NBA ontology

연세대학교 산업공학과
온톨로지공학
4조 김병진, 정세영, 장건, 김윤석

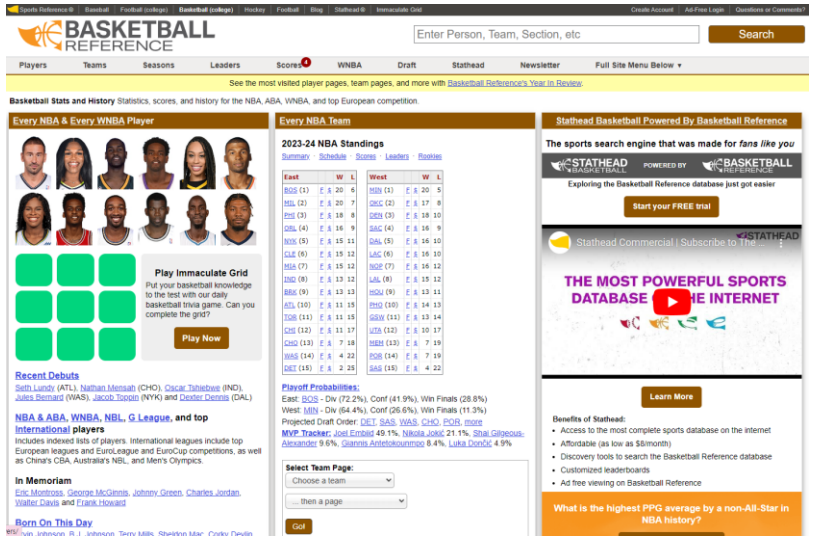
Contents

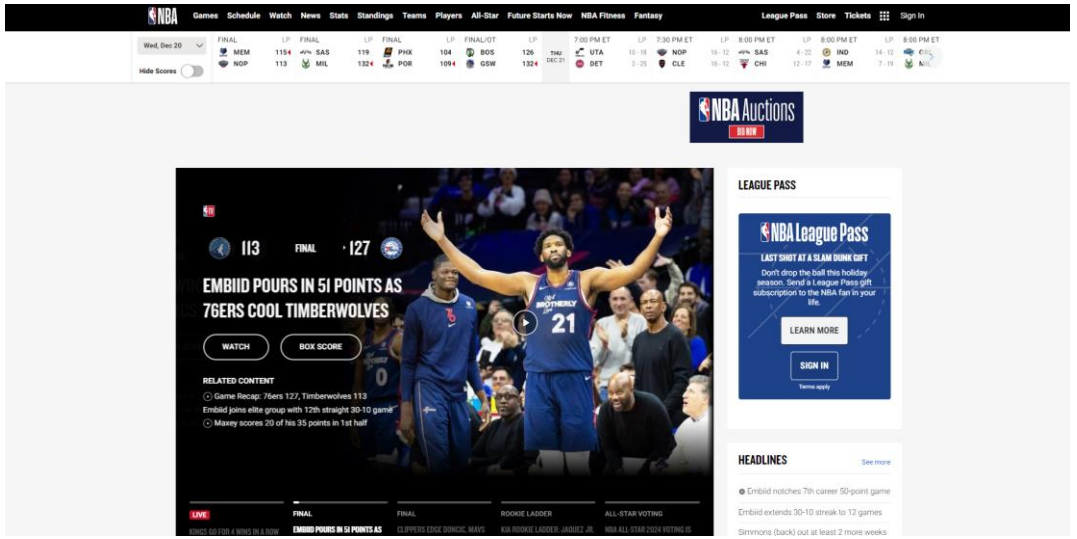
서비스 소개

개발 동기 및 목적

● 서비스 개발 동기

- 분산된 NBA 정보 (https://www.basketball-reference.com, https://www.nba.com)
  - 웹에서 선수 정보, 팀 구성원 정보, 팀 정보 등 다양한 정보 제공
  - 정보가 분산되어 있는 단점이 있음





개발 동기 및 목적

● 서비스 목적

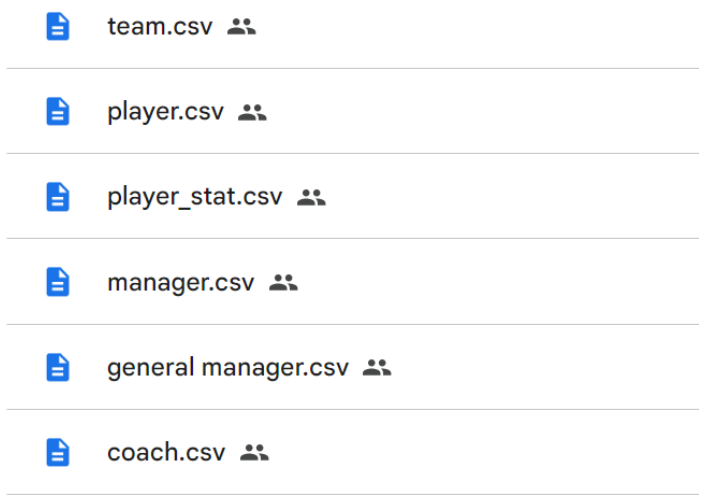
- NBA 팀, 감독, 코치, 선수 정보 조회
  - 스카우터는 선수들의 능력을 평가하기 위해 경기를 항상 관찰해야함
  - NBA 관련 정보 및 스탯을 조회할 수 있는 서비스로 스카우터들에게 매우 유용하고 중요한 도구를 제공함

개발 동기 및 목적

● ontology 구축 계획

▪ 데이터 구축

- 웹에서 각 정보를 csv 또는 xlsx 파일로 가져옴
- 각 파일은 coach, general manager, manager, player, team 정보를 갖는 파일로 각 객체에 대한 정보를 포함하고 있음
- player 파일은 나이, 출전 관련 정보, 야투 관련 정보, 자유투 관련 정보, 필드 플레이 내 정보 등이 있음
- team 파일은 팀의 창단 연도, 총 경기 수, 승리 및 패배 수 등 정보를 포함하고 있음
- 파일을 python을 통해 owlready2 코드 형식으로 바꿈



NBA ontology 구축

NBA ontology 구축

● Ontology 구축 내용

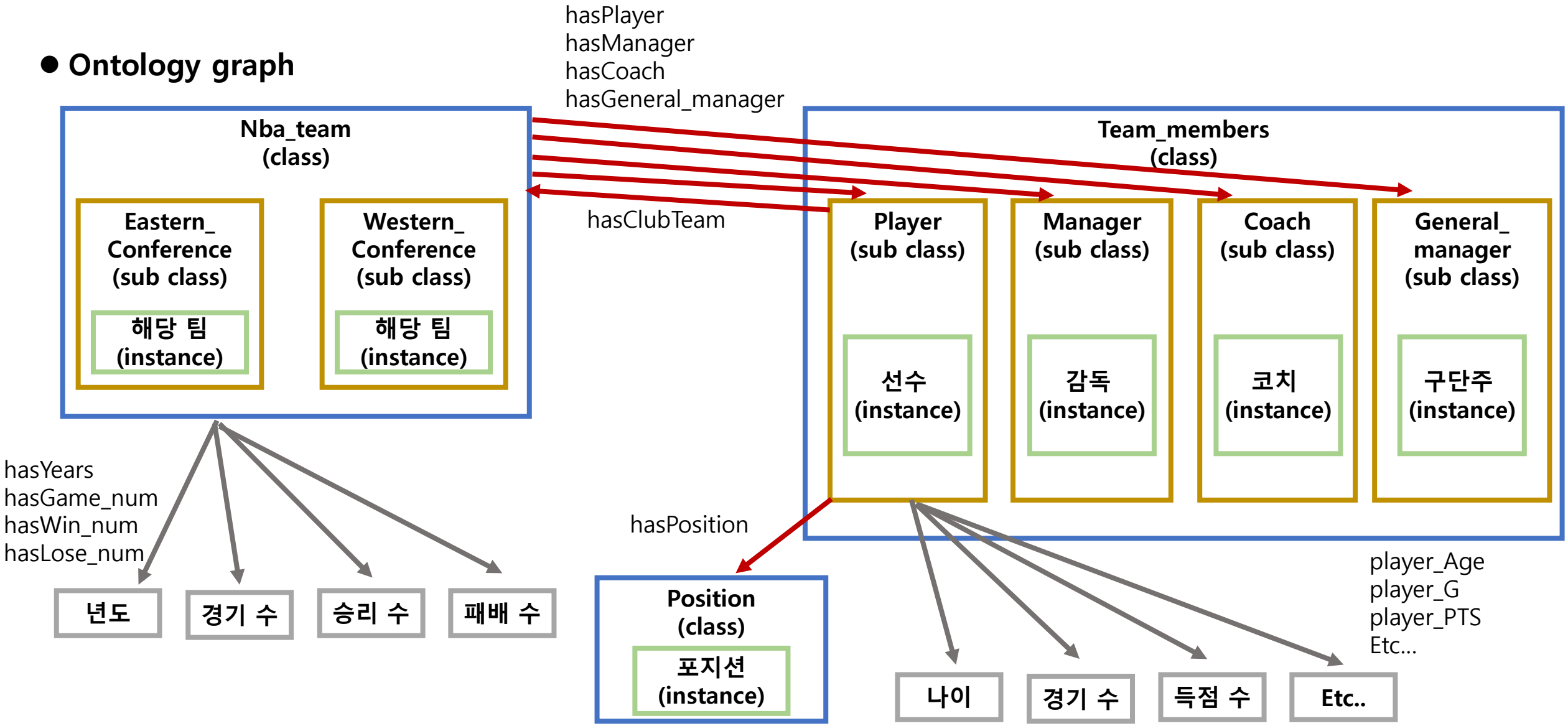
- **Class**
  - Team
    - Eastern_Conference
    - Western_Conference
  - Team_members
    - Coach
    - Manager
    - General_manager
    - Player
  - Position

- **DataProperty**
  - player_Age, player_G, player_MP, player_FG, player_FGA, player_FT, player_FTA, player_TRB, player_AST, player_STL, player_BLK, player_TOV, player_PF, player_PTS, hasYears, hasGame_num, hasWin_num, hasLose_num

- **Object Property**
  - hasCoach
    - NBA_Team -> Manager
  - hasManager
    - NBA_Team -> Manager
  - hasGeneral_manager
    - NBA_Team -> General_manager
  - hasClubTeam
    - Player -> NBA_Team
  - hasPlayer
    - NBA_Team -> Player
  - hasPosition
    - Player -> Position

NBA ontology 구축

● Ontology graph



NBA ontology 구축

● Ontology 로딩

- 웹상의 NBA 온톨로지를 get_ontology로 가져와 데이터 구조와 관계를 효율적으로 다룰 수 있음

```
from owlready2 import *
onto = get_ontology("http://NBA_ontology/onto.owl")

onto
```

```
* Owlready2 * Warning: optimized Cython parser module 'owlready2_optimized' is not available, defaulting to slower Python implementation

Warning: SQLite3 version 3.40.0 and 3.41.2 have huge performance regressions; please install version 3.41.1 or 3.42!

get_ontology("http://NBA_ontology/onto.owl#")
```

NBA ontology 구축

● Class 정의

- owlready2를 활용해 NBA 팀과 구성원을 온톨로지 클래스로 정의
- NBA_Team 클래스로 팀 분류하고, Team_members로 구성원 범주화
- 각 서브클래스로 코치, 매니저, 선수 등 세부적으로 나눔
- 이를 통해 NBA 팀 구조와 역할 체계적으로 분석 가능

```
[ ]  with onto:
        class NBA_Team(Thing): pass
        class Eastern_Conference(NBA_Team): pass
        class Western_Conference(NBA_Team): pass

        class Team_members(Thing): pass
        class Coach(Team_members): pass
        class Manager(Team_members): pass
        class General_manager(Team_members): pass

        class Player(Team_members): pass
        class Position(Thing): pass
```

NBA ontology 구축

● Property 정의

- owlready2 활용해 NBA 팀과 선수간 관계, 데이터 특성 온톨로지로 정의
- ObjectProperty로 코치, 매니저, 일반 매니저, 팀과 선수 관계 설정
- DataProperty, FunctionalProperty로 팀과 선수의 수치 데이터(출전 수, 출전 시간 등) 모델링
- 이를 통해 NBA 팀 구조와 선수 개인 성적 상세하게 분석 및 관리 가능

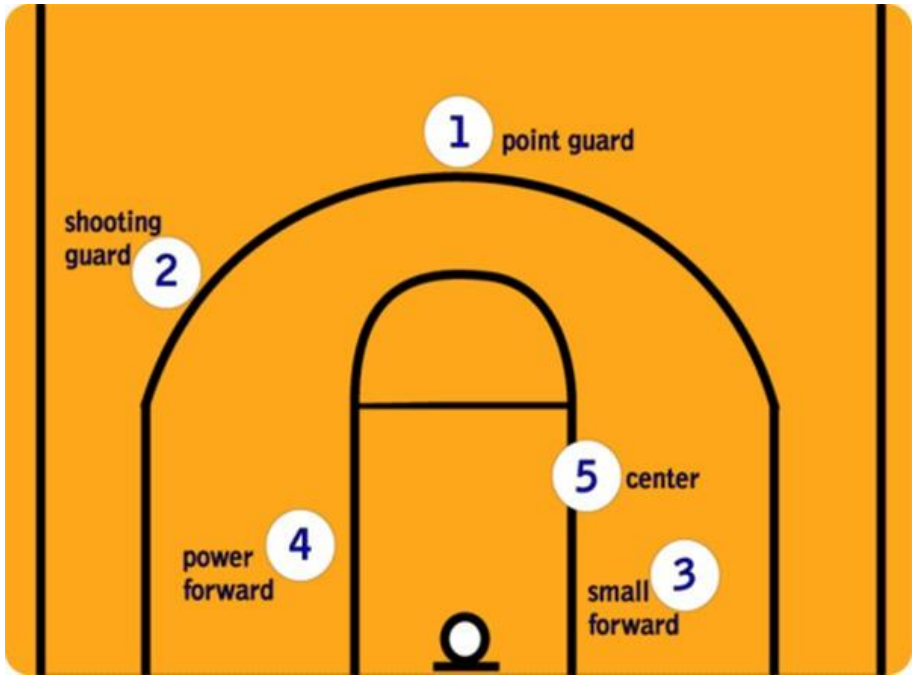
```
with onto:
    class hasCoach(ObjectProperty):
        domain  = [NBA_Team]
        range   = [Coach]
    class hasManager(ObjectProperty):
        domain  = [NBA_Team]
        range   = [Manager]
    class hasGeneral_manager(ObjectProperty):
        domain  = [NBA_Team]
        range   = [General_manager]
    class hasClubTeam(ObjectProperty):
        domain  = [Player]
        range   = [NBA_Team]
    class hasPlayer(ObjectProperty):
        domain  = [NBA_Team]
        range   = [Player]
        inverse_property = hasClubTeam
```

```
class hasYears(DataProperty, FunctionalProperty):
    domain = [NBA_Team]
    range  = [int]
class hasGame_num(DataProperty, FunctionalProperty):
    domain = [NBA_Team]
    range  = [int]
class hasWin_num(DataProperty, FunctionalProperty):
    domain = [NBA_Team]
    range  = [int]
class hasLose_num(DataProperty, FunctionalProperty):
    domain = [NBA_Team]
    range  = [int]
```

NBA ontology 구축

● Position instance 생성

- Position 클래스 이용해 NBA 포지션 인스턴스(CE, SG, PG, PF, SF) 생성. 각 인스턴스는 특정 포지션(센터, 슈팅가드 등) 대표.




```
[ ]  CE = Position("CE")
     SG = Position("SG")
     PG = Position("PG")
     PF = Position("PF")
     SF = Position("SF")
```

NBA ontology 구축

● Team instance 생성

- Eastern_Conference와 Western_Conference 클래스 사용해 NBA 팀 인스턴스 생성
- 팀마다 창단 연도, 총 경기 수, 승리 및 패배 수 데이터 속성 할당
- 이 인스턴스들로 팀 역사와 성적 데이터를 온톨로지에 구조화하여 저장

```
[ ]  Atlanta_Hawks = Eastern_Conference('Atlanta Hawks')
     Atlanta_Hawks.hasYears = 75
     Atlanta_Hawks.hasGame_num = 5872
     Atlanta_Hawks.hasWin_num = 2899
     Atlanta_Hawks.hasLose_num = 2973

     Boston_Celtics = Eastern_Conference('Boston Celtics')
     Boston_Celtics.hasYears = 78
     Boston_Celtics.hasGame_num = 6050
     Boston_Celtics.hasWin_num = 3584
     Boston_Celtics.hasLose_num = 2466

     Brooklyn_Nets = Eastern_Conference('Brooklyn Nets')
     Brooklyn_Nets.hasYears = 57
     Brooklyn_Nets.hasGame_num = 4547
     Brooklyn_Nets.hasWin_num = 2005
     Brooklyn_Nets.hasLose_num = 2542
```

NBA ontology 구축

● Coach instance 생성

- 각 NBA 팀에 다수의 코치 인스턴스 할당
- Coach 클래스로 코치 개인 생성 후, 해당 팀의 hasCoach 속성에 추가
- 이 데이터를 통해 각 팀의 코칭 스태프 정보를 체계적으로 관리 및 조회 가능

```
[ ]  Gregg_Farnam = Coach('Gregg_Farnam')
     Minnesota_Timberwolves.hasCoach.append(Gregg_Farnam)

     Elston_Turner = Coach('Elston_Turner')
     Minnesota_Timberwolves.hasCoach.append(Elston_Turner)

     Micah_Nori = Coach('Micah_Nori')
     Minnesota_Timberwolves.hasCoach.append(Micah_Nori)

     Pablo_Prigioni = Coach('Pablo_Prigioni')
     Minnesota_Timberwolves.hasCoach.append(Pablo_Prigioni)

     Kevin_Hanson = Coach('Kevin_Hanson')
     Minnesota_Timberwolves.hasCoach.append(Kevin_Hanson)

     Joe_Boylan = Coach('Joe_Boylan')
     Minnesota_Timberwolves.hasCoach.append(Joe_Boylan)
```

NBA ontology 구축

● General manager instance 생성

- General_manager 클래스로 각 NBA 팀의 구단주 인스턴스화
- 각 팀별 구단주 명단을 hasGeneral_manager 속성에 추가하여 온톨로지에 등록
- 이 데이터는 팀 경영진의 구성을 분석하는 데 활용 가능

```
[ ]  Calvin_Booth = General_manager('Calvin_Booth')
     Denver_Nuggets.hasGeneral_manager.append(Calvin_Booth)

     Elton_Brand = General_manager('Elton_Brand')
     Philadelphia_76ers.hasGeneral_manager.append(Elton_Brand)

     Chad_Buchanan = General_manager('Chad_Buchanan')
     Indiana_Pacers.hasGeneral_manager.append(Chad_Buchanan)

     Joe_Cronin = General_manager('Joe_Cronin')
     Portland_Trail_Blazers.hasGeneral_manager.append(Joe_Cronin)
```

NBA ontology 구축

● Manager instance 생성

- Manager 클래스로 NBA 각 팀의 감독 인스턴스를 생성
- 각 팀의 hasManager 속성에 감독 할당
- 데이터로 팀별 리더십 구조와 관리진을 명확하게 정리

```
[ ]  Quin_Snyder = Manager('Quin_Snyder')
     Atlanta_Hawks.hasManager.append(Quin_Snyder)

     Joe_Mazzulla = Manager('Joe_Mazzulla')
     Boston_Celtics.hasManager.append(Joe_Mazzulla)

     Jacque_Vaughn = Manager('Jacque_Vaughn')
     Brooklyn_Nets.hasManager.append(Jacque_Vaughn)

     Billy_Donovan = Manager('Billy_Donovan')
     Charlotte_Hornets.hasManager.append(Billy_Donovan)

     Steve_Clifford = Manager('Steve_Clifford')
     Chicago_Bulls.hasManager.append(Steve_Clifford)
```

NBA ontology 구축

● Player instance 생성

- Player 클래스로 각 선수 인스턴스 생성
- 각 선수의 소속팀, 포지션, 나이 및 경기 통계(게임 수, 분, 득점 등) 속성으로 저장
- 이 데이터는 선수별 성적 분석과 팀 구성 파악에 활용

```
Cody_Zeller = Player('Cody_Zeller')
Cody_Zeller.hasClubTeam.append(New_Orleans_Pelicans)
Cody_Zeller.hasPosition.append(CE)
Cody_Zeller.player_Age = 31
Cody_Zeller.player_G = 14
Cody_Zeller.player_MP = 129
Cody_Zeller.player_MP = 129
Cody_Zeller.player_FG = 9
Cody_Zeller.player_FGA = 25
Cody_Zeller.player_FT = 6
Cody_Zeller.player_FTA = 16
Cody_Zeller.player_TRB = 47
Cody_Zeller.player_AST = 17
Cody_Zeller.player_STL = 2
Cody_Zeller.player_BLK = 2
Cody_Zeller.player_TOV = 4
Cody_Zeller.player_PF = 14
Cody_Zeller.player_PTS = 24
```

NBA ontology 구축

● SPARQL 예시

- SPARQL을 이용한 NBA 선수 또는 팀 분석
- SPARQL을 이용한 NBA 시즌 MVP 후보 예측도 가능함
- 이 SPARQL 쿼리는 선수의 전반적인 경기 영향력을 평가하여 MVP 후보로서의 가능성이 높은 상위 5명의 NBA 선수를 선정함

```
query = '''
    SELECT ?player ?team ?position ?age ?gamesPlayed ?points
    WHERE {
      ?player rdf:type nba:Player ;
              nba:hasClubTeam ?team ;
              nba:hasPosition ?position ;
              nba:player_Age ?age ;
              nba:player_G ?gamesPlayed ;
              nba:player_PTS ?points ;
              nba:player_TRB ?rebounds ;
              nba:player_AST ?assists ;
              nba:player_STL ?steals ;
              nba:player_BLK ?blocks ;
              nba:player_TOV ?turnovers ;
              nba:player_FGA ?fieldGoalAttempts ;
              nba:player_FG ?fieldGoals ;
              nba:player_FTA ?freeThrowAttempts ;
              nba:player_FT ?freeThrows .

      BIND(?points + ?rebounds + ?assists + ?steals + ?blocks - ?turnovers - ?fieldGoalAttempts + ?fieldGoals - ?freeThrowAttempts + ?freeT

      FILTER(?mvpProbability > 0)
    }
    ORDER BY DESC(?mvpProbability)
    LIMIT 5
'''
```

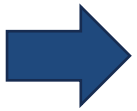
	player	team	position	age	gamesPlayed	points
0	Nikola_Jokic	Denver Nuggets	CE	28	17	490
1	Joel_Embiid	Philadelphia 76ers	CE	29	16	512
2	Giannis_Antetokounmpo	Milwaukee Bucks	PF	29	17	511
3	Luka_Doncic	Dallas Mavericks	PG	24	17	529
4	Shai_Gilgeous-Alexander	Oklahoma City Thunder	PG	25	16	488

NBA ontology 구축

● SPARQL 예시

- 이 SPARQL 쿼리는 NBA 서부 컨퍼런스와 동부 컨퍼런스 팀의 총 개수를 세는 쿼리임

```
query = '''
    SELECT (COUNT(?team) AS ?totalTeam)
    WHERE {
      {
        ?team rdf:type nba:Western_Conference .
      } UNION {
        ?team rdf:type nba:Eastern_Conference .
      }
    }
'''
```



	totalTeam
0	30

NBA ontology 구축

● SPARQL 예시

- 이 SPARQL 쿼리는 24세 이하 NBA 선수들 중 총 득점(포인트, 자유투, 필드골 합계)이 높은 상위 10명을 찾아, 선수 이름, 팀, 나이, 소속 컨퍼런스와 함께 나열함

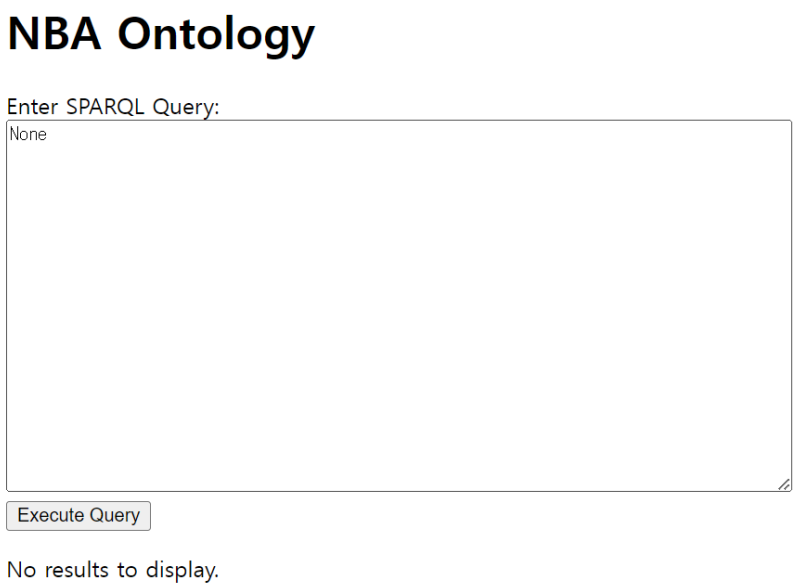
```
query = '''
    SELECT ?player ?team ?age ?conference ?totalScore
    WHERE {
      ?player rdf:type nba:Player ;
              nba:hasClubTeam ?team ;
              nba:player_Age ?age ;
              nba:player_PTS ?points ;
              nba:player_FT ?freeThrows ;
              nba:player_FG ?fieldGoals .

      OPTIONAL {
      ?team rdf:type nba:Eastern_Conference .
      BIND("East" AS ?conference)
      }

      OPTIONAL {
        ?team rdf:type nba:Western_Conference .
        BIND("West" AS ?conference)
      }

      BIND(?points + ?freeThrows + ?fieldGoals AS ?totalScore)

      FILTER(?age <= 24)
    }
    ORDER BY DESC(?totalScore)
    LIMIT 10
'''
```

	player	team	age	conference	totalScore
0	Luka_Doncic	Dallas Mavericks	24	West	811
1	Tyrese_Maxey	Philadelphia 76ers	23	East	694
2	Anthony_Edwards	Minnesota Timberwolves	22	West	692
3	Tyrese_Haliburton	Indiana Pacers	23	East	586
4	Cade_Cunningham	Detroit Pistons	22	East	584
5	LaMelo_Ball	Charlotte Hornets	22	East	556
6	Paolo_Banchero	Orlando Magic	21	East	528
7	Zion_Williamson	New Orleans Pelicans	23	West	521
8	Scottie_Barnes	Toronto Raptors	22	East	519
9	Franz_Wagner	Orlando Magic	22	East	511

Service 실행

Service 실행

- **Web**
  - flask 라이브러리를 활용한 간단한 웹 페이지 구축 (Python)
  - Input: Sparql query
  - Output: query에 대한 온톨로지 정보를 데이터프레임 형식으로 출력



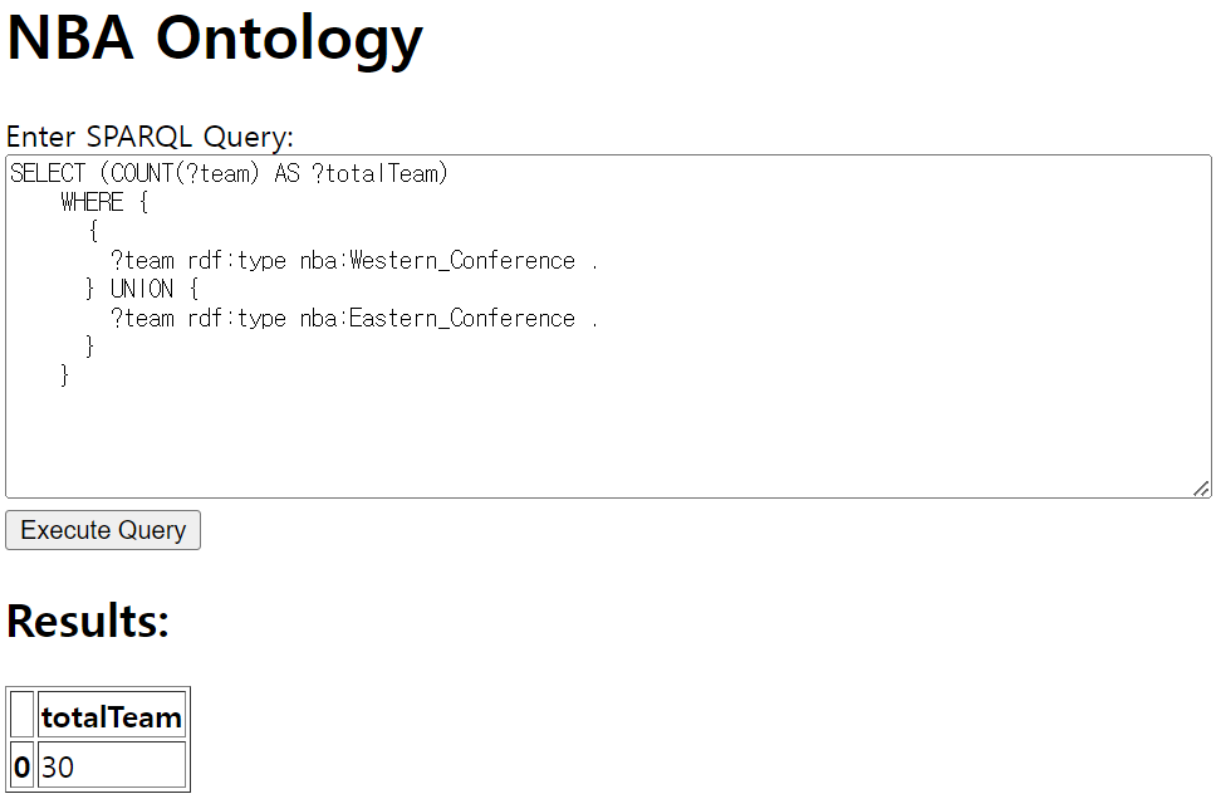
Service 실행

● SPARQL 입력

- **Q: NBA Ontology에 저장된 전체 팀 수?**

Query

```
SELECT (COUNT(?team) AS ?totalTeam
WHERE {
  {
     ?team rdf:type nba:Western_Conference .
   }  UNION {
     ?team rdf:type nba:Eastern_Conference .
   }
}
```



Service 실행

● SPARQL 입력

▪ Q: MVP 수상 확률이 높은 선수 상위 5명을 보여줘

***PTS+TRB+AST+STL+BLK-TOV-FGA+FG-FTA+FT**

***득점+리바운드+어시스트+스틸+블록-실책-야투시도+야투성공-자유투시도+자유투성공**

Query

```
SELECT ?player ?team ?position ?age ?gamesPlayed ?points
    WHERE {
      ?player rdf:type nba:Player ;
              nba:hasClubTeam ?team ;
              nba:hasPosition ?position ;
              nba:player_Age ?age ;
              nba:player_G ?gamesPlayed ;
              nba:player_PTS ?points ;
              nba:player_TRB ?rebounds ;
              nba:player_AST ?assists ;
              nba:player_STL ?steals ;
              nba:player_BLK ?blocks ;
              nba:player_TOV ?turnovers ;
              nba:player_FGA ?fieldGoalAttempts ;
              nba:player_FG ?fieldGoals ;
              nba:player_FTA ?freeThrowAttempts ;
              nba:player_FT ?freeThrows .

      BIND(?points + ?rebounds + ?assists + ?steals + ?blocks - ?turnovers -
 ?fieldGoalAttempts + ?fieldGoals - ?freeThrowAttempts + ?freeThrows AS ?mvpProbability)

      FILTER(?mvpProbability > 0)
    }
    ORDER BY DESC(?mvpProbability)
    LIMIT 5
```

NBA Ontology

Enter SPARQL Query:

```
SELECT ?player ?team ?position ?age ?gamesPlayed ?points
    WHERE {
      ?player rdf:type nba:Player ;
              nba:hasClubTeam ?team ;
              nba:hasPosition ?position ;
              nba:player_Age ?age ;
              nba:player_G ?gamesPlayed ;
              nba:player_PTS ?points ;
              nba:player_TRB ?rebounds ;
              nba:player_AST ?assists ;
              nba:player_STL ?steals ;
              nba:player_BLK ?blocks ;
              nba:player_TOV ?turnovers ;
              nba:player_FGA ?fieldGoalAttempts ;
              nba:player_FG ?fieldGoals ;
              nba:player_FTA ?freeThrowAttempts ;
              nba:player_FT ?freeThrows .

      BIND(?points + ?rebounds + ?assists + ?steals + ?blocks - ?turnovers - ?
fieldGoalAttempts + ?fieldGoals - ?freeThrowAttempts + ?freeThrows AS ?mvpProbability)

      FILTER(?mvpProbability > 0)
    }
    ORDER BY DESC(?mvpProbability)
    LIMIT 5
```

Execute Query

Results:

	player	team	position	age	gamesPlayed	points
0	Nikola_Jokic	Denver Nuggets	CE	28	17	490
1	Joel_Embiid	Philadelphia 76ers	CE	29	16	512
2	Giannis_Antetokounmpo	Milwaukee Bucks	PF	29	17	511
3	Luka_Doncic	Dallas Mavericks	PG	24	17	529
4	Shai_Gilgeous-Alexander	Oklahoma City Thunder	PG	25	16	488

Service 실행

● Service 시연

▪ Q: 24세 이하이면서 PTS + FT + FG가 높은 상위 10명에 대한 정보를 보여줘 (유망주)

```
SELECT ?player ?team ?age ?conference ?totalScore
    WHERE {
      ?player rdf:type nba:Player ;
              nba:hasClubTeam ?team ;
              nba:player_Age ?age ;
              nba:player_PTS ?points ;
              nba:player_FT ?freeThrows ;
              nba:player_FG ?fieldGoals .

      OPTIONAL {
      ?team rdf:type nba:Eastern_Conference .
      BIND("East" AS ?conference)
      }

      OPTIONAL {
        ?team rdf:type nba:Western_Conference .
        BIND("West" AS ?conference)
      }

      BIND(?points + ?freeThrows + ?fieldGoals AS ?totalScore)

      FILTER(?age <= 24)
    }
    ORDER BY DESC(?totalScore)
    LIMIT 10
```

Service 실행

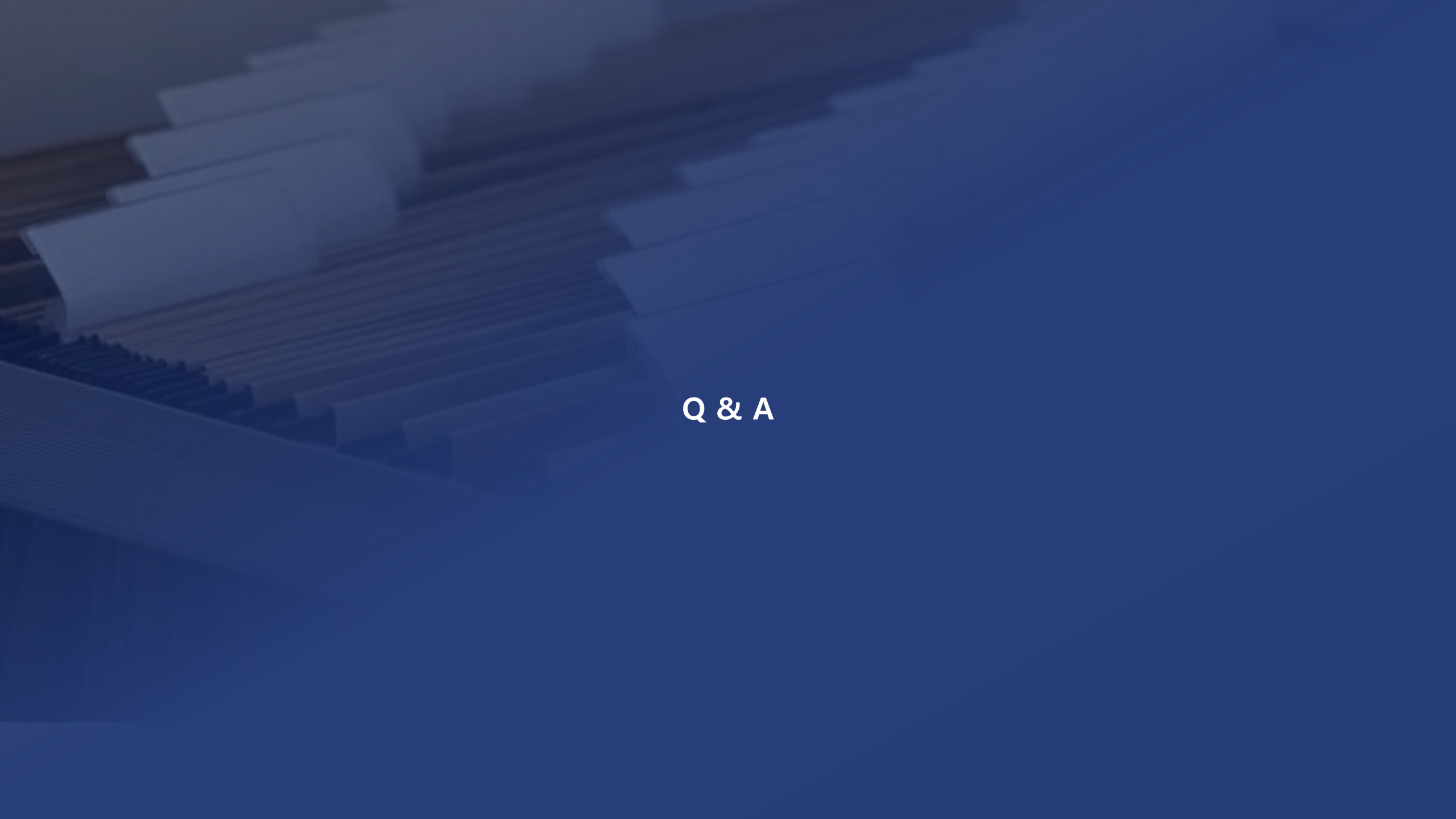
● Service 시연

▪ Q: 동부, 서부 별로 승률이 높은 상위 3개 팀을 각각 보여줘

```
# 동부
SELECT ?team ?winNum ?gameNum ?roundedWinPercentage
    WHERE {
      ?team rdf:type nba:Eastern_Conference ;
            nba:hasWin_num ?winNum ;
            nba:hasGame_num ?gameNum .

      BIND(?winNum / ?gameNum AS ?winPercentage)
      BIND(FLOOR(?winPercentage * 100) / 100
AS ?roundedWinPercentage)
    }
    ORDER BY DESC(?winPercentage)
    LIMIT 3
```

```
# 서부
SELECT ?team ?winNum ?gameNum ?roundedWinPercentage
    WHERE {
      ?team rdf:type nba:Western_Conference ;
            nba:hasWin_num ?winNum ;
            nba:hasGame_num ?gameNum .

      BIND(?winNum / ?gameNum AS ?winPercentage)
      BIND(FLOOR(?winPercentage * 100) / 100
AS ?roundedWinPercentage)
    }
    ORDER BY DESC(?winPercentage)
    LIMIT 3
```

감사합니다.


Q & A