAR}, #{conditions,jdbcType=VARCHAR}, #{pheotype,jdbcType=VARCHAR},

#{breed,jdbcType=VARCHAR}, #{sourceDatabase,jdbcType=VARCHAR}, #{bioprojectDescription,jdbcType=VARCHAR},

#{sampleNumber,jdbcType=VARCHAR}, #{lianjieb,jdbcType=VARCHAR}, #{outline,jdbcType=VARCHAR},

#{orders,jdbcType=VARCHAR}, #{family,jdbcType=VARCHAR}, #{genus,jdbcType=VARCHAR},

#{species,jdbcType=VARCHAR}, #{lianjiep,jdbcType=VARCHAR}, #{lianjies,jdbcType=VARCHAR},

#{lianjiee,jdbcType=VARCHAR}, #{bioprojectType,jdbcType=VARCHAR}, #{sampleType,jdbcType=VARCHAR}

)

</insert>

<insert id="insertSelective" parameterType="com.srt.pojo.Animals">

insert into animals

<trim prefix="(" suffix=")" suffixOverrides=",">

<if test="experimentAccession != null">

Experiment\_Accession,

</if>

<if test="experimentTitle != null">

Experiment\_Title,

</if>

<if test="organismName != null">

Organism\_Name,

</if>

<if test="instrument != null">

Instrument,

</if>

<if test="submitter != null">

Submitter,

</if>

<if test="studyAccession != null">

Study\_Accession,

</if>

<if test="studyTitle != null">

Study\_Title,

</if>

<if test="sampleAccession != null">

Sample\_Accession,

</if>

<if test="sampleTitle != null">

Sample\_Title,

</if>

<if test="totalSize != null">

Total\_Size,

</if>

<if test="totalRuns != null">

Total\_RUNs,

</if>

<if test="totalSpots != null">

Total\_Spots,

</if>

<if test="totalBases != null">

Total\_Bases,

</if>

<if test="sampleName != null">

Sample\_Name,

</if>

<if test="libraryStrategy != null">

Library\_Strategy,

</if>

<if test="librarySource != null">

Library\_Source,

</if>

<if test="librarySelection != null">

Library\_Selection,

</if>

<if test="bioprojectAccession != null">

Bioproject\_Accession,

</if>

<if test="biosampleAccession != null">

BioSample\_Accession,

</if>

<if test="sampleSite1 != null">

Sample\_site1,

</if>

<if test="sampleSite2 != null">

Sample\_site2,

</if>

<if test="sex != null">

Sex,

</if>

<if test="age != null">

Age,

</if>

<if test="creationDate != null">

Creation\_date,

</if>

<if test="collectionDate != null">

Collection\_date,

</if>

<if test="geographicLocation != null">

Geographic\_location,

</if>

<if test="latitude != null">

Latitude,

</if>

<if test="longitude != null">

Longitude,

</if>

<if test="pubmedId != null">

PubMed\_ID,

</if>

<if test="conditions != null">

Conditions,

</if>

<if test="pheotype != null">

Pheotype,

</if>

<if test="breed != null">

Breed,

</if>

<if test="sourceDatabase != null">

Source\_database,

</if>

<if test="bioprojectDescription != null">

Bioproject\_description,

</if>

<if test="sampleNumber != null">

Sample\_number,

</if>

<if test="lianjieb != null">

LianjieB,

</if>

<if test="outline != null">

Outline,

</if>

<if test="orders != null">

Orders,

</if>

<if test="family != null">

Family,

</if>

<if test="genus != null">

Genus,

</if>

<if test="species != null">

Species,

</if>

<if test="lianjiep != null">

LianjieP,

</if>

<if test="lianjies != null">

LianjieS,

</if>

<if test="lianjiee != null">

LianjieE,

</if>

<if test="bioprojectType != null">

Bioproject\_type,

</if>

<if test="sampleType != null">

Sample\_type,

</if>

</trim>

<trim prefix="values (" suffix=")" suffixOverrides=",">

<if test="experimentAccession != null">

#{experimentAccession,jdbcType=VARCHAR},

</if>

<if test="experimentTitle != null">

#{experimentTitle,jdbcType=VARCHAR},

</if>

<if test="organismName != null">

#{organismName,jdbcType=VARCHAR},

</if>

<if test="instrument != null">

#{instrument,jdbcType=VARCHAR},

</if>

<if test="submitter != null">

#{submitter,jdbcType=VARCHAR},

</if>

<if test="studyAccession != null">

#{studyAccession,jdbcType=VARCHAR},

</if>

<if test="studyTitle != null">

#{studyTitle,jdbcType=VARCHAR},

</if>

<if test="sampleAccession != null">

#{sampleAccession,jdbcType=VARCHAR},

</if>

<if test="sampleTitle != null">

#{sampleTitle,jdbcType=VARCHAR},

</if>

<if test="totalSize != null">

#{totalSize,jdbcType=VARCHAR},

</if>

<if test="totalRuns != null">

#{totalRuns,jdbcType=VARCHAR},

</if>

<if test="totalSpots != null">

#{totalSpots,jdbcType=VARCHAR},

</if>

<if test="totalBases != null">

#{totalBases,jdbcType=VARCHAR},

</if>

<if test="sampleName != null">

#{sampleName,jdbcType=VARCHAR},

</if>

<if test="libraryStrategy != null">

#{libraryStrategy,jdbcType=VARCHAR},

</if>

<if test="librarySource != null">

#{librarySource,jdbcType=VARCHAR},

</if>

<if test="librarySelection != null">

#{librarySelection,jdbcType=VARCHAR},

</if>

<if test="bioprojectAccession != null">

#{bioprojectAccession,jdbcType=VARCHAR},

</if>

<if test="biosampleAccession != null">

#{biosampleAccession,jdbcType=VARCHAR},

</if>

<if test="sampleSite1 != null">

#{sampleSite1,jdbcType=VARCHAR},

</if>

<if test="sampleSite2 != null">

#{sampleSite2,jdbcType=VARCHAR},

</if>

<if test="sex != null">

#{sex,jdbcType=VARCHAR},

</if>

<if test="age != null">

#{age,jdbcType=VARCHAR},

</if>

<if test="creationDate != null">

#{creationDate,jdbcType=VARCHAR},

</if>

<if test="collectionDate != null">

#{collectionDate,jdbcType=VARCHAR},

</if>

<if test="geographicLocation != null">

#{geographicLocation,jdbcType=VARCHAR},

</if>

<if test="latitude != null">

#{latitude,jdbcType=VARCHAR},

</if>

<if test="longitude != null">

#{longitude,jdbcType=VARCHAR},

</if>

<if test="pubmedId != null">

#{pubmedId,jdbcType=VARCHAR},

</if>

<if test="conditions != null">

#{conditions,jdbcType=VARCHAR},

</if>

<if test="pheotype != null">

#{pheotype,jdbcType=VARCHAR},

</if>

<if test="breed != null">

#{breed,jdbcType=VARCHAR},

</if>

<if test="sourceDatabase != null">

#{sourceDatabase,jdbcType=VARCHAR},

</if>

<if test="bioprojectDescription != null">

#{bioprojectDescription,jdbcType=VARCHAR},

</if>

<if test="sampleNumber != null">

#{sampleNumber,jdbcType=VARCHAR},

</if>

<if test="lianjieb != null">

#{lianjieb,jdbcType=VARCHAR},

</if>

<if test="outline != null">

#{outline,jdbcType=VARCHAR},

</if>

<if test="orders != null">

#{orders,jdbcType=VARCHAR},

</if>

<if test="family != null">

#{family,jdbcType=VARCHAR},

</if>

<if test="genus != null">

#{genus,jdbcType=VARCHAR},

</if>

<if test="species != null">

#{species,jdbcType=VARCHAR},

</if>

<if test="lianjiep != null">

#{lianjiep,jdbcType=VARCHAR},

</if>

<if test="lianjies != null">

#{lianjies,jdbcType=VARCHAR},

</if>

<if test="lianjiee != null">

#{lianjiee,jdbcType=VARCHAR},

</if>

<if test="bioprojectType != null">

#{bioprojectType,jdbcType=VARCHAR},

</if>

<if test="sampleType != null">

#{sampleType,jdbcType=VARCHAR},

</if>

</trim>

</insert>

<select id="countByExample" parameterType="com.srt.pojo.AnimalsExample" resultType="java.lang.Long">

select count(\*) from animals

<if test="\_parameter != null">

<include refid="Example\_Where\_Clause" />

</if>

</select>

<select id="countByView" resultType="java.lang.Long">

select counts from view\_animals

</select>

<update id="updateByExampleSelective" parameterType="map">

update animals

<set>

<if test="record.experimentAccession != null">

Experiment\_Accession = #{record.experimentAccession,jdbcType=VARCHAR},

</if>

<if test="record.experimentTitle != null">

Experiment\_Title = #{record.experimentTitle,jdbcType=VARCHAR},

</if>

<if test="record.organismName != null">

Organism\_Name = #{record.organismName,jdbcType=VARCHAR},

</if>

<if test="record.instrument != null">

Instrument = #{record.instrument,jdbcType=VARCHAR},

</if>

<if test="record.submitter != null">

Submitter = #{record.submitter,jdbcType=VARCHAR},

</if>

<if test="record.studyAccession != null">

Study\_Accession = #{record.studyAccession,jdbcType=VARCHAR},

</if>

<if test="record.studyTitle != null">

Study\_Title = #{record.studyTitle,jdbcType=VARCHAR},

</if>

<if test="record.sampleAccession != null">

Sample\_Accession = #{record.sampleAccession,jdbcType=VARCHAR},

</if>

<if test="record.sampleTitle != null">

Sample\_Title = #{record.sampleTitle,jdbcType=VARCHAR},

</if>

<if test="record.totalSize != null">

Total\_Size = #{record.totalSize,jdbcType=VARCHAR},

</if>

<if test="record.totalRuns != null">

Total\_RUNs = #{record.totalRuns,jdbcType=VARCHAR},

</if>

<if test="record.totalSpots != null">

Total\_Spots = #{record.totalSpots,jdbcType=VARCHAR},

</if>

<if test="record.totalBases != null">

Total\_Bases = #{record.totalBases,jdbcType=VARCHAR},

</if>

<if test="record.sampleName != null">

Sample\_Name = #{record.sampleName,jdbcType=VARCHAR},

</if>

<if test="record.libraryStrategy != null">

Library\_Strategy = #{record.libraryStrategy,jdbcType=VARCHAR},

</if>

<if test="record.librarySource != null">

Library\_Source = #{record.librarySource,jdbcType=VARCHAR},

</if>

<if test="record.librarySelection != null">

Library\_Selection = #{record.librarySelection,jdbcType=VARCHAR},

</if>

<if test="record.bioprojectAccession != null">

Bioproject\_Accession = #{record.bioprojectAccession,jdbcType=VARCHAR},

</if>

<if test="record.biosampleAccession != null">

BioSample\_Accession = #{record.biosampleAccession,jdbcType=VARCHAR},

</if>

<if test="record.sampleSite1 != null">

Sample\_site1 = #{record.sampleSite1,jdbcType=VARCHAR},

</if>

<if test="record.sampleSite2 != null">

Sample\_site2 = #{record.sampleSite2,jdbcType=VARCHAR},

</if>

<if test="record.sex != null">

Sex = #{record.sex,jdbcType=VARCHAR},

</if>

<if test="record.age != null">

Age = #{record.age,jdbcType=VARCHAR},

</if>

<if test="record.creationDate != null">

Creation\_date = #{record.creationDate,jdbcType=VARCHAR},

</if>

<if test="record.collectionDate != null">

Collection\_date = #{record.collectionDate,jdbcType=VARCHAR},

</if>

<if test="record.geographicLocation != null">

Geographic\_location = #{record.geographicLocation,jdbcType=VARCHAR},

</if>

<if test="record.latitude != null">

Latitude = #{record.latitude,jdbcType=VARCHAR},

</if>

<if test="record.longitude != null">

Longitude = #{record.longitude,jdbcType=VARCHAR},

</if>

<if test="record.pubmedId != null">

PubMed\_ID = #{record.pubmedId,jdbcType=VARCHAR},

</if>

<if test="record.conditions != null">

Conditions = #{record.conditions,jdbcType=VARCHAR},

</if>

<if test="record.pheotype != null">

Pheotype = #{record.pheotype,jdbcType=VARCHAR},

</if>

<if test="record.breed != null">

Breed = #{record.breed,jdbcType=VARCHAR},

</if>

<if test="record.sourceDatabase != null">

Source\_database = #{record.sourceDatabase,jdbcType=VARCHAR},

</if>

<if test="record.bioprojectDescription != null">

Bioproject\_description = #{record.bioprojectDescription,jdbcType=VARCHAR},

</if>

<if test="record.sampleNumber != null">

Sample\_number = #{record.sampleNumber,jdbcType=VARCHAR},

</if>

<if test="record.lianjieb != null">

LianjieB = #{record.lianjieb,jdbcType=VARCHAR},

</if>

<if test="record.outline != null">

Outline = #{record.outline,jdbcType=VARCHAR},

</if>

<if test="record.orders != null">

Orders = #{record.orders,jdbcType=VARCHAR},

</if>

<if test="record.family != null">

Family = #{record.family,jdbcType=VARCHAR},

</if>

<if test="record.genus != null">

Genus = #{record.genus,jdbcType=VARCHAR},

</if>

<if test="record.species != null">

Species = #{record.species,jdbcType=VARCHAR},

</if>

<if test="record.lianjiep != null">

LianjieP = #{record.lianjiep,jdbcType=VARCHAR},

</if>

<if test="record.lianjies != null">

LianjieS = #{record.lianjies,jdbcType=VARCHAR},

</if>

<if test="record.lianjiee != null">

LianjieE = #{record.lianjiee,jdbcType=VARCHAR},

</if>

<if test="record.bioprojectType != null">

Bioproject\_type = #{record.bioprojectType,jdbcType=VARCHAR},

</if>

<if test="record.sampleType != null">

Sample\_type = #{record.sampleType,jdbcType=VARCHAR},

</if>

</set>

<if test="\_parameter != null">

<include refid="Update\_By\_Example\_Where\_Clause" />

</if>

</update>

<update id="updateByExample" parameterType="map">

update animals

set Experiment\_Accession = #{record.experimentAccession,jdbcType=VARCHAR},

Experiment\_Title = #{record.experimentTitle,jdbcType=VARCHAR},

Organism\_Name = #{record.organismName,jdbcType=VARCHAR},

Instrument = #{record.instrument,jdbcType=VARCHAR},

Submitter = #{record.submitter,jdbcType=VARCHAR},

Study\_Accession = #{record.studyAccession,jdbcType=VARCHAR},

Study\_Title = #{record.studyTitle,jdbcType=VARCHAR},

Sample\_Accession = #{record.sampleAccession,jdbcType=VARCHAR},

Sample\_Title = #{record.sampleTitle,jdbcType=VARCHAR},

Total\_Size = #{record.totalSize,jdbcType=VARCHAR},

Total\_RUNs = #{record.totalRuns,jdbcType=VARCHAR},

Total\_Spots = #{record.totalSpots,jdbcType=VARCHAR},

Total\_Bases = #{record.totalBases,jdbcType=VARCHAR},

Sample\_Name = #{record.sampleName,jdbcType=VARCHAR},

Library\_Strategy = #{record.libraryStrategy,jdbcType=VARCHAR},

Library\_Source = #{record.librarySource,jdbcType=VARCHAR},

Library\_Selection = #{record.librarySelection,jdbcType=VARCHAR},

Bioproject\_Accession = #{record.bioprojectAccession,jdbcType=VARCHAR},

BioSample\_Accession = #{record.biosampleAccession,jdbcType=VARCHAR},

Sample\_site1 = #{record.sampleSite1,jdbcType=VARCHAR},

Sample\_site2 = #{record.sampleSite2,jdbcType=VARCHAR},

Sex = #{record.sex,jdbcType=VARCHAR},

Age = #{record.age,jdbcType=VARCHAR},

Creation\_date = #{record.creationDate,jdbcType=VARCHAR},

Collection\_date = #{record.collectionDate,jdbcType=VARCHAR},

Geographic\_location = #{record.geographicLocation,jdbcType=VARCHAR},

Latitude = #{record.latitude,jdbcType=VARCHAR},

Longitude = #{record.longitude,jdbcType=VARCHAR},

PubMed\_ID = #{record.pubmedId,jdbcType=VARCHAR},

Conditions = #{record.conditions,jdbcType=VARCHAR},

Pheotype = #{record.pheotype,jdbcType=VARCHAR},

Breed = #{record.breed,jdbcType=VARCHAR},

Source\_database = #{record.sourceDatabase,jdbcType=VARCHAR},

Bioproject\_description = #{record.bioprojectDescription,jdbcType=VARCHAR},

Sample\_number = #{record.sampleNumber,jdbcType=VARCHAR},

LianjieB = #{record.lianjieb,jdbcType=VARCHAR},

Outline = #{record.outline,jdbcType=VARCHAR},

Orders = #{record.orders,jdbcType=VARCHAR},

Family = #{record.family,jdbcType=VARCHAR},

Genus = #{record.genus,jdbcType=VARCHAR},

Species = #{record.species,jdbcType=VARCHAR},

LianjieP = #{record.lianjiep,jdbcType=VARCHAR},

LianjieS = #{record.lianjies,jdbcType=VARCHAR},

LianjieE = #{record.lianjiee,jdbcType=VARCHAR},

Bioproject\_type = #{record.bioprojectType,jdbcType=VARCHAR},

Sample\_type = #{record.sampleType,jdbcType=VARCHAR}

<if test="\_parameter != null">

<include refid="Update\_By\_Example\_Where\_Clause" />

</if>

</update>

<update id="updateByPrimaryKeySelective" parameterType="com.srt.pojo.Animals">

update animals

<set>

<if test="experimentTitle != null">

Experiment\_Title = #{experimentTitle,jdbcType=VARCHAR},

</if>

<if test="organismName != null">

Organism\_Name = #{organismName,jdbcType=VARCHAR},

</if>

<if test="instrument != null">

Instrument = #{instrument,jdbcType=VARCHAR},

</if>

<if test="submitter != null">

Submitter = #{submitter,jdbcType=VARCHAR},

</if>

<if test="studyAccession != null">

Study\_Accession = #{studyAccession,jdbcType=VARCHAR},

</if>

<if test="studyTitle != null">

Study\_Title = #{studyTitle,jdbcType=VARCHAR},

</if>

<if test="sampleAccession != null">

Sample\_Accession = #{sampleAccession,jdbcType=VARCHAR},

</if>

<if test="sampleTitle != null">

Sample\_Title = #{sampleTitle,jdbcType=VARCHAR},

</if>

<if test="totalSize != null">

Total\_Size = #{totalSize,jdbcType=VARCHAR},

</if>

<if test="totalRuns != null">

Total\_RUNs = #{totalRuns,jdbcType=VARCHAR},

</if>

<if test="totalSpots != null">

Total\_Spots = #{totalSpots,jdbcType=VARCHAR},

</if>

<if test="totalBases != null">

Total\_Bases = #{totalBases,jdbcType=VARCHAR},

</if>

<if test="sampleName != null">

Sample\_Name = #{sampleName,jdbcType=VARCHAR},

</if>

<if test="libraryStrategy != null">

Library\_Strategy = #{libraryStrategy,jdbcType=VARCHAR},

</if>

<if test="librarySource != null">

Library\_Source = #{librarySource,jdbcType=VARCHAR},

</if>

<if test="librarySelection != null">

Library\_Selection = #{librarySelection,jdbcType=VARCHAR},

</if>

<if test="bioprojectAccession != null">

Bioproject\_Accession = #{bioprojectAccession,jdbcType=VARCHAR},

</if>

<if test="biosampleAccession != null">

BioSample\_Accession = #{biosampleAccession,jdbcType=VARCHAR},

</if>

<if test="sampleSite1 != null">

Sample\_site1 = #{sampleSite1,jdbcType=VARCHAR},

</if>

<if test="sampleSite2 != null">

Sample\_site2 = #{sampleSite2,jdbcType=VARCHAR},

</if>

<if test="sex != null">

Sex = #{sex,jdbcType=VARCHAR},

</if>

<if test="age != null">

Age = #{age,jdbcType=VARCHAR},

</if>

<if test="creationDate != null">

Creation\_date = #{creationDate,jdbcType=VARCHAR},

</if>

<if test="collectionDate != null">

Collection\_date = #{collectionDate,jdbcType=VARCHAR},

</if>

<if test="geographicLocation != null">

Geographic\_location = #{geographicLocation,jdbcType=VARCHAR},

</if>

<if test="latitude != null">

Latitude = #{latitude,jdbcType=VARCHAR},

</if>

<if test="longitude != null">

Longitude = #{longitude,jdbcType=VARCHAR},

</if>

<if test="pubmedId != null">

PubMed\_ID = #{pubmedId,jdbcType=VARCHAR},

</if>

<if test="conditions != null">

Conditions = #{conditions,jdbcType=VARCHAR},

</if>

<if test="pheotype != null">

Pheotype = #{pheotype,jdbcType=VARCHAR},

</if>

<if test="breed != null">

Breed = #{breed,jdbcType=VARCHAR},

</if>

<if test="sourceDatabase != null">

Source\_database = #{sourceDatabase,jdbcType=VARCHAR},

</if>

<if test="bioprojectDescription != null">

Bioproject\_description = #{bioprojectDescription,jdbcType=VARCHAR},

</if>

<if test="sampleNumber != null">

Sample\_number = #{sampleNumber,jdbcType=VARCHAR},

</if>

<if test="lianjieb != null">

LianjieB = #{lianjieb,jdbcType=VARCHAR},

</if>

<if test="outline != null">

Outline = #{outline,jdbcType=VARCHAR},

</if>

<if test="orders != null">

Orders = #{orders,jdbcType=VARCHAR},

</if>

<if test="family != null">

Family = #{family,jdbcType=VARCHAR},

</if>

<if test="genus != null">

Genus = #{genus,jdbcType=VARCHAR},

</if>

<if test="species != null">

Species = #{species,jdbcType=VARCHAR},

</if>

<if test="lianjiep != null">

LianjieP = #{lianjiep,jdbcType=VARCHAR},

</if>

<if test="lianjies != null">

LianjieS = #{lianjies,jdbcType=VARCHAR},

</if>

<if test="lianjiee != null">

LianjieE = #{lianjiee,jdbcType=VARCHAR},

</if>

<if test="bioprojectType != null">

Bioproject\_type = #{bioprojectType,jdbcType=VARCHAR},

</if>

<if test="sampleType != null">

Sample\_type = #{sampleType,jdbcType=VARCHAR},

</if>

</set>

where Experiment\_Accession = #{experimentAccession,jdbcType=VARCHAR}

</update>

<update id="updateByPrimaryKey" parameterType="com.srt.pojo.Animals">

update animals

set Experiment\_Title = #{experimentTitle,jdbcType=VARCHAR},

Organism\_Name = #{organismName,jdbcType=VARCHAR},

Instrument = #{instrument,jdbcType=VARCHAR},

Submitter = #{submitter,jdbcType=VARCHAR},

Study\_Accession = #{studyAccession,jdbcType=VARCHAR},

Study\_Title = #{studyTitle,jdbcType=VARCHAR},

Sample\_Accession = #{sampleAccession,jdbcType=VARCHAR},

Sample\_Title = #{sampleTitle,jdbcType=VARCHAR},

Total\_Size = #{totalSize,jdbcType=VARCHAR},

Total\_RUNs = #{totalRuns,jdbcType=VARCHAR},

Total\_Spots = #{totalSpots,jdbcType=VARCHAR},

Total\_Bases = #{totalBases,jdbcType=VARCHAR},

Sample\_Name = #{sampleName,jdbcType=VARCHAR},

Library\_Strategy = #{libraryStrategy,jdbcType=VARCHAR},

Library\_Source = #{librarySource,jdbcType=VARCHAR},

Library\_Selection = #{librarySelection,jdbcType=VARCHAR},

Bioproject\_Accession = #{bioprojectAccession,jdbcType=VARCHAR},

BioSample\_Accession = #{biosampleAccession,jdbcType=VARCHAR},

Sample\_site1 = #{sampleSite1,jdbcType=VARCHAR},

Sample\_site2 = #{sampleSite2,jdbcType=VARCHAR},

Sex = #{sex,jdbcType=VARCHAR},

Age = #{age,jdbcType=VARCHAR},

Creation\_date = #{creationDate,jdbcType=VARCHAR},

Collection\_date = #{collectionDate,jdbcType=VARCHAR},

Geographic\_location = #{geographicLocation,jdbcType=VARCHAR},

Latitude = #{latitude,jdbcType=VARCHAR},

Longitude = #{longitude,jdbcType=VARCHAR},

PubMed\_ID = #{pubmedId,jdbcType=VARCHAR},

Conditions = #{conditions,jdbcType=VARCHAR},

Pheotype = #{pheotype,jdbcType=VARCHAR},

Breed = #{breed,jdbcType=VARCHAR},

Source\_database = #{sourceDatabase,jdbcType=VARCHAR},

Bioproject\_description = #{bioprojectDescription,jdbcType=VARCHAR},

Sample\_number = #{sampleNumber,jdbcType=VARCHAR},

LianjieB = #{lianjieb,jdbcType=VARCHAR},

Outline = #{outline,jdbcType=VARCHAR},

Orders = #{orders,jdbcType=VARCHAR},

Family = #{family,jdbcType=VARCHAR},

Genus = #{genus,jdbcType=VARCHAR},

Species = #{species,jdbcType=VARCHAR},

LianjieP = #{lianjiep,jdbcType=VARCHAR},

LianjieS = #{lianjies,jdbcType=VARCHAR},

LianjieE = #{lianjiee,jdbcType=VARCHAR},

Bioproject\_type = #{bioprojectType,jdbcType=VARCHAR},

Sample\_type = #{sampleType,jdbcType=VARCHAR}

where Experiment\_Accession = #{experimentAccession,jdbcType=VARCHAR}

</update>

</mapper>