# NBA - prvá konzultácia

### Analýza dát

Informácie o hráčovi z basketball-reference [https://www.basketball-reference.com/] budú pozostávať zo základných informácií o hráčovi zobrazené na obrázku tu:


V html tento element vyzerá nasledovne:

List informácií o hráčovi, ktoré budeme uchovávať:

- meno (text)
- výška (cm)
- váha (kg)
- pozícia (text)
- dominantná ruka (text)
- vek (v rokoch)
- rok narodenia (dátum)
- miesto narodenia (štát)
- stredná škola (text)
- college (text)
- draftovaný tímom (text)
- rok draftovania (rok)
- debut (dátum)
- dĺžka kariéry (v rokoch)
- sieň slávy (uvedený v roku pridania do siene)

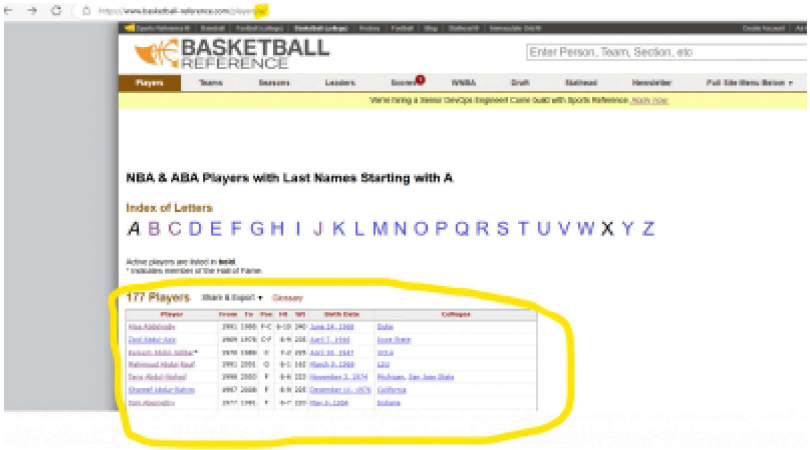
Ďalej budú pozostávať z jeho vlastností na ihrisku (body, doskoky, asistencie, bloky, obrátené lopty, fauly, percentuálna úspešnosť,…) zobrazené na basketball-reference [https://www.basketball-reference.com/] v takomto formáte:

Html záznam tohto elementu s vyznačenými vlastnosťami vyzerá nasledovne:

List vlastností o hráčovej hre (v rámci danej sezóny v priemere za hru), ktoré budeme uchovávať:

- sezóna (season)
- vek (age)
- pozícia (pos)
- tím (team)
- liga (league)
- hry odohrané (g)
- štartoval v hrách (gs)
- minúty (mpg)
- trafené koše (fgm)
- pokusy striel (fga)
- percentuálna úspešnosť trafených košov (fg%)
- trafené trojky (3pm)
- pokusy trojek (3pa)
- percentuálna úspešnosť trafených trojek (3p%)
- trafené dvojky (2pm)
- pokusy dvojek (2pa)
- percentuálna úspešnosť trafených dvojek (2p%)
- efektívna úspešnosť trafených košov (efg%)
- trafené pokutové koše (ftm)
- vystrelené pokutové koše (fta)
- percentuálna úspešnosť pokutových košov (ft%)
- ofenzívne doskoky (orb)
- defenzívne doskoky (drb)
- všeobecné doskoky (trb)
- asistencie (ast)
- ukradnutia lopty (stl)
- bloky (blk)
- obrátenia lopty (tov)
- fauly (pf)
- body (pts)

Linky všetkých hráčov sa budú zbierať veľmi jednoducho, keďže stránka ich triedi podľa začiatočného písmena v ich priezvisku, viď tu:




Podľa linku viem určiť, že ktoré začiatočné písmeno prezerám (napr. pre hráčov so začiatočným písmenom "D" v priezvisku je určená linka https://www.basketball-reference.com/players/ [https://www.basketball-reference.com/players/]**d**) a táto linka mi zobrazí všetky odkazy na daný list hráčov v tabuľke, čiže sa iba zoberie tabuľka hráčov a element po elemente sa budú zbierať linky hráčov. Takýmto spôsobom sa dajú vyzbierať všetci NBA hráči.

## Zdroje dát

- NBA štatistiky hráčov [https://www.basketball-reference.com/players/]
- Wikipedia [https://www.wikipedia.com]

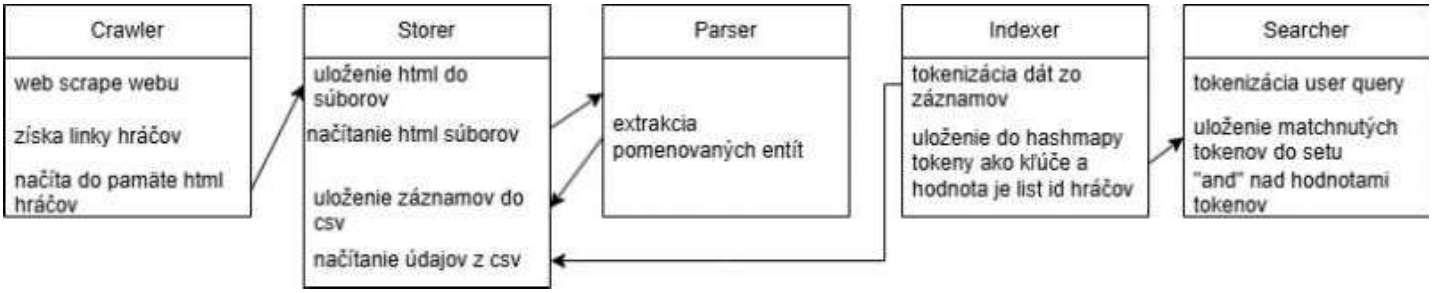
Poznamky: zoznam vlastnosti hraca, ukoncene

---

# NBA - druhá konzultácia

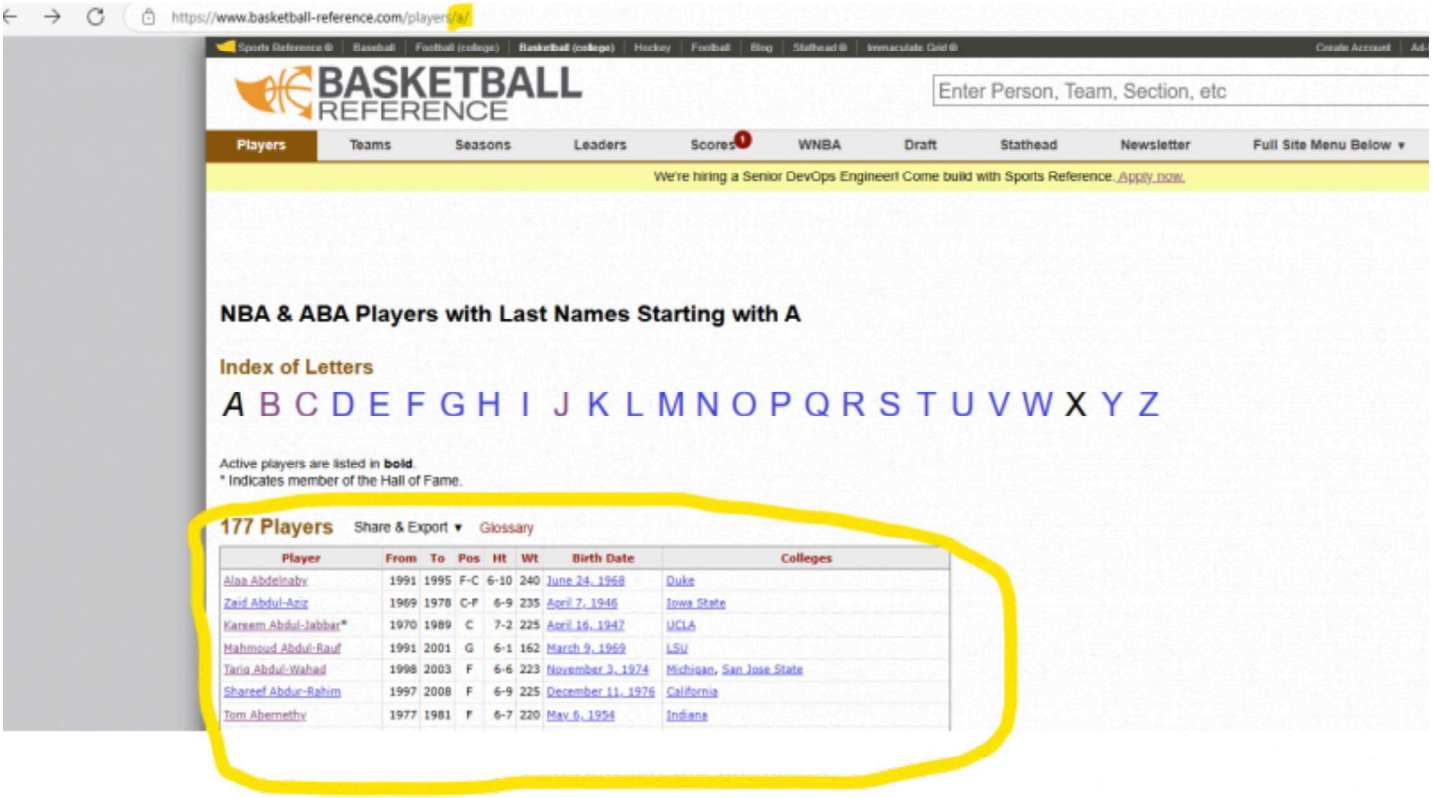
## Architektúra

Crawler získa linky na hráčov, následne posťahuje html hráčov z liniek a storer to uloží html do súborov. Tieto súbory sa načítajú v parser-i a ten vyextrahuje z nich elementy, ktoré sú zaujímavé. Tieto záznamy sa uložia do csv storer-om. Následne sa csv načítajú v indexer-i pomocou storer-a a jednotlivé záznamy sa začnú tokenizovať. Tieto tokeny sa uložia do hashmapy ako kľúče a vedú sa k nim listy id hráčov. V searcher-i sa zas táto hashmapa indexer-a zavolá, prebehne tokenizácia používateľského dopytu a jednotlivé kľúče z indexer-a sa match-ujú a ukladajú do výsledku. Nad uloženými id hráčov vo výsledkoch sa spraví prienik a searcher mi dá následné záznamy.




## Crawler

Web scraping hráčov zo stránky basketball-reference [https://www.basketball-reference.com/] je možné spraviť veľmi jednoducho, keďže ak sa použije linka basketball-reference/players/a [https://www.basketball-reference.com/players/a], tak sa zobrazia iba tí hráči, ktoré majú začiatočné písmeno “a” v priezvisku (čiže linka sa dá meniť /players/{písmeno} a budeme vedieť postupne získať všetky linky hráčov, keď použijeme všetky písmená v abecede). Linky hráčov sú zobrazené v tabuľke s ich menom a ďalšími dodatočnými informáciami:




### Získavanie linky hráča

Linky na hráčov budú tiež uložené do csv súbora, aby vždy nemusel prebiehať web scraping. Pseudokód na získavanie všetkých liniek na hráčov vyzerá nasledovne:

Download

```
Function get_players_links:
    Create a list "alphabet"
    Create empty list "player_page_links"
    For each "letter" in "alphabet":
        "url" to "https://www.basketball-reference.com/players/<letter>"
        "list_of_players_html" to the HTML content from making a GET request to "url"
        "pattern" regex matching player links based on HTML
        "player_links" finding all matches using "pattern" on "list_of_players_html"
        For each "player_link" in "player_links":
            Append "player_link" to "player_page_links"
            Store "player_link" to player links csv file
        Sleep for 3 seconds # comply with the website's crawler policy
    Return "player_page_links"
```

Pre získanie liniek pre hráčov by sa využili iba natívne knižnice Pythonu:

- "re" - regexy
- "time" - spánok na určitý čas
- "requests" - vykonávanie dopytov na stránky

### Získavanie informácií o hráčovi

Pseudokód na získavanie informácií o všetkých hráčoch vyzerá nasledovne:

Download

```
Function get_player_page_html_content(player_page_link: string):
    "player_page_html" is the HTML result of making a GET request to "player_page_link"
    Remove special characters from HTML in "player_page_html"
    Store "player_page_html" content into a file
    Return

Function get_pages_of_players(player_page_links: list[string]):
    For each "player_page_link" in "player_page_links":
        Call get_player_page_html_content function with "player_page_link"
        Sleep for 3 seconds # comply with the website's crawler policy
    Return
```

Pre získanie html všetkých hráčov by sa tiež využili iba natívne knižnice Pythonu:

- "os" - ukladanie súborov (os api)
- "re" - regexy
- "time" - spánok na určitý čas
- "requests" - vykonávanie dopytov na stránky

## Parser

Z jednotlivých html súboroch, ktoré majú informácie o danom hráčovi je potrebné vyparsovať pomocou regexov informácie, ktoré sme si určili v prvej konzultácií (konzultácia 1). Vyparsované informácie z html sa budú ukladať do csv súborov pre ľahkú manipuláciu s nimi neskôr. Html časti, ktoré budeme parsovať vyzerajú nasledovne:

```
...
<div id="meta">
    </div><!-- div.media-item --><div>
        <h1>
            <span>Kareem Abdul-Jabbar</span>
        </h1>
      <!-- all other pages -->
        <p><strong>Pronunciation</strong>: \kuh-REEM ab-dool juh-BAR\</p>
    <p>
      <strong><strong>Kareem Abdul-Jabbar</strong></strong>
    </p>
    <p><span class="desc">(Formerly known as Ferdinand Lewis Alcindor Jr.)</span></p>
    <p>(Lew, Cap, Murdock, Big Fella, The Tower of Power)</p>
    <p><strong>Position:</strong>Center•<strong>Shoots:</strong>Right</p>
    <p><span>7-2</span>, <span>225lb</span> (218cm, 102kg)</p>
    <p><strong>Born: </strong><span id="necro-birth" data-birth="1947-04-16"></span> <span><nobr>(Age: 77-167d)</nobr></span><span>
        in New York, <a href="/friv/birthplaces.fcgi?country=US&amp;state=NY">New York</a></span>
      <span class="f-i f-us" style="">us</span>
    </p>
    <p><strong>College:</strong><a href="/friv/colleges.fcgi?college=ucla">UCLA</a></p>
    <p><strong>High School:</strong>Power Memorial in New York, <a href="/friv/high_schools.fcgi?country=US&amp;state=NY">New York</a></p>
    <p>
        <strong>Draft:</strong>
        <a href="/teams/MIL/draft.html">Milwaukee Bucks</a>, 1st round (1st pick, 1st overall), <a href="/draft/NBA_1969.html">1969 NBA Draft</a>
    </p>
    <p><strong>NBA Debut: </strong><a href="/boxscores/196910180MIL.html">October 18, 1969</a></p>
    <p><strong>Hall of Fame:</strong> Inducted as Player in 1995 (<a href="/awards/hof.html">Full List</a>)</p>
    <p><strong>Career Length:</strong> 20 years</p>
</div>
...
```

```
<tbody>
  <tr id="per_game.1991" class="full_table" data-row="0">
    <th scope="row" class="left " data-stat="season"><a href="/players/a/abdelal01/gamelog/1991">1990-91</a></th>
    <td class="center " data-stat="age">22</td>
    <td class="left " data-stat="team_id"><a href="/teams/POR/1991.html">POR</a></td>
    <td class="left " data-stat="lg_id"><a href="/leagues/NBA_1991.html">NBA</a></td>
    <td class="center " data-stat="pos">PF</td><td class="right " data-stat="g">43</td>
    <td class="right iz" data-stat="gs">0</td><td class="right " data-stat="mp_per_g">6.7</td>
    <td class="right " data-stat="fg_per_g">1.3</td><td class="right " data-stat="fga_per_g">2.7</td>
    <td class="right " data-stat="fg_pct">.474</td><td class="right iz" data-stat="fg3_per_g">0.0</td>
    <td class="right iz" data-stat="fg3a_per_g">0.0</td><td class="right iz" data-stat="fg3_pct"></td>
    <td class="right " data-stat="fg2_per_g">1.3</td><td class="right " data-stat="fg2a_per_g">2.7</td>
    <td class="right " data-stat="fg2_pct">.474</td><td class="right " data-stat="efg_pct">.474</td>
    <td class="right " data-stat="ft_per_g">0.6</td><td class="right " data-stat="fta_per_g">1.0</td>
    <td class="right " data-stat="ft_pct">.568</td><td class="right " data-stat="orb_per_g">0.6</td>
    <td class="right " data-stat="drb_per_g">1.4</td><td class="right " data-stat="trb_per_g">2.1</td>
    <td class="right " data-stat="ast_per_g">0.3</td><td class="right " data-stat="stl_per_g">0.1</td>
    <td class="right " data-stat="blk_per_g">0.3</td><td class="right " data-stat="tov_per_g">0.5</td>
    <td class="right " data-stat="pf_per_g">0.9</td><td class="right " data-stat="pts_per_g">3.1</td>
    <td class="left iz" data-stat="award_summary"></td>
  </tr>
</tbody>
```

Pre parsovanie informácií o hráčovi je použitý nasledujúci pseudokód:

Download

```
Function parse_player_info_from_html(player_page_html: string):
    'name' to the result of pattern '<strong>\s*<strong>(.*?)</strong>\s*</strong>' in 'player_page_html'
    'height' to the result of pattern '\(\s*(\d+)cm' in 'player_page_html'
    'weight' to the result of  pattern ',\s*(\d+)kg' in 'player_page_html'
    'position' to the result of pattern in 'player_page_html'
    'shoots' to the result of pattern '(?i)Shoots:\s*</strong>\s*(right|left)' in 'player_page_html'
    'born' to the result of pattern 'data-birth="(\d{4})(?:-(\d{2}))?(?:-(\d{2}))?"' in 'player_page_html'
    'died' to the result of pattern 'data-death="(\d{4})(?:-(\d{2}))?(?:-(\d{2}))?"' in 'player_page_html'
    'born_in_city' to the result of pattern '/friv/birthplaces\.fcgi\?country=([^&]+)' in 'player_page_html'
    'high_school' to the result of pattern 'High School[s]?:\s*</strong>\s*([^<]+)\s*<' in 'player_page_html'
    'college' to the result of pattern 'College[s]?:\s*</strong>\s*<a\s*href=[\'"]([^\'"]*)[\'"]\s*>' in 'player_page_html'
    'nba_debut' to the result of pattern 'NBA Debut:\s*</strong>\s*<a href="[^"]*">(.*?)</a>' in 'player_page_html'
    'career_length' to the result of pattern '(?:Career Length|Experience):\s*</strong>\s*(\d+ years)' in 'player_page_html'
    'hall_of_fame_year' to the result of pattern 'Inducted as Player in (\d+)' in 'player_page_html'
    Store variables into a player info csv file
```

Pre parsovanie vlastností hráča v jednotlivých sezónach je použitý nasledujúci pseudokód:

Download

```
Function parse_player_stats_from_html(player_page_html: str):
    pattern as '<tr id="per_game\.\d+".*?>.*?</tr>' and store all occurences in a list "per_game_rows_html"
    For each "per_game_row_html" in "per_game_rows_html":
        "season" to the result of pattern '"season"\\s*>\\s*<\\s*a\\s*href="[^"]+"\\s*>\\s*([^<]+)\\s*<'
        "team" to the result of pattern '"team_id"\\s*>\\s*<\\s*a\\s*href="[^"]*"\\s*>\\s*([^<]*)\\s*<'
        "league" to the result of pattern '"lg_id"\s*>\s*<\s*a\s*href="[^"]*"\s*>\s*([^<]*)\s*<'
        "pos" to the resul of pattern '"pos"\s*>\s*([^<]*)\s*<'
        Store variables into player stats csv file
```

Toto parsovanie sa vykoná pre každé html hráča a tým sa získa v csv súboroch databáza informácií, s ktorou sa bude ďalej pracovať.

Pre parsovanie html súborov by sa tiež využili iba natívne knižnice Pythonu:

- "os" - ukladanie súborov (os api)
- "re" - regexy
- "datetime" - dátumové operácie

# Indexer

Terajšia verzia vlastného indexu sa robí iba nad atribútmi "name", "position", "shoots", "college", "high_school", "life_status", "season", "team" a "born_in_country". Ako index sa zadefinuje hash mapa a kľúče v nej sú jednotlivé tokeny (slová). Funguje to tak, že sa najprv prečítajú všetky záznamy z info o hráčoch csv súboru. Po načítaní sa ide záznam po zázname a tokenizujú sa jednotlivé atribúty (iba podľa medzery sa rozdelia slová do listu). Ak sa token nachádza v indexe, tak sa ako hodnota k tomuto kľúču pripíše do listu id hráča. Tým pádom keď vo vyhľadávači chcem hľadať podľa nejakého slova, tak sa mi skontroluje index pre dané slovo (skontroluje sa, či v indexe kľúč existuje) a keď sa dané slovo v indexe nájde, tak sa mi vráti list id-čiek všetkých hráčov, kde sa toto slovo nachádza. Následne sa to potom ešte zoradí podľa vypočítaného tf-idf. TF a IDF sa počíta za behu indexácie a následne zbehne ešte ďalší cyklus cez tokeny, ktorý vypočíta TF-IDF a zapíše do indexu ako hodnotu (doc id, tf-idf-value).

Dáta v csv vyzerajú takto:

| Id | Name | Height | Weight | Position | Shoots | Age | Born | Life_status | Born_in_country | High_school | College | Nba_draft_team | Nba_draft_year | Nba_debut | Career_length | Hall_of_fame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Jaylen Talnique A | 183cm | 102kg | Point Guard | Righty | 28yo | 1996-05-04 | alive | US | Mount St. Joseph | stbonny | | | 2018-10-17 | 3 years | |
| 1 | Paul Carlyle Arm | 180cm | 77kg | Guard | Righty | 64yo | 1918-11-01 | deceased | US | Central in Fort W | indiana | | | 1948-11-03 | 3 years | |
| 2 | Mark Richard Ac | 211cm | 99kg | Center | Righty | 61yo | 1962-11-15 | alive | US | Palos Verdes in F | oraltob | Dallas Mavericks | 1985 | 1987-11-06 | 6 years | 2010 |
| 3 | Chester J. Aubuc | 178cm | 62kg | Guard | Righty | 88yo | 1916-05-18 | deceased | US | Horace Mann in | michiganst | | | 1946-11-02 | 1 year | |
| 4 | James Randall A | 203cm | 99kg | Shooting Gu | Righty | 59yo | 1965-01-26 | alive | US | Milton in Milton | floridast | | | 1989-03-09 | 2 years | |
| 5 | Andrew Emil An | 188cm | 83kg | Shooting Gu | Righty | 73yo | 1945-07-06 | deceased | US | Marywale in Che | canisius | Boston Celtics | 1967 | | 3 years | |
| 6 | Luke Robert Bab | 206cm | 102kg | Small Forwa | Lefty | 35yo | 1989-06-20 | alive | US | Galena in Reno | nevada | Minnesota Timbe | 2010 | 2010-11-01 | 8 years | |
| 7 | Kyle F. Anderson | 206cm | 104kg | Small Forwa | Righty | 31yo | 1993-09-20 | alive | US | St. Anthony in Je | ucla | San Antonio Spur | 2014 | 2014-11-06 | 10 years | |
| 8 | LaMark Anthony | 185cm | 79kg | Point Guard | Righty | 54yo | 1969-11-11 | alive | US | Paul Laurence Di | ohiost | | | 1999-02-06 | 1 year | |
| 9 | Thomas Craig Al | 201cm | 99kg | Small Forwa | Righty | 70yo | 1954-05-06 | alive | US | Saint Joseph in S | indiana | Los Angeles Lake | 1976 | 1976-11-19 | 5 years | |
| 10 | Cole Hinton Ant | 188cm | 83kg | Point Guard | Righty | 24yo | 2000-05-15 | alive | US | Archbishop Moll | unc | Orlando Magic | 2020 | 2020-12-23 | 4 years | |
| 11 | Rafael Addison | 201cm | 97kg | Small Forwa | Righty | 60yo | 1964-07-22 | alive | US | Snyder in Jersey | syracuse | Phoenix Suns | 1986 | 1986-11-01 | 6 years | |
| 12 | Brandon Simone | 196cm | 85kg | Shooting Gu | Lefty | 44yo | 1980-06-16 | alive | US | Vallejo in Vallejo | pepperdine | Houston Rockets | 2001 | 2001-11-02 | 3 years | |
| 13 | Alvin Austin Att | 183cm | 79kg | Point Guard | Righty | 87yo | 1936-11-07 | deceased | US | Weequahic in Ne | ncat | Philadelphia Warr | 1960 | 1960-10-28 | 11 years | |

Pseudokód pre tento indexer vyzerá nasledovne:

Download

```
function build_player_info_index():
    "index" as hashmap containing key as token to value represented by list of player ids
    "tf" list for storage of term frequencies
    "idf" hashmap to store idf
    "players_doc_tokens" hashmap that stores all the unique tokens of each document
    open player info csv file
        read all lines of the csv and put into a list
        for each "row" in "rows":
            extract "player_id" from row's id column
            for each "player_property" in "PROPERTIES_IN_INDEX":
                if "player_property" in "row" is not empty:
                    "tokens" is set from splitting "player_property" in "row" by space and lowered
                    for each "token" in "tokens":
                        get the value from the "index" hashmap using "token" as key and append "player_id" to value
            calculate term frequency and store in "tf"
            calculate "idf" after aquiring all the tokens for current doc
        for each "row" in "rows":
            for each "token" in "players_doc_tokens"["doc id"]:
                calculate "tf-idf" value
                store the value in index for this "token"
    store "index" in csv file
```

Pre indexovanie by sa tiež využili iba natívne knižnice Pythonu:

- “collections” - pre hashmapu reprezentujúcu index
- “csv” - čítanie csv súboru

## Searcher

Vlastný vyhľadávač je tiež veľmi jednoduchý. Rozdelí vyhľadávací dotaz na tokeny (čiže slová, teda rozdelí string podľa medzier do listu) a tieto tokeny hľadá v indexe (do indexu je poskytnutý token ako kľúč). Ak sa tento kľúč nachádza v indexe, tak sa uloží do výsledkov. Následne sa spraví prienik nad všetkými výsledkami a ten sa uloží do premennej. Teraz prejde index a hľadá podľa hodnôt z výsledkov a zapisuje si čo najvyššie tf-idf s doc id do listu s neusporiadanými výsledkami. Keď prejde index, tak neusporiadané výsledky usporiada podľa tf-idf a vráti výsledky usporiadané podľa tf-idf zostupne.

Download

```
function search(index, query):
    "query_tokens" is provided by tokenizing "query"
    "results" is initialized as a list
    for each "token" in "query_tokens":
        if "token" exists in "index":
            add "index"["token"] to results
    if "results" is not empty:
        "unique_results" is intersection of all values in results
        gets "unsorted_results" by calculating score for each item in "unique_results"
        returns "sorted_results" that are sorted by highest score from "unsorted_results"
    else:
        return empty results
```

Pre vyhľadávač by sa tiež využili iba natívne knižnice Pythonu:

- “re” - regexy

Poznamky: crawler 5000 +/-, prerobit reprezentaciu zaznamu/dokumentu, index na disku, odstrániť duplikáty v teams, skúsiť druhý index pre unikátne tímy v season_team, term id miesto termov

---

vi2024

# NBA - odovzdanie 1

## Crawler & Parser

Dáta po použití crawlera:


Dáta po použití parsera:

| Name | Height | Weight | Position | Shoots | Age | Born | Deceased | Born In | High_school | College | Nba_draft_team | Nba_draft_year | Nba_debut | Career_length | Hall_of_fame | Seasons | Teams | Season_team |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Malcolm | 206 | 112 | Center | Right | 54 | 1970-1 | False | US | Brainerd In Cl | gatech | Phoenix Suns | 1993 | 1994-01-13 | 1 year | | 1967 | SDR | 1967_SDR |
| Sam Mac | 201 | 99 | Small For | Right | 54 | 1970-1 | False | US | Thornridge In | iowast | | | 1992-11-07 | 7 years | | 1956 | SYR | 1956_SYR |
| Donald Ja | 208 | 106 | Small For | Right | 54 | 1970-1 | False | US | Simi Valley In | ucla | Detroit Pistons | 1992 | 1992-11-06 | 9 years | | 1946 1947 19 | PIT STB BLB | 1946_PIT 1947_STE |
| Richard A | 211 | 114 | Power For | Right | 54 | 1970-1 | False | US | Center In Ant | syracuse | Atlanta Hawks | 1993 | 1996-01-27 | 2 years | | 2012 2013 20 | TOR TOR SA | 2012_TOR 2013_TO |
| Robert W | 213 | 113 | Center | Right | 54 | 1969-1 | False | US | Apple Valley I | minnesot | | | 1993-11-07 | 2 years | | 1953 | NYK | 1953_NYK |
| Dwa Lee | 165 | 78 | Point Guar | Right | 54 | 1970-1 | False | US | Will Rogers H | arkansas | Milwaukee Bucks | 1992 | 1992-11-06 | 7 years | | 1976 1977 19 | LAL LAL GSV | 1976_LAL 1977_LA |
| Oliver J. N | 206 | 127 | Center | Right | 54 | 1970-1 | False | US | Southwest In | arkansas | Phoenix Suns | 1992 | 1992-11-07 | 9 years | | 2006 2008 | NJN TOR | 2006_NJN 2008_TO |
| Christoph | 198 | 97 | Small For | Right | 54 | 1970-1 | False | US | Fairfax In Los | kentucky | Cleveland Cavalie | 1993 | 1993-11-05 | 10 years | | 2016 2017 20 | OKC OKC OK | 2016_OKC 2017_O |
| Alonzo H | 208 | 108 | Center | Right | 54 | 1970-1 | False | US | Indian River H | georgetc | Charlotte Hornets | 1992 | 1992-11-13 | 15 years | 2014 | 1987 1988 19 | BOS BOS OK | 1987_BOS 1988_BO |
| Todd Mac | 213 | 113 | Center | Right | 54 | 1970-1 | False | US | Central Merry | memphis | | | 1995-11-04 | 1 year | | 1990 1991 19 | POR POR MI | 1990_POR 1991_PO |
| Terrah Jer | 208 | 107 | Power For | Right | 54 | 1969-1 | False | US | DeMatha Cat | marylanc | New York Knicks | 1990 | 1990-11-02 | 4 years | | 2020 2021 20 | MIA TOR TO | 2020_MIA 2021_TO |
| Anthony | 193 | 94 | Shooting ( | Left | 54 | 1969-1 | False | US | Paseo In Kans | missouri | Los Angeles Lake | 1992 | 1992-11-06 | 13 years | | 1990 1991 19 | DEN DEN DE | 1990_DEN 1991_DE |
| Eric Todd | 201 | 97 | Shooting ( | Right | 54 | 1970-1 | False | US | Stevens In Ra | nebraska | Indiana Pacers | 1994 | 1994-11-04 | 14 years | | 1968 1968 19 | CIN MIL MIL | 1968_CIN 1968_MI |
| David Wil | 201 | 99 | Small For | Right | 54 | 1962-1 | True | US | Menchville In | norfolkst | Utah Jazz | 1984 | 1984-12-09 | 2 years | | 1996 1997 19 | VAN VAN VA | 1996_VAN 1997_VA |
| Raymond | 193 | 92 | Small For | Right | 54 | 1928-1 | True | US | Washington I | nwestern | Rochester Royals | 1951 | 1951-11-01 | 1 year | | 2005 2008 20 | DET DET LAC | 2005_DET 2008_DE |
| Richard Ju | 201 | 97 | Power For | Right | 54 | 1933-1 | True | US | Pottstown In | duquesn | St. Louis Hawks | 1955 | 1955-11-05 | 3 years | | 1997 1998 19 | SAC SAC OR | 1997_SAC 1998_SA |

## Indexovanie (TF-IDF)

Termy sú poukladané podľa ich **id** a **value**:

```
   1  id;term
3190  3188;greenville
3191  3189;greenway
3192  3190;greenwood
3193  3191;greer
3194  3192;greg
3195  3193;gregg
3196  3194;gregor
3197  3195;gregory
3198  3196;greig
3199  3197;greivis
3200  3198;grekin
3201  3199;grenadines
3202  3200;gresham
3203  3201;grevey
3204  3202;grey
3205  3203;griffin
3206  3204;griffith
3207  3205;grigsby
3208  3206;grimes
3209  3207;grimm
3210  3208;grimshaw
3211  3209;grimsley
3212  3210;grizzlies
3213  3211;groat
```

Pre **TF** som vyskúšal dva prístupy: prirodzený a logaritmický.

Prirodzený vyzerá nasledovne:

Download

```
...
{
  'joe': 0.037, 'depre': 0.037, 'shooting': 0.037, 'guard': 0.037, 'righty': 0.037,
  'stjohns': 0.037, 'westbury': 0.074, 'in': 0.037, 'alive': 0.037, '1970': 0.037,
  '1971': 0.037, '1972': 0.037, 'new': 0.11, 'york': 0.11, 'americans': 0.11,
  'united': 0.037, 'states': 0.037
},
...
```

Príklad pre logaritmický:

Download

```
...
{
  'mike': -2.89, 'theodore': -2.89, 'glenn': -2.89, 'shooting': -2.89, 'guard': -2.89,
  'righty': -2.89, 'sillinois': -2.89, 'coosa': -2.89, 'in': -2.89, 'rome': -2.89,
  'alive': -2.89, '1977': -2.89, '1978': -2.89, '1979': -2.89, '1980': -2.89,
  '1981': -2.89, '1982': -2.89, '1983': -2.89, '1984': -2.89, '1985': -2.89,
  '1986': -2.89, 'buffalo': -2.89, 'braves': -2.89, 'new': -1.79, 'york': -1.79,
  'knicks': -1.79, 'atlanta': -1.50, 'hawks': -1.50, 'milwaukee': -2.19,
  'bucks': -2.19, 'united': -2.89, 'states': -2.89
},
...
```

Pre **DF** som vyskúšal tiež dva prístupy: IDF a probabilistic IDF.

Príklad pre IDF:

Download

```
...
'mike': {
         1: 0.032, 24: 0.032, 37: 0.032, 128: 0.064,...
       },
...
```

Príklad pre probabilistic IDF vyzerá približne rovnako ako IDF len hodnoty sa menia:

Download

```
...
'mike': {
        1: 0.02, 24: 0.02, 37: 0.02, 128: 0.061,...
      },
...
```

Vypočítavam **TF-IDF** klasicky jednoduchým násobením:

Download

```
...
  # calculate tf_idf value for each token
  tf_idf[token] = tf[player_id].get(token, 0) * idf[token]
```

Následne to aj **sort**ujem, keď TF-IDF vraciam z funkcie:

Download

```
...
  # sorting the terms in the dict before returning
  return dict(sorted(tf_idf.items()))
...
```

Skóre následne počítam **nasčítavaním**:

Download

```
...
  # go through terms in query
  for term in query_tokens:
    # if term is in terms dict
    if term in term_dict:
      # then add the score
      term_id = term_dict[term]
      score += index.get(term_id, 0)
...
```

## Výsledky

Pri využití prirodzeného TF a IDF (v obr. naľavo) vyzerajú výsledky pre dopyt “michael chicago” trošku inak ako pri použití logaritmického TF a IDF (v obr. napravo):

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Alex Michael Caruso | 30yo | alive |
4: | Kristofer Michael Dunn | 30yo | alive |
5: | Aaron Michael Gray | 39yo | alive |
6: | Michael Edward Harper | 66yo | alive |
7: | Michael George Williams | 61yo | alive |
8: | Michael Donald Novak | 63yo | deceased |
9: | Michael David Ruffin | 47yo | alive |
10: | Andrew Michael Phillip | 79yo | deceased |
```

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Michael Edward Harper | 66yo | alive |
4: | Michael George Williams | 61yo | alive |
5: | Michael Donald Novak | 63yo | deceased |
6: | Alex Michael Caruso | 30yo | alive |
7: | Kristofer Michael Dunn | 30yo | alive |
8: | Aaron Michael Gray | 39yo | alive |
9: | Larry Michael Spriggs | 65yo | alive |
10: | Michael David Ruffin | 47yo | alive |
```

Keď použijem prirodzený TF probabilistic IDF (v obr. napravo) tak pre dopyt “michael chicago” vyzerajú výsledky znova trošku inak oproti prirodzenému TF a IDF (v obr. naľavo):

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Alex Michael Caruso | 30yo | alive |
4: | Kristofer Michael Dunn | 30yo | alive |
5: | Aaron Michael Gray | 39yo | alive |
6: | Michael Edward Harper | 66yo | alive |
7: | Michael George Williams | 61yo | alive |
8: | Michael Donald Novak | 63yo | deceased |
9: | Michael David Ruffin | 47yo | alive |
10: | Andrew Michael Phillip | 79yo | deceased |
```

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Alex Michael Caruso | 30yo | alive |
4: | Kristofer Michael Dunn | 30yo | alive |
5: | Michael Edward Harper | 66yo | alive |
6: | Michael George Williams | 61yo | alive |
7: | Aaron Michael Gray | 39yo | alive |
8: | Michael Donald Novak | 63yo | deceased |
9: | Michael David Ruffin | 47yo | alive |
10: | Andrew Michael Phillip | 79yo | deceased |
```

Po použití logaritmického TF a probabilistic IDF (v obr. napravo) tak pre dopyt “michael chicago” vyzerajú výsledky úplne inak oproti prirodzenému TF a IDF (v obr. naľavo):

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Alex Michael Caruso | 30yo | alive |
4: | Kristofer Michael Dunn | 30yo | alive |
5: | Aaron Michael Gray | 39yo | alive |
6: | Michael Edward Harper | 66yo | alive |
7: | Michael George Williams | 61yo | alive |
8: | Michael Donald Novak | 63yo | deceased |
9: | Michael David Ruffin | 47yo | alive |
10: | Andrew Michael Phillip | 79yo | deceased |
```

```
1: | Michael Damien Sweetney | 41yo | alive |
2: | Michael Jeffrey Jordan | 61yo | alive |
3: | Michael Edward Harper | 66yo | alive |
4: | Michael George Williams | 61yo | alive |
5: | Michael Donald Novak | 63yo | deceased |
6: | Alex Michael Caruso | 30yo | alive |
7: | Kristofer Michael Dunn | 30yo | alive |
8: | Aaron Michael Gray | 39yo | alive |
9: | Larry Michael Spriggs | 65yo | alive |
10: | Michael David Ruffin | 47yo | alive |
```

Tiež som pre tento dopyt “michael chicago” skúšal rátať vypísaný tím pre danú sezónu ako slovo iba raz, aby to nefavorizovalo hráčom, ktorý odohrali za daný tím viac sezón, preto v predošlých obrázkoch vidíme medzi prvými pozíciami vždy “Michael Jeffrey Jordan”, lebo za “Chicago Bulls” odohral veľa sezón. Po vyskúšaní výsledky dopytu vyzerajú nasledovne:

```
1: | Michael Jeffrey Jordan | 61yo | alive |
2: | Michael Damien Sweetney | 41yo | alive |
3: | Alex Michael Caruso | 30yo | alive |
4: | Kristofer Michael Dunn | 30yo | alive |
5: | Aaron Michael Gray | 39yo | alive |
6: | Michael Edward Harper | 66yo | alive |
7: | Michael George Williams | 61yo | alive |
8: | Michael Donald Novak | 63yo | deceased |
9: | Michael David Ruffin | 47yo | alive |
10: | Andrew Michael Phillip | 79yo | deceased |
```

```
1: | Michael Edward Harper | 66yo | alive |
2: | Michael George Williams | 61yo | alive |
3: | Michael Donald Novak | 63yo | deceased |
4: | Michael Damien Sweetney | 41yo | alive |
5: | Larry Michael Spriggs | 65yo | alive |
6: | Alex Michael Caruso | 30yo | alive |
7: | Michael David Holton | 63yo | alive |
8: | Aaron Michael Gray | 39yo | alive |
9: | Kristofer Michael Dunn | 30yo | alive |
10: | Michael Jeffrey Jordan | 61yo | alive |
```

## Zdrojový kód

xbartosp2_odovzdanie_1.zip

vi2024

# NBA - štvrtá konzultácia

## PySpark

Celý wikidump treba načítavať postupne, aby program nespadol s nedostatkom pamäte. To program robí tak, že číta súbor pomocou natívnej knižice pre python bz2 ešte komprimovaný. Ďalej si zadefinuje pole “pages” a postupne číta súbor riadok po riadku a hľadá “<page>” a “</page>” tagy. Keď nájde “<page>”, tak vie, že začína stránka, tak to appenduje do stringu “page” každý prečítaný riadok. Keď nájde tag “</page>”, tak končí appendovanie stringu, uloží “page” string do poľa “pages”, premaže “page” a ide znova. Teraz sa nazbiera určitý počet stránok v poli “pages” a ten počet je definovaný premennou “pages_size”, čo predstavuje chunk stránok. Pseudokód vyzerá nasledovne:

Download

```
Function read_wikidump_in_chunks(wikidump_path: string, pages_size=5000):
    Open file reading with bz2 as "FILE":
        Init "pages" list
        Init "page" string
        Init "is_in_page" boolean
        For "line" in "FILE":
            If length of "pages" == "pages_size":
                Process pages further
                Reset "pages" list
            If "<page>" in "line" and "is_in_page" is False:
                Reset "page" string
                Set "is_in_page" to True
            Elif "</page" not in "line" and "is_in_page" is True:
                Append "line" to "page" string
            Elif "</page>" in "line" and "is_in_page" is True:
                Append "page" string to "pages" list
                Set "is_in_page" to False
```

Keď sa teda nazbiera určitý počet stránok v poli, tak sa ide najprv určiť, že či je stránka vôbec relevantná. Ak je relevantná, tak sa spustia regexy na extrakciu nových entít (“Rookie of the year”, “NBA Most Valuable Player”, “NBA Champion”, “Hall of Famer”), ktoré doplnia predošlé dáta, viď obrázok:


Najprv sa ale toto pole stránok Sparkom paralelne spracuje a pomocou flatMap funkcie sa regexom vyfiltrujú iba relevantné stránky. Ďalej sa flatMapom a filtrom z tohto RDD vyextrahujú relevantné informácie pre hráčov NBA a tie sa následne uložia do CSV. Pseudokód vyzerá nasledovne:

Download

```
Function extract_relevant_records(text: str):
    Gets "matches" from all regex matches "{{Infobox[\s\S]basketball[\s\S]biography[\s\S]*?}}[ \t\r\n]*'''"
    Return empty list if no "matches", otherwise return "matches"

Function extract_additional_info(relevant_page: str, players_names: list[str]):
    Init "player_name" from searching via regex "\|\s*name\s*=\s*([^\n|]+)"
    If "player_name" from "relevant_page" is in "players_names":
        Extract "roty" via regex "\*[\s\S]?(\d+)?×?[\s\S]?\[\[.*?NBA Rookie of the Year\]\]"
        Extract "mvp" via regex "\*[\s\S]?(\d+)?×?[\s\S]?\[\[.*?NBA Most Valuable Player\]\]"
        Extract "champ" via regex "\*[\s\S]?(\d+)?×?[\s\S]?\[\[.*?NBA champion\]\]"
        Extract "hall_of_fame" via regex "\| HOF_player"

Function wikidump_parse_pages(pages: list[str], players_names: list[str]):
    Get "wiki_rdd" that is initialized through sparks parallelize function on "pages" list
    Get "relevant_pages" using sparks flatMap on "wiki_rdd" and function "extract_relevant_records"
    Get "additional_information" using sparks flatMap and filter on "relevant_pages" using "extract_additional_info"
    Write "additional_information" into CSV dump
```

V tejto časti boli použité knižnice Pythonu:

- “re” - regexy (natívna)

- “os” - na narábanie s OS (natívna)
- “bz2” - na čítanie komprimovaných súborov (natívna)
- “pyspark”

## Lucene

### Indexer

Najprv sa začne čítať CSV s uloženými dátami z wiki a použije sa knižnica “csv” pre načítavanie riadkov rovno do dictionary, aby sa nemusela robiť logika na parsovanie fieldov. Teraz pôjde cyklus záznam po zázname a na každý záznam sa vytvorí lucene dokument a k tomu dokumentu sa budú pridávať jednotlivé fieldy. Textové fieldy sú zadefinované pre “name”, “position”, “shoots”, “born_country”, “highschool”, “college”, “nba_roty”, “nba_mvp”, “nba_champion”, “seasons”, “teams” a “hall_of_fame”. String fieldy sú zadefinované pre “weight”, “height”, “age”, “life_status”. Pseudokód pre indexer vyzerá nasledovne:

Download

```
Function index_wiki_data():
    Init "WRITER" as lucene's index writer
    With open read file with raw wiki player info as "FILE"
        Init "READER" using "csv" lib dict reader
        For "player" in "READER":
            Init "doc" as lucene dcument for player
            Add "name" field as TextField to "doc"
            Add "height" field as StringField to "doc"
            ...

            Add "teams" field as TextField to "doc"
            Using "WRITER" add "doc" to index
    Commit with "WRITER"
    Close "WRITER"
```

V tejto časti boli použité knižnice Pythonu:

- “os” - na narábanie s OS (natívna)
- “csv” - na narábanie s CSV súbormi (natívna)
- “lucene”

### Searcher

Používateľ poskytne query na vyhľadávanie. Najprv sa inicializuje lucene query parser, kde sa mu poskytnú všetky fieldy a štandartný analyzér. Ďalej z query parsera sa zavolá funkcia parse, do ktorej ako vstupné parametre ide používateľská query, list fieldov, list flagov pre jednotlivé fieldy a analyzér. Potom nad lucene index searcherom sa zavolá funkcia search a ako vstupné parametre ide používateľova query a číslo reprezentujúce aký počet výsledkov sa vráti. Z výsledkov sa zavolá atribút scoreDocs, ktorý predstavuje jednotlivé hity. Následne v cykle sa z týchto výsledkov vyťahuje doc id a vytiahne sa celý dokument z lucene search indexera. Nakoniec sa tieto výsledky zobrazia používateľovi. Pseudokód pre searcher vyzerá nasledovne:

Download

```
Function search(query: str, fields: list[str], flags: list, result_num=10):
    Init "SEARCHER" as lucene's index searcher
    Init "ANALYZER" as lucene's standard analyzer
    Init "query_parser" as lucene's query parser with params "fields" and "ANALYZER"
    Init "query" with "query_parser" parse function with params "query", "fields", "flags" and "ANALYZER"
    Init "hits" with "SEARCHER" search function with params "query" and "result_num"
    Init "results" list to store result documents
    For "hit" in "hits":
        Init "doc" with "SEARCHER" document function with param "hit" using attribute "doc"
        Append "doc" into "results" list
    Return "results" list
```

Vzorka výstupu vyhľadávača:

```
-> search serbia center champion
(+) Listing found results
| NAME | AGE | VITAL |
1: | Miloš Babić | 55yo | alive |
2: | Radisav Ćurčić | 59yo | alive |
3: | Darko Miličić | 39yo | alive |
4: | Nikola Jokić | 29yo | alive |
5: | Rastko Cvetković | 54yo | alive |
6: | Vlade Divac | 56yo | alive |
7: | Nenad Krstić | 41yo | alive |
8: | Boban Marjanović | 36yo | alive |
9: | Filip Petrušev | 24yo | alive |
10: | Miroslav Raduljica | 36yo | alive |
(-) Write number to lookup from results: 4
(+) Detailed player result
| NAME | AGE | POSITION | SHOOTS | HEIGHT | WEIGHT | HIGHSCHOOL | COLLEGE | BORN | VITAL | ROTY | MVP | CHAMPION | HOF |
 SEASONS |
4: | Nikola Jokić | 29yo | Center | Righty | 211cm | 128kg | Unknown | Unknown | Serbia | alive | No | Yes | Yes | Not a
 member | 2015_DEN 2016_DEN 2017_DEN 2018 DEN 2019_DEN 2020_DEN 2021_DEN 2022_DEN 2023_DEN |
(-) Write number to lookup from results:
```

V tejto časti boli použité knižnice Pythonu:

- “os” - na narábanie s OS (natívna)
- “csv” - na narábanie s CSV súbormi (natívna)
- “lucene”

## Zapracovanie nedostatkov

4/5. Konzultacia - body na zapracovanie:

- treba spraviť fuzzy merge záznamov
- extrakcia z textu wikipedie pre buď HoF aleb rank (vybral som si HoF)
- extrakcia nickname-u z wikipedie
- porovnanie s 1. riešením → 2. odovzdanie 02.12.2024 (pretože sa nedá porovnávať ani s google a ani s basketball-reference)

### Fuzzy merge

Je potrebné pri merge-ovaní dát z wikipédie a dát z basketball-reference počítať skóre podobnosti. To bolo počítané podľa mena hráča a ak bolo nad 80%, tak sa záznam o hráčovi z wikipédie spojil zo záznamom o hráčovi z basketbal-reference. Pseudokód vyzerá nasledovne:

Download

```
Function fuzzy_merge_based_on_names(csv_name: str, wiki_name: str):
    Init "similarity_score" as integer
    Use function "fuzz.ratio" with parameters "csv_name" and "wiki_name" to calculate "similarity_score"
    if "similarity_score" is higher than 80:
        return True
    return False
```

V tejto časti bola použitá knižnica Pythonu:

- "fuzzywuzzy" - na počítanie similarity score medzi ret'azcami (externá)

### Extrakcia vety o uvedení hráča v sieni slávy (HOF)

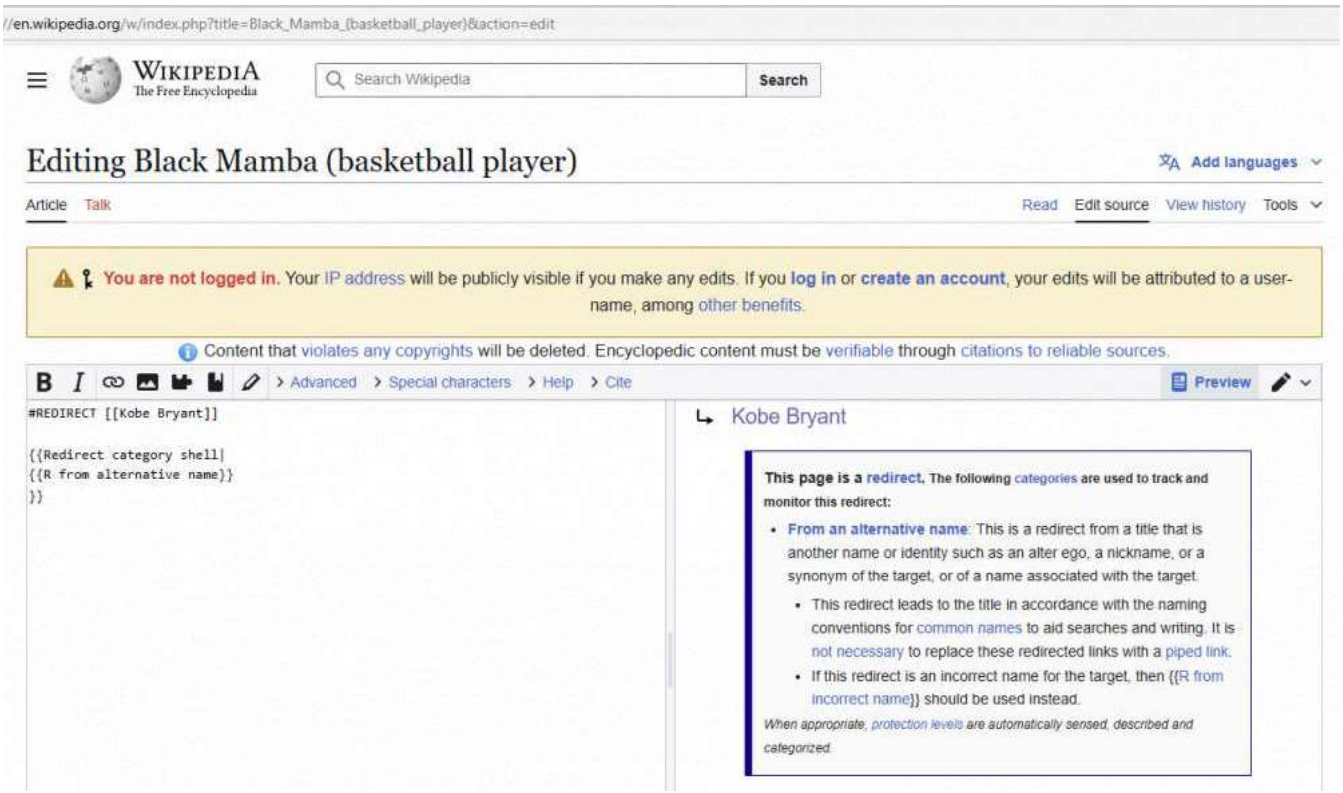
Miesto nedostačujúceho regexu v pôvodnej implementácií bolo potrebné nahradiť takým, aby v úvodnej časti odseku o hráčovi bola vyregexovaná celá veta o jeho uvedení v HOF pre obohatenie Lucene indexu textom. Vytiahnutie tejto vety bolo spravené tak, že najprv sa vyregexoval celá úvodná kapitola o hráčovi. Odtiaľto sa spomedzi všetkých viet podľa vhodne určených slov (member|inductee,induct|ed,vote|ed,enshrine|ed,…, hall of fame,…) táto veta vytiahla. Vďaka starému regexu na tento field, ktorý sa t'ahal z infoboxu, bolo možné určiť úspešnosť tohto regexu a je to 83.5% (66 z 79). Pseudokód extrakcie vyzerá nasledovne:

Download

```
def extract_first_section_and_sentence_about_hof(page_text: str, player_name: str):
    Init regex "'''{player_name}'''(.*?)\n\s*==\s*" to extract whole introduction text
    Init "section_match" from using regex on "page_text"
    If "section_match" was found:
        Init "intro_text" from the found match
        Init regex "(?:[A-Z][^.?!]*?\s)?(?:induct(?:ee|ed)?|vote(?:d)?|enshrine(?:d)?|member|elect(?:ed)?|name(?:d)?)\s+?(?:\[\[)?(?:[Nn]aismith\s+)?(?:.*?\s+)?[Hh]all\s+o
        to match the sentence
        Init "sentence_match" from using regex on "intro_text"
        if "sentence_match":
            return "hof_sentence" string from "sentence_match"
    return ""
```

### Extrakcia prezývky hráča

Pri poslednej konzultácií sa prišlo na to, že ako redirecty sa na wikipédií ukladajú prezývky hráčov, napr. je stránka "Black Mamba" a jediný jej kontent je redirect na Kobe Bryanta (#REDIRECT \[\[Kobe Bryant\]\]). Takýto redirect vyzerá nasledovne:




Čiže implementácia pre hľadanie prezýviek vyzerala tak, že všade sa iba po stránkach kontrolujú redirecty, kde do regexu sa natlačí meno hráča. Zoberie sa zoznam mien hráčov z basketball-reference a na každej stránke sa skúšajú takéto regexy a ak sa nájde, tak sa uloží prezývka. Pseudokód vyzerá nasledovne:

Download

```
def try_to_extract_nickname_from_page(page_text: str, players_names: list[str]):
    for player_name in players_names:
        Init "redirect_regex" as "#REDIRECT \[\[{player_name}\]\]"
        Init "redirect_match" on "page_text" using pattern "redirect_regex"
        If "redirect_match":
            # nickname is in the page
            Init "nickname_regex" as "<title>(.+?)</title>"
            Init "nickname_match" on "page_text" using pattern "nickname_regex"
            if "nickname_match":
                return "nickname" from "nickname_match"
```

vi2024

# NBA - odovzdanie 2

## Spark

Na celej wikipédii sa nachádzalo 3196 NBA hráčov z 5204 NBA hráčov, čo tvorí 61.4%. Pre týchto hráčov sa podarilo pridať nové informácie ohľadom trofejí hráčov, ako "Rookie of the year", "Most Valuable Player", "Champion" a "Hall of Fame". Z celkového wiki datasetu iba 39 NBA hráčov má trofej "Rookie of the year" (1.2% z 3196). Ďalej "Most Valuable Player" má iba 23 NBA hráčov (0.7% z 3196). Pohár NBA vyhralo 343 NBA hráčov (10.7% z 3196). Nakoniec v "Hall of Fame" je 79 NBA hráčov (2.4% z 3196). Po zapracovaní nedostatkov z poslednej konzultácií bola pridaná ďalšia entita "nickname" z wikipédie, kde 347 hráčov malo nejakú prezývku (10.8% z 3196).

Dáta pred spracovaním Sparkom vyzerajú nasledovne:

| Career_length | Seasons | Teams | Season_team | Nba_roty | Nba_mvp | Nba_champion | Hall_of_fame |
|---|---|---|---|---|---|---|---|
| 5 years | 2010 2011 2012 | GSW HOU CHA | 2010_GSW 2011 | | | | |
| 7 years | 2008 2009 2010 | CHA CHA DAL T | 2008_CHA 2009_ | | | | |
| 1 year | 2020 | PHO | 2020_PHO | | | | |
| 11 years | 2007 2008 2009 | DET DET DEN DE | 2007_DET 2008_ | | | | |
| 7 years | 1968 1969 1970 | SDR SDR POR P | 1968_SDR 1969_ | | | | |
| 3 years | 2016 2017 2018 | OKC OKC OKC | 2016_OKC 2017_ | | | | |
| 13 years | 1975 1976 1977 | PHO PHO PHO F | 1975_PHO 1976_ | | | | |
| 2 years | 1959 1960 | DET LAL | 1959_DET 1960_ | | | | |
| 11 years | 1985 1986 1987 | SAC WSB DEN D | 1985_SAC 1986_ | | | | |
| 20 years | 1969 1970 1971 | MIL MIL MIL MIL | 1969_MIL 1970_ | | | | |
| 8 years | 2010 2011 2012 | OKC OKC HOU S | 2010_OKC 2011_ | | | | |
| 7 years | 2017 2018 2019 | BRK BRK BRK BR | 2017_BRK 2018_ | | | | |
| 1 year | 1984 | GSW | 1984_GSW | | | | |
| 5 years | 1990 1991 1992 | POR POR MIL BC | 1990_POR 1991_ | | | | |
| 5 years | 1986 1987 1988 | DEN WSB WSB V | 1986_DEN 1987_ | | | | |
| 1 year | 2018 | DEN | 2018_DEN | | | | |
| 2 years | 2005 2008 2008 | DET DET LAC | 2005_DET 2008_ | | | | |

Dáta po spracovaní Sparkom vyzerajú nasledovne:

| Teams | Season_team | Nba_roty | Nba_mvp | Nba_champion | Hall_of_fame | Nickname | Hof_text |
|---|---|---|---|---|---|---|---|
| MEM M | 2021_MEM 20 | | | | Hall of Famer | | |
| MIL MI | 1969_MIL 197 | NBA Rookie | NBA Most Val | NBA Champion | Hall of Famer | | |
| MIL MI | 1996_MIL 199 | | | NBA Champion | Hall of Famer | | He played 18 seasons in the National Basketball Association (NBA) and wa |
| PHI PH | 1984_PHI 198 | | NBA Most Val | | Hall of Famer | Sir Charles | Barkley is a two-time inductee into the Naismith Memorial Basketball Hall |
| STL STI | 1962_STL 196 | | | | Hall of Famer | | A three-time ABA All-Star and two-time NBA All-Star, Beaty was inducted i |
| BOS TC | 1997_BOS 199 | | | NBA Champion | Hall of Famer | | In 2024, it was announced that Billups would be inducted into the Naismit |
| BOS BC | 1979_BOS 198 | NBA Rookie | NBA Most Val | NBA Champion | Hall of Famer | | He was inducted into the Naismith Memorial Basketball Hall of Fame twice |
| CIN CI | 1958_CIN 195 | | | | Hall of Famer | | |
| NYK N | 1947_NYK 194 | | | NBA Champion | Hall of Famer | | |
| LAL LA | 1996_LAL 199 | | NBA Most Val | NBA Champion | Hall of Famer | Black Mamba | Bryant was posthumously voted into the Naismith Memorial Basketball Hal |
| CHI CH | 1984_CHI 198 | NBA Rookie | NBA Most Val | NBA Champion | Hall of Famer | MJ | Jordan was twice inducted into the Naismith Memorial Basketball Hall of F |
| MEM M | 2004_MEM 20 | | | | Hall of Famer | | |

Implementácia čítania veľkého súboru:

Download

```
def read_wikidump_in_chunks(file_path: str, pages_size=5000):
    # opening compressed wikidump
    with bz2.open(file_path, "rt", encoding="utf-8") as FILE:
        pages = []
        page = ""
        in_page = False
        # page extraction
        for line in FILE:
            # if its time to process the pages chunk in spark
            if len(pages) == pages_size:
                # parse the pages
                wikidump_parse(pages)
                # reset the pages chunk
                pages = []
            # if page is in line, loop is in page, so reset page string
            if START_PAGE in line and not in_page:
                in_page = True
                page = ""
            # if loop is in page, then append the line to page string
            elif END_PAGE not in line and in_page:
                page += line
            # if at the end of the page, then return this specific page and continue then
            elif END_PAGE in line and in_page:
                pages.append(page)
                in_page = False
```

## Lucene - indexer

Implementácia lucene indexera:

Download

```
def index_wiki_data():
    # opening file to read
    with open(constants.WIKI_PLAYER_INFO_PATH, mode='r', encoding='utf-8') as FILE:
        # reading file rows into dicts
        READER = csv.DictReader(FILE, delimiter=';')
        # going through each player record
        for player in READER:
            # doc record for player
            doc = Document()
            # adding fields for indexing
            doc.add(TextField("name", player["name"], Field.Store.YES))
            doc.add(StringField("weight", player["weight"], Field.Store.YES))
            doc.add(StringField("height", player["height"], Field.Store.YES))
            doc.add(TextField("position", player["position"], Field.Store.YES))
            country_of_origin = constants.COUNTRY_CODES[player["born_in_country"]]
            ...
            doc.add(TextField("nba_roty", player["nba_roty"], Field.Store.YES))
            doc.add(TextField("nba_mvp", player["nba_mvp"], Field.Store.YES))
            doc.add(TextField("nba_champion", player["nba_champion"], Field.Store.YES))
            doc.add(TextField("hall_of_fame", player["hall_of_fame"], Field.Store.YES))
            ...
            doc.add(StoredField("season_team", player["season_team"]))
            # write the document of the newly populated player
            WRITER.addDocument(doc)
    WRITER.commit()
    WRITER.close()
```

## Lucene - searcher

Default fields:

- name
- position
- shoots
- height
- age
- college
- highschool
- country
- alive

Implementácia lucene searchera:

Download

```
def full_text_search(query_string: str, searcher, analyzer, fields: list[str], flags: list, result_num=10):
    # init query parser
    query_parser = MultiFieldQueryParser(fields, analyzer)
    query = query_parser.parse(query_string, fields, flags, analyzer)
    # retrieving top n results
    hits = searcher.search(query, result_num).scoreDocs
    # now retrieving the results
    results = []
    for hit in hits:
        # getting the document
        doc = searcher.storedFields().document(hit.doc)
        # getting the result into a dict
        result = {field: doc.get(field) for field in fields if doc.get(field)}
        results.append(result)
    # returning results
    return results
```

## Výsledky

Naľavo pylucene napravo vlastné riešenie (výsledky ohraničené na 10 podľa konzultácie):

1. Pre dopyt zobrazený na obrázku dole je potrebné po vyhľadávaní získať iba hráčov zo Srbska, ktorí popri tom majú pozíciu pivota. Obidve riešenia tu spravili všetko správne, líši sa iba usporiadanie. Každý jeden z nich je aj v jednom aj druhom riešení je zo Srbska a hral/hráva pozíciu pivota. P = 10/10+0 = 100%, R = 10/10+0 = 100%. Zobrazenie vykonaného dopytu:

```
-> search serbia center
(+) Listing found results
| NAME | AGE | VITAL |
1: | Miloš Babić | 55yo | alive |
2: | Radisav Ćurčić | 59yo | alive |
3: | Darko Miličić | 39yo | alive |
4: | Rastko Cvetković | 54yo | alive |
5: | Vlade Divac | 56yo | alive |
6: | Nikola Jokić | 29yo | alive |
7: | Nenad Krstić | 41yo | alive |
8: | Boban Marjanović | 36yo | alive |
9: | Filip Petrušev | 24yo | alive |
10: | Miroslav Raduljica | 36yo | alive |
```

```
-> search serbia center
(+) Listing found results
| NAME | AGE | VITAL |
1: | Radisav Ćurčić | 59yo | alive |
2: | Miloš Babić | 55yo | alive |
3: | Rastko Cvetković | 54yo | alive |
4: | Dragan Tarlač | 51yo | alive |
5: | Vladimir Vranеš | 41yo | alive |
6: | Miroslav Raduljica | 36yo | alive |
7: | Filip Petrušev | 24yo | alive |
8: | Željko Rebrača | 52yo | alive |
9: | Darko Miličić | 39yo | alive |
10: | Nikola Jokić | 29yo | alive |
```

2. Ďalší dopyt je ohľadom nájdenia ľavákov, ktorý hrali pozíciu rozohrávača v sezóne 1994. Oba dopyty sú znova správne, všetci sú ľaváci, rozohrávači a zo sezóny 1994. Tentokrát sa líšia v zoradení iba 3 hráčov. Pre Lucene sa ale musí query líšiť od query vlastného riešenia. P = 10/10+0 = 100%, R = 10/10+0 = 100%. Zobrazenie druhého dopytu:

```
-> search position:"point guard" AND shoots:lefty AND seasons:1994
(+) Listing found results
| NAME | AGE | VITAL | SHOOTS | POSITION |
1: | Anthony Guy Bennett | 55yo | alive | Lefty | Point Guard |
2: | Brooks James Thompson | 45yo | deceased | Lefty | Point Guard |
3: | Johnny Earl Dawkins Jr. | 61yo | alive | Lefty | Point Guard |
4: | Gregory Carleton Anthony | 57yo | alive | Lefty | Point Guard |
5: | John Kevin Crotty | 55yo | alive | Lefty | Point Guard |
6: | Elliot Lamonte Perry | 55yo | alive | Lefty | Point Guard |
7: | Darrick David Martin | 53yo | alive | Lefty | Point Guard |
8: | Nickey Maxwell Van Exel | 53yo | alive | Lefty | Point Guard |
9: | Kenneth Anderson | 54yo | alive | Lefty | Point Guard |
10: | Avery DeWitt Johnson | 59yo | alive | Lefty | Point Guard |
```

```
-> search point guard lefty 1994
(+) Listing found results
| NAME | AGE | VITAL | SHOOTS | POSITION |
1: | Anthony Guy Bennett | 55yo | alive | Lefty | Point Guard |
2: | Brooks James Thompson | 45yo | deceased | Lefty | Point Guard |
3: | Johnny Earl Dawkins Jr. | 61yo | alive | Lefty | Point Guard |
4: | John Kevin Crotty | 55yo | alive | Lefty | Point Guard |
5: | Elliot Lamonte Perry | 55yo | alive | Lefty | Point Guard |
6: | Gregory Carleton Anthony | 57yo | alive | Lefty | Point Guard |
7: | Darrick David Martin | 53yo | alive | Lefty | Point Guard |
8: | Nickey Maxwell Van Exel | 53yo | alive | Lefty | Point Guard |
9: | Kenneth Anderson | 54yo | alive | Lefty | Point Guard |
10: | Avery DeWitt Johnson | 59yo | alive | Lefty | Point Guard |
```

3. Ten istý dopyt ako druhý, ale tentokrát je nechaná rovnaká query v Lucene vyhľadávači ako vo vlastnom riešení. Usporadanie výsledkov je o dosť iné, ale hlavne sú získané odlišné záznamy. Lucene kvôli jeho default-nému OR berie výsledky, ktoré nie sú relevantné. Tým pádom vznikajú horšie metriky: P = 5/5+5 = 50%, R = 5/5+5 = 50%. Zobrazenie tretieho dopytu:

```
-> search point guard lefty 1994
(+) Listing found results
| NAME | AGE | VITAL | SHOOTS | POSITION |
1: | Melvin Jermaine Booker | 52yo | alive | Righty | Point Guard |
2: | Litterial Green | 54yo | alive | Righty | Point Guard |
3: | Devin Armani Booker | 28yo | alive | Righty | Shooting Guard |
4: | Anthony Guy Bennett | 55yo | alive | Lefty | Point Guard |
5: | Brooks James Thompson | 45yo | deceased | Lefty | Point Guard |
6: | Samuel David Hauser | 26yo | alive | Righty | Small Forward |
7: | Ryan Lorthridge | 52yo | alive | Lefty | Shooting Guard |
8: | Johnny Earl Dawkins Jr. | 61yo | alive | Lefty | Point Guard |
9: | Gregory Carleton Anthony | 57yo | alive | Lefty | Point Guard |
10: | John Kevin Crotty | 55yo | alive | Lefty | Point Guard |
```

```
-> search point guard lefty 1994
(+) Listing found results
| NAME | AGE | VITAL | SHOOTS | POSITION |
1: | Anthony Guy Bennett | 55yo | alive | Lefty | Point Guard |
2: | Brooks James Thompson | 45yo | deceased | Lefty | Point Guard |
3: | Johnny Earl Dawkins Jr. | 61yo | alive | Lefty | Point Guard |
4: | John Kevin Crotty | 55yo | alive | Lefty | Point Guard |
5: | Elliot Lamonte Perry | 55yo | alive | Lefty | Point Guard |
6: | Gregory Carleton Anthony | 57yo | alive | Lefty | Point Guard |
7: | Darrick David Martin | 53yo | alive | Lefty | Point Guard |
8: | Nickey Maxwell Van Exel | 53yo | alive | Lefty | Point Guard |
9: | Kenneth Anderson | 54yo | alive | Lefty | Point Guard |
10: | Avery DeWitt Johnson | 59yo | alive | Lefty | Point Guard |
```

4. Štvrtý dopyt sa zameriava na hráčov, ktorý žijú, hrali pozíciu krídla a na VŠ chodili do UCLA. Výsledky pre obidve riešenia sú všetky relevantné, aj keď záznamy sa líšia, či už v usporiadaní alebo celkovo. Sú iba 3 záznamy rovnaké (Jaquez, Henderson, Fields). P = 10/10+0 = 100%, R = 10/10+0 = 100%. Zobrazenie štvrtého dopytu:

```
-> search players from ucla playing small forward who are still alive
(+) Listing found results
| NAME | AGE | VITAL | POSITION | COLLEGE |
1: | Kyle F. Anderson | 31yo | alive | Small Forward | Ucla |
2: | Trevor Anthony Ariza | 39yo | alive | Small Forward | Ucla |
3: | Matt Kelly Barnes | 44yo | alive | Small Forward | Ucla |
4: | Darren Keefe Daye | 64yo | alive | Small Forward | Ucla |
5: | Keith Raymond Erickson | 80yo | alive | Small Forward | Ucla |
6: | Kenneth Henry Fields | 62yo | alive | Small Forward | Ucla |
7: | Milton Henderson Jr. | 48yo | alive | Small Forward | Ucla |
8: | Jaime Jaquez Jr. | 23yo | alive | Small Forward | Ucla |
9: | Marques Kevin Johnson | 68yo | alive | Small Forward | Ucla |
10: | Jason Alan Kapono | 43yo | alive | Small Forward | Ucla |
```

```
-> search players from ucla playing small forward who are still alive
(+) Listing found results
| NAME | AGE | VITAL | POSITION | COLLEGE |
1: | Ray Lynn Shackelford | 77yo | alive | Small Forward | Ucla |
2: | Milton Henderson Jr. | 47yo | alive | Small Forward | Ucla |
3: | Jaime Jaquez Jr. | 23yo | alive | Small Forward | Ucla |
4: | Keith Kensel Owens | 55yo | alive | Small Forward | Ucla |
5: | Michael Edward Lynn | 78yo | alive | Small Forward | Ucla |
6: | Dijon Lynn Thompson | 41yo | alive | Small Forward | Ucla |
7: | James Robert Wilkes | 66yo | alive | Small Forward | Ucla |
8: | Edward Charles O'Bannon Jr. | 52yo | alive | Small Forward | Ucla |
9: | Kenneth Henry Fields | 62yo | alive | Small Forward | Ucla |
10: | Shabazz Naige Muhammad | 31yo | alive | Small Forward | Ucla |
```

5. Piaty dopyt mal vyhľadať ľavákov hráčov z Kansasu, ktorý sú strelci a hrali v Los Angeles Lakers. Takýto boli v NBA iba dvaja. Vlastné riešenie tým pádom zobrazí iba 2 záznamy. Naopak Lucene prinesie aj ďalších 8 záznamov okrem týchto dvoch, ktoré sú podobné. P = 2/2+8 = 20%, R = 2/2+0 = 100%. Zobrazenie piateho dopytu:

```
-> search lefty players from Kansas who played or are playing shooting guard in los angeles lakers
(+) Listing found results
| NAME | SHOOTS | POSITION | HIGHSCHOOL |
1: | Gail Charles Goodrich Jr. | Lefty | Shooting Guard | Polytechnic in Los Angeles |
2: | Robert Antell Sims Jr. | Righty | Shooting Guard | Jordan in Los Angeles |
3: | Kareem Lamar Rush | Lefty | Shooting Guard | Pembroke Hill in Kansas City |
4: | Nicholas Aaron Young | Righty | Shooting Guard | Cleveland in Los Angeles |
5: | Anthony Eugene Peeler | Lefty | Shooting Guard | Paseo in Kansas City |
6: | Jerome D. Henderson | Righty | Power Forward | Jefferson in Los Angeles |
7: | Darius Aaron Morris | Righty | Point Guard | Windward in Los Angeles |
8: | Larry Donnell Drew Sr. | Righty | Point Guard | Wyandotte in Kansas City |
9: | Dwayne L. Polee Sr. | Righty | Shooting Guard | Manual Arts in Los Angeles |
10: | Lucius Oliver Allen Jr. | Righty | Point Guard | Wyandotte in Kansas City |
```

```
-> search lefty players from Kansas who played or are playing shooting guard in los angeles lakers
(+) Listing found results
| NAME | SHOOTS | POSITION | HIGHSCHOOL |
1: | Kareem Lamar Rush | Lefty | Shooting Guard | Pembroke Hill in Kansas City |
2: | Anthony Eugene Peeler | Lefty | Shooting Guard | Paseo in Kansas City |
```

Priemerný precision: 74%. Priemerný recall: 90%.

Výhodou vlastného riešenia je to, že je presné a vždy vráti iba to, čo si používateľ vypýtal. Na druhú stranu ale vracia niekedy menej než dohodnutých 10 záznamov a nesnaží sa vôbec pri tom vracať aj podobné záznamy. Presné vyhľadávanie je tiež niekedy nevhod, keď používateľ presne nevie, že čo hľadá a systém mu nevie ani trochu napovedať. Zatiaľ čo Lucene vracia oproti presným výsledkom aj omáčky, ale na druhú stranu pri lepšie špecifikovanej query vie vracať rovnako presné výsledky ako vracalo vlastné riešenie. Nevýhodou tohto je, že používateľ musí byť aspoň trochu zaškolený ako to vyhľadávanie funguje na to, aby vedel toto presné vyhľadávanie využívať.

## Zdrojový kód

xbartosp2_odovzdanie_2.zip