**OSRS Data Scraper Suite**

This repository contains scrapers for Old School RuneScape (OSRS) game data, pulling information from the official OSRS Wiki to create structured databases.

**Scraper Components**

The suite consists of two main scrapers:

1. **Item Scraper** (osrs\_item\_scraper.py): Collects item data, including combat stats, passive effects, special attacks, and item slots.
2. **Boss Scraper** (extract.py): Collects boss data, including combat stats, special mechanics, immunities, and boss forms/phases.

**Databases Generated**

**Item Databases**

The Item Scraper generates three SQLite databases:

1. **osrs\_all\_items.db**: Contains all items from the game.
2. **osrs\_combat\_items.db**: Contains only items with combat stats, special attacks, or passive effects.
3. **osrs\_tradeable\_items.db**: Contains only tradeable items.

**Item Database Schema**

All three item databases share the same schema:

sql

CREATE TABLE items (

id INTEGER PRIMARY KEY,

name TEXT,

has\_special\_attack BOOLEAN,

special\_attack\_text TEXT,

has\_passive\_effect BOOLEAN,

passive\_effect\_text TEXT,

has\_combat\_stats BOOLEAN,

is\_tradeable BOOLEAN,

slot TEXT,

combat\_stats TEXT, *-- JSON string*

raw\_html TEXT

)

**Combat Stats JSON Structure**

The combat\_stats column contains a JSON object with the following structure:

json

{

"attack\_bonuses": {

"stab": 0,

"slash": 0,

"crush": 0,

"magic": 0,

"ranged": 70

},

"defence\_bonuses": {

"stab": 0,

"slash": 0,

"crush": 0,

"magic": 0,

"ranged": 0

},

"other\_bonuses": {

"strength": 0,

"ranged strength": 20,

"magic damage": "+0%",

"prayer": 0

},

"combat\_styles": [

{

"name": "Accurate",

"attack\_type": "Standard",

"style": "Accurate",

"speed": "6 ticks (3.6s)",

"range": "10 tiles",

"experience": "",

"boost": "+3 Ranged"

},

{

"name": "Rapid",

"attack\_type": "Standard",

"style": "Rapid",

"speed": "5 ticks (3.0s)",

"range": "10 tiles",

"experience": "",

"boost": "None"

},

{

"name": "Longrange",

"attack\_type": "Standard",

"style": "Longrange",

"speed": "6 ticks (3.6s)",

"range": "10 tiles",

"experience": "",

"boost": "+3 Defence"

}

]

}

**Boss Database**

The Boss Scraper generates a single SQLite database:

**osrs\_bosses.db**: Contains boss information and their various forms/phases.

**Boss Database Schema**

The boss database has a more complex structure with two tables:

sql

*-- Main boss information*

CREATE TABLE bosses (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT UNIQUE,

raid\_group TEXT,

examine TEXT,

release\_date TEXT,

location TEXT,

slayer\_level INTEGER,

slayer\_xp INTEGER,

slayer\_category TEXT,

has\_multiple\_forms BOOLEAN DEFAULT 0,

raw\_html TEXT

)

*-- Different forms/phases of bosses*

CREATE TABLE boss\_forms (

id INTEGER PRIMARY KEY AUTOINCREMENT,

boss\_id INTEGER,

form\_name TEXT,

form\_order INTEGER,

combat\_level INTEGER,

hitpoints INTEGER,

max\_hit TEXT,

attack\_speed INTEGER,

attack\_style TEXT,

attack\_level INTEGER,

strength\_level INTEGER,

defence\_level INTEGER,

magic\_level INTEGER,

ranged\_level INTEGER,

aggressive\_attack\_bonus INTEGER,

aggressive\_strength\_bonus INTEGER,

aggressive\_magic\_bonus INTEGER,

aggressive\_magic\_strength\_bonus INTEGER,

aggressive\_ranged\_bonus INTEGER,

aggressive\_ranged\_strength\_bonus INTEGER,

defence\_stab INTEGER,

defence\_slash INTEGER,

defence\_crush INTEGER,

defence\_magic INTEGER,

elemental\_weakness\_type TEXT,

elemental\_weakness\_percent TEXT,

defence\_ranged\_light INTEGER,

defence\_ranged\_standard INTEGER,

defence\_ranged\_heavy INTEGER,

attribute TEXT,

xp\_bonus TEXT,

aggressive BOOLEAN,

poisonous BOOLEAN,

poison\_immunity BOOLEAN,

venom\_immunity BOOLEAN,

melee\_immunity BOOLEAN,

magic\_immunity BOOLEAN,

ranged\_immunity BOOLEAN,

cannon\_immunity BOOLEAN,

thrall\_immunity BOOLEAN,

special\_mechanics TEXT,

image\_url TEXT,

size INTEGER,

npc\_ids TEXT,

assigned\_by TEXT,

FOREIGN KEY (boss\_id) REFERENCES bosses(id),

UNIQUE(boss\_id, form\_name)

)

**Notes on JSON fields:**

* max\_hit: JSON structure containing different attack styles and their max hits (e.g., {"melee": 30, "ranged": 25})
* attack\_style: JSON array of attack styles (e.g., ["slash", "crush"])
* npc\_ids: JSON array of NPC IDs
* assigned\_by: JSON array of Slayer masters
* special\_mechanics: JSON array of special mechanics descriptions

**Sample Queries**

**Item Queries**

1. **Get all weapons with a specific slot:**

sql

SELECT name, slot FROM items

WHERE slot = 'mainhand' AND has\_combat\_stats = 1

ORDER BY name;

1. **Get all weapons with a special attack:**

sql

SELECT name, special\_attack\_text FROM items

WHERE has\_special\_attack = 1

ORDER BY name;

1. **Get all items with ranged strength bonus:**

sql

SELECT name, json\_extract(combat\_stats, '$.other\_bonuses."ranged strength"') as ranged\_str

FROM items

WHERE json\_extract(combat\_stats, '$.other\_bonuses."ranged strength"') > 0

ORDER BY ranged\_str DESC;

1. **Find items with specific combat style:**

sql

SELECT name

FROM items

WHERE combat\_stats LIKE '%"name": "Rapid"%'

ORDER BY name;

1. **Get all 2h weapons with attack speed information:**

sql

SELECT

name,

json\_extract(combat\_stats, '$.combat\_styles[0].speed') as speed

FROM items

WHERE slot = '2h'

AND json\_extract(combat\_stats, '$.combat\_styles[0].speed') IS NOT NULL

ORDER BY name;

**Boss Queries**

1. **Get all raid bosses:**

sql

SELECT name, raid\_group FROM bosses

WHERE raid\_group IS NOT NULL

ORDER BY raid\_group, name;

1. **Find bosses with multiple forms:**

sql

SELECT b.name, COUNT(bf.id) as form\_count

FROM bosses b

JOIN boss\_forms bf ON b.id = bf.boss\_id

GROUP BY b.name

HAVING form\_count > 1

ORDER BY form\_count DESC;

1. **Get bosses with immunity to a specific attack type:**

sql

SELECT b.name

FROM bosses b

JOIN boss\_forms bf ON b.id = bf.boss\_id

WHERE bf.magic\_immunity = 1 OR bf.ranged\_immunity = 1 OR bf.melee\_immunity = 1

GROUP BY b.name;

1. **Get the highest hitpoint bosses:**

sql

SELECT b.name, MAX(bf.hitpoints) as max\_hp

FROM bosses b

JOIN boss\_forms bf ON b.id = bf.boss\_id

WHERE bf.hitpoints IS NOT NULL

GROUP BY b.name

ORDER BY max\_hp DESC

LIMIT 10;

1. **Find bosses with specific attributes:**

sql

SELECT b.name, bf.attribute

FROM bosses b

JOIN boss\_forms bf ON b.id = bf.boss\_id

WHERE bf.attribute IN ('undead', 'demon', 'dragon')

ORDER BY bf.attribute, b.name;

**Data Fields Explained**

**Item Data Fields**

* **id**: Unique identifier for the item
* **name**: Item name
* **has\_special\_attack**: Boolean indicating if the item has a special attack
* **special\_attack\_text**: Description of the special attack
* **has\_passive\_effect**: Boolean indicating if the item has a passive effect
* **passive\_effect\_text**: Description of the passive effect
* **has\_combat\_stats**: Boolean indicating if the item has combat stats
* **is\_tradeable**: Boolean indicating if the item can be traded
* **slot**: Equipment slot (e.g., 'mainhand', '2h', 'body', 'head')
* **combat\_stats**: JSON object containing detailed combat stats
* **raw\_html**: Full HTML content of the wiki page

**Boss Data Fields**

**Main Boss Table**

* **id**: Unique identifier for the boss
* **name**: Boss name
* **raid\_group**: Raid identifier for grouped raid bosses
* **examine**: Examine text for the boss
* **release\_date**: Date the boss was released
* **location**: Where the boss can be found
* **slayer\_level**: Required Slayer level (if applicable)
* **slayer\_xp**: Slayer experience gained
* **slayer\_category**: Category for Slayer assignments
* **has\_multiple\_forms**: Boolean indicating if the boss has multiple forms/phases
* **raw\_html**: Full HTML content of the wiki page

**Boss Forms Table**

* **id**: Unique identifier for the form
* **boss\_id**: Foreign key to the bosses table
* **form\_name**: Name of this specific form/phase
* **form\_order**: Order of phases for multi-phase bosses

Combat Stats:

* **combat\_level**, **hitpoints**, **max\_hit**, **attack\_speed**, **attack\_style**
* **attack\_level**, **strength\_level**, **defence\_level**, **magic\_level**, **ranged\_level**
* Various aggressive and defensive bonuses

Immunities:

* **poison\_immunity**, **venom\_immunity**, **melee\_immunity**
* **magic\_immunity**, **ranged\_immunity**, **cannon\_immunity**, **thrall\_immunity**

Other:

* **special\_mechanics**: JSON array of special mechanics descriptions
* **image\_url**: URL to the boss image
* **size**: Size of the boss in tiles
* **npc\_ids**: JSON array of NPC IDs
* **assigned\_by**: JSON array of Slayer masters who assign this boss

**Implementation Notes**

**Item Scraper Features**

1. **Slot Detection**: Automatically detects equipment slots using multiple methods.
2. **Combat Stats Extraction**: Extracts comprehensive combat stats from tables.
3. **Combat Styles**: Captures all combat styles with accurate speed values.
4. **Passive Effects**: Identifies and extracts passive effect descriptions.
5. **Resume Capability**: Can resume scraping from a checkpoint if interrupted.

**Boss Scraper Features**

1. **Multiple Forms**: Handles bosses with multiple forms or phases.
2. **Critical Stats**: Prioritizes extraction of combat level and hitpoints.
3. **Special Mechanics**: Extracts detailed fight mechanics from wiki pages.
4. **Immunity Detection**: Identifies various immunities from text and tables.
5. **Raid Grouping**: Associates raid bosses with their respective raids.

**Usage**

**Item Scraper**

bash

python osrs\_item\_scraper.py

This will:

1. Initialize three databases
2. Test extraction on a set of known items
3. Process all items in the OSRS wiki
4. Save data to the three databases

**Boss Scraper**

bash

python extract.py

This will:

1. Initialize the boss database
2. Get a list of all bosses from the wiki
3. Process each boss and its various forms
4. Save data to the boss database

**Requirements**

* Python 3.7+
* Required packages:
  + requests
  + beautifulsoup4
  + sqlite3 (standard library)
  + json (standard library)
  + re (standard library)
  + os (standard library)
  + time (standard library)
  + logging (standard library)