Курсовой проект: «Разработка интеграционной базы данных на основе парсинга веб-ресурсов»

1. Цель проекта

Целью проекта является приобретение практических навыков по сбору, обработке и интеграции разнородных данных из публичных веб-источников, проектированию реляционной базы данных и созданию простого веб-приложения для работы с этой базой.

2. Задачи проекта

1. Выбрать предметную область и найти 3 веб-сайта с каталогами товаров/услуг/объектов выбранной категории.
2. Разработать парсеры для каждого из сайтов, обеспечивающие сбор не менее 1000 уникальных позиций с каждого.
3. Спроектировать и реализовать единую реляционную базу данных (например, в PostgreSQL или MySQL), которая будет интегрировать данные из всех источников.
4. Провести процедуру очистки и объединения данных (дедупликацию) на основе разработанных критериев.
5. Реализовать минимальное веб-приложение (бэкенд + фронтенд), предоставляющее пользовательский интерфейс для поиска и просмотра данных из созданной базы.
6. Опубликовать проект на хостинге (например, Heroku, PythonAnywhere, Vercel) или предоставить контейнеризированную версию (Docker).

3. Варианты предметных областей (на выбор студента)

Студенту предлагается выбрать одну из следующих областей или предложить свою, согласовав с преподавателем.

Вариант 1: Электроника и гаджеты

- **Объекты:** Смартфоны, ноутбуки, наушники, умные часы.
- **Примеры сайтов:** citilink.ru, dns-shop.ru, eldorado.ru, e-katalog.ru.
- **Особенности:** Много технических характеристик, которые нужно унифицировать. Сложная дедупликация по моделям.

Вариант 2: Книги и литература

- **Объекты:** Книги (художественные, научные, учебники).
- **Примеры сайтов:** labirint.ru, chitai-gorod.ru, book24.ru, ozon.ru/books/.
- **Особенности:** Дедупликация по ISBN. Важные атрибуты: автор, издательство, год издания, жанр.

Вариант 3: Игровые видеоигры

- **Объекты:** Игры для PC, PlayStation, Xbox, Nintendo.
- **Примеры сайтов:** store.steampowered.com, epicgames.com, gog.com, xbox.com.
- **Особенности:** Разные платформы, жанры, даты выхода. Можно парсить рейтинги и отзывы.

Вариант 4: Недвижимость (аренда/продажа)

- **Объекты:** Квартиры, дома, коммерческая недвижимость.
- **Примеры сайтов:** cian.ru, avito.ru/nedvizhimost, domclick.ru.
- **Особенности:** Важны геоданные (район, город), площадь, количество комнат, цена. Высокая динамика изменений.

Вариант 5: Вакансии и работа

- **Объекты:** Вакансии от работодателей.
- **Примеры сайтов:** hh.ru, habr.com/career, superjob.ru, zarplata.ru.
- **Особенности:** Структура данных о компании, требованиях, зарплате. Можно анализировать рынок труда.

Вариант 6: Фильмы и сериалы

- **Объекты:** Фильмы, сериалы, мультфильмы.
- **Примеры сайтов:** kinopoisk.ru, imdb.com, themoviedb.org.
- **Особенности:** Богатая мета-информация: режиссеры, актеры, рейтинги, жанры, описание.

Вариант 7: СВОЯ ОБЛАСТЬ

4. Детализация этапов выполнения

Этап 1: Парсинг данных

- Выбрать 3 сайта в выбранной предметной области.
- Разработать парсеры на Python с использованием библиотек (requests, BeautifulSoup, Scrapy, Selenium).
- Собрать как минимум следующие данные для каждой позиции:
    - Наименование
    - Цена (если есть)
    - Ссылка на оригинальную страницу

    - Описание
    - **Ключевой атрибут для дедупликации** (например, ISBN для книг, модель для смартфонов, название + год выпуска для фильмов).
    - Дополнительные характеристики (в зависимости от области): бренд, технические specs, автор, жанр, дата выхода и т.д.
- Сохранить сырые данные в структурированном виде (JSON, CSV).

Этап 2: Проектирование и создание БД

- Проанализировать собранные данные с разных сайтов.
- Разработать ER-диаграмму единой базы данных. Примерная структура:
    - **Таблица** products: Уникальные товары после дедупликации.
        - Id (PK), canonical_name, description, brand, model, image_url, ... (универсальные атрибуты).
    - **Таблица** offers: Предложения с конкретных сайтов.
        - Id (PK), product_id(FK), website_name, price, url, date_parsed.
    - **Таблица** attributes: Для хранения различных характеристик (например, "Объем памяти", "Диагональ экрана").
        - Id (PK), product_id (FK), attribute_name, attribute_value.
    - *(Опционально) Таблицы для категорий, жанров и т.д.*
- Создать скрипт миграции базы данных (SQL-файл).
- Наполнить базу данными.

Этап 3: Очистка и дедупликация данных

- Написать скрипт (на Python или SQL) для поиска и объединения дубликатов из таблицы offers в таблицу products.
- **Критерии дедупликации** должны быть четко определены (например, для книг: normalize(isbn) == normalize(isbn_from_another_site), для смартфонов: fuzzy_match(model) AND brand == brand).
- Использовать методы точного совпадения (по кодам ISBN, артикулам) и, при необходимости, нечеткого сравнения (библиотеки fuzzywuzzy, rapidfuzz).

Этап 4: Разработка пользовательского интерфейса

- Реализовать бэкенд на выбранном фреймворке (например, Flask, Django, FastAPI).
- Реализовать базовые функции API:

    - GET /search?q=... - поиск по названию и описанию.
    - GET /product/<id> - получение детальной информации о товаре и всех предложениях на него.
- Реализовать простой фронтенд (можно на чистом HTML/CSS/JS или с использованием легких фреймворков like Bootstrap, Vue.js).
- **Обязательные элементы интерфейса:**
    - Строка поиска.
    - Список результатов поиска (название, картинка, минимальная цена).
    - Страница товара с деталями и списком всех предложений с ценами и ссылками на сайты-источники.

Этап 5: Публикация и документация

- Разместить код проекта на GitHub/GitLab.
- Подготовить документацию в README.md с описанием проекта, инструкцией по запуску и скриншотами интерфейса.
- Опубликовать работающее приложение на хостинге или предоставить docker-compose.yml для локального запуска.

---

5. Критерии оценки

- **Корректность и полнота данных (25%):** Собрано >1000 позиций с каждого сайта, данные качественные.
- **Качество проектирования БД (25%):** Логичная, нормализованная структура, продуманные связи, эффективные индексы.
- **Алгоритм дедупликации (20%):** Обоснованный выбор критериев, эффективная реализация, низкий уровень ложных срабатываний.
- **Функциональность и usability интерфейса (15%):** Поиск работает корректно, интерфейс интуитивно понятен.
- **Качество кода и документация (15%):** Чистый, читаемый код, наличие README.md, описание процесса запуска.