R Markdown

Dit is het begin van ons RMD bestand voor de capstone!

Including libraries

```
if (!require("dplyr")) install.packages("dplyr")
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
if (!require("RMySQL")) install.packages("RMySQL")
```

```
## Loading required package: RMySQL
```

```
## Loading required package: DBI
```

```
if (!require("CityPlot")) install.packages("CityPlot")
```

```
## Loading required package: CityPlot
```

```
library(dplyr)
library(RMySQL)
library(CityPlot)
```

Importing data using mySQL

Waar bestaat de dataset uit?

```
my_db <- dbConnect(MySQL(),
                   user='klaasberbee',
                   password='1190KVU',
                   dbname='Batting',
                   host='localhost')

print(dbListTables(my_db))
```

```
##  [1] "AllstarFull"         "Appearances"         "AwardsManagers"
##  [4] "AwardsPlayers"       "AwardsShareManagers" "AwardsSharePlayers"
##  [7] "Batting"             "BattingPost"         "CollegePlaying"
## [10] "Fielding"            "FieldingOF"          "FieldingOFsplit"
## [13] "FieldingPost"        "HallOfFame"          "HomeGames"
## [16] "Managers"            "ManagersHalf"        "Master"
## [19] "Parks"               "Pitching"            "PitchingPost"
## [22] "Salaries"            "Schools"             "SeriesPost"
## [25] "Teams"               "TeamsFranchises"     "TeamsHalf"
```

```
dbTables <- dbListTables(my_db)

allTables <-
  lapply(dbTables, function(table) {
    dbGetQuery(my_db, paste("select * from", table))
  })
names(allTables) <- paste("db", dbTables, sep = "_")
list2env(allTables, envir = .GlobalEnv)
```

```
## <environment: R_GlobalEnv>
```

Uit deze datasets kan alle data gevonden worden over de de gehele teams, maar ook over de individuele spelers en managers, zoals hun salaris en hun stats. Verder wordt er weergegeven hoe een game verloopt en wat voor activiteiten er in voor komen en door wie deze zijn uitgevoert. Als laatst kan er ook uit de datasets opgemaakt worden hoeveel een speler verdiend.

Het doel is om deze dateset om te zetten naar informatie en dit dan vervolgens om te zetten naar kennis. Er kan namelijk met al deze data kennis worden opgebouwt zoals welke factoren zijn belangrijk voor een sterk team of wat maakt een bepaalde speler zo goed.

belanghebbende voor deze dataset: gokkers: Wanneer gokkers een voorspelmodel kunnen maken van deze dataset met wie er wint, dan kunnen zij er veel geld mee verdienen op goksites.

De coaches van de team: De coach kan uit deze data set halen wat de sterke en de zwake punten zijn van zijn spelers, hierdoor weet hij welke skils hij bij bepaalde spelers moet verbeteren. Ook is deze dataset handig voor hem omdat hij de zwake en de sterke punten van de tegenstander weet. Op deze mannier kan hij de inzet van zijn spelers inzetten dat hij er een voordeel uit kan halen.

Spelers die op zoek zijn naar een opleidings schoool: De spelers die op zoek zijn naar een school waar ze hun basebal skils willen verbeteren kunnen uit deze dataset opmaken waar welke speler is opgeleid. Uit de dataset kan mischien ook wel opgemaakt worden welke skils je bij welke school goed leert. Dit zou dan kunnen helpen bij het maken van zijn beslissing.

de raad van een stadion: Uit deze dataset kan opgemaakt worden hoeveel kijkers er komen kijken bij een wedstrijd. Bij de ene wedstrijd komen er namelijk meer dan bij een ander. Dit wetende kan de raad van een stadion bijvoorbeeld zijn hoeveelheid personeel hierop aanpassen of de mate van beveiliging.

Doormiddel van de onderstaande code kan er uitgezocht worden wat de Primary Keys zijn.

```
db_AllstarFull[(duplicated(db_AllstarFull[c("playerID","gameID")])
                 | duplicated(db_AllstarFull[c("playerID","gameID")], fromLast = TRUE)), ]
```

```
## [1] playerID    yearID      gameNum     gameID      teamID      lgID
## [7] GP          startingPos
## <0 rows> (or 0-length row.names)
```

```
db_Appearances[(duplicated(db_Appearances[c("playerID","yearID","teamID")])
                 | duplicated(db_Appearances[c("playerID","yearID", "teamID")], fromLast = TRUE)), ]
```

```
##  [1] yearID    teamID    lgID      playerID  G_all     GS        G_batting
##  [8] G_defense G_p       G_c       G_1b      G_2b      G_3b      G_ss
## [15] G_lf      G_cf      G_rf      G_of      G_dh      G_ph      G_pr
## <0 rows> (or 0-length row.names)
```

```
db_AwardsManagers[(duplicated(db_AwardsManagers[c("playerID","awardID", "yearID")])
                 | duplicated(db_AwardsManagers[c("playerID","awardID", "yearID")], fromLast = TRUE)),]
```

```
## [1] playerID awardID  yearID   lgID     tie      notes
## <0 rows> (or 0-length row.names)
```

```
db_AwardsPlayers[(duplicated(db_AwardsPlayers[c("playerID","awardID", "yearID", "lgID")])
                 | duplicated(db_AwardsPlayers[c("playerID","awardID", "yearID", "lgID")], fromLast = T
```

```
## [1] playerID awardID  yearID   lgID     tie      notes
## <0 rows> (or 0-length row.names)
```

```
db_AwardsShareManagers[(duplicated(db_AwardsShareManagers[c("playerID", "yearID")])
                 | duplicated(db_AwardsShareManagers[c("playerID", "yearID")], fromLast = TRUE)), ]
```

```
## [1] awardID    yearID     lgID       playerID   pointsWon  pointsMax
## [7] votesFirst
## <0 rows> (or 0-length row.names)
```

```
db_AwardsSharePlayers[(duplicated(db_AwardsSharePlayers[c("playerID", "yearID", "awardID")])
                 | duplicated(db_AwardsSharePlayers[c("playerID", "yearID", "awardID")], fromLast = TRU
```

```
## [1] awardID    yearID     lgID       playerID   pointsWon  pointsMax
## [7] votesFirst
## <0 rows> (or 0-length row.names)
```

```
db_Batting[(duplicated(db_Batting[c("playerID", "yearID", "stint")])
                 | duplicated(db_Batting[c("playerID", "yearID", "stint")], fromLast = TRUE)), ]
```

```
##  [1] playerID yearID   stint    teamID   lgID     G        AB
##  [8] R        H        2B       3B       HR       RBI      SB
## [15] CS       BB       SO       IBB      HBP      SH       SF
## [22] GIDP
## <0 rows> (or 0-length row.names)
```

```
db_BattingPost[(duplicated(db_BattingPost[c("playerID", "yearID", "round")])
                 | duplicated(db_BattingPost[c("playerID", "yearID", "round")], fromLast = TRUE)), ]
```

```
##  [1] yearID   round    playerID teamID   lgID     G        AB
##  [8] R        H        2B       3B       HR       RBI      SB
## [15] CS       BB       SO       IBB      HBP      SH       SF
## [22] GIDP
## <0 rows> (or 0-length row.names)
```

```
db_CollegePlaying[(duplicated(db_CollegePlaying[c("playerID", "yearID", "schoolID")])
                 | duplicated(db_CollegePlaying[c("playerID", "yearID", "schoolID")], fromLast = TRUE))
```

```
## [1] playerID schoolID yearID
## <0 rows> (or 0-length row.names)
```

```
db_Fielding[(duplicated(db_Fielding[c("playerID", "yearID", "stint", "POS")])
                 | duplicated(db_Fielding[c("playerID", "yearID", "stint", "POS")], fromLast = TRUE)),]
```

```
##  [1] playerID yearID   stint    teamID   lgID     POS      G
##  [8] GS       InnOuts  PO       A        E        DP       PB
## [15] WP       SB       CS       ZR
## <0 rows> (or 0-length row.names)
```

```
db_FieldingOF[(duplicated(db_FieldingOF[c("playerID", "yearID", "stint")])
                 | duplicated(db_FieldingOF[c("playerID", "yearID", "stint")], fromLast = TRUE)),]
```

```
## [1] playerID yearID   stint    Glf      Gcf      Grf
## <0 rows> (or 0-length row.names)
```

```
db_FieldingOFsplit[(duplicated(db_FieldingOFsplit[c("playerID", "yearID", "stint", "POS")])
                 | duplicated(db_FieldingOFsplit[c("playerID", "yearID", "stint", "POS")], fromLast = T
```

```
##  [1] playerID yearID   stint    teamID   lgID     POS      G
##  [8] GS       InnOuts  PO       A        E        DP       PB
## [15] WP       SB       CS       ZR
## <0 rows> (or 0-length row.names)
```

```
db_FieldingPost[(duplicated(db_FieldingPost[c("playerID", "yearID", "round", "POS")])
                 | duplicated(db_FieldingPost[c("playerID", "yearID", "round", "POS")], fromLast = TRUE
```

```
##  [1] playerID yearID   teamID   lgID     round    POS      G
##  [8] GS       InnOuts  PO       A        E        DP       TP
## [15] PB       SB       CS
## <0 rows> (or 0-length row.names)
```

```
db_HallOfFame[(duplicated(db_HallOfFame[c("playerID", "yearid", "votedBy")])
                 | duplicated(db_HallOfFame[c("playerID", "yearid", "votedBy")], fromLast = TRUE)),]
```

```
## [1] playerID    yearid      votedBy     ballots     needed      votes
## [7] inducted    category    needed_note
## <0 rows> (or 0-length row.names)
```

```
db_HomeGames[(duplicated(db_HomeGames[c("year.key", "team.key", "park.key")])
                 | duplicated(db_HomeGames[c("year.key", "team.key", "park.key")], fromLast = TRUE)),]
```

```
## [1] year.key   league.key team.key   park.key   span.first span.last
## [7] games      openings   attendance
## <0 rows> (or 0-length row.names)
```

```
db_Managers[(duplicated(db_Managers[c("playerID", "yearID", "inseason")])
                 | duplicated(db_Managers[c("playerID", "yearID", "inseason")], fromLast = TRUE)),]
```

```
##  [1] playerID yearID   teamID   lgID     inseason G        W
##  [8] L        rank     plyrMgr
## <0 rows> (or 0-length row.names)
```

```
db_ManagersHalf[(duplicated(db_ManagersHalf[c("playerID","yearID","half")])
      | duplicated(db_ManagersHalf[c("playerID","yearID","half")], fromLast = TRUE)), ]
```

```
##  [1] playerID yearID   teamID   lgID     inseason half     G
##  [8] W        L        rank
## <0 rows> (or 0-length row.names)
```

```
db_Master[(duplicated(db_Master["playerID"])
                       | duplicated(db_Master["playerID"], fromLast = TRUE)), ]
```

```
##  [1] playerID     birthYear    birthMonth   birthDay     birthCountry
##  [6] birthState   birthCity    deathYear    deathMonth   deathDay
## [11] deathCountry deathState   deathCity    nameFirst    nameLast
## [16] nameGiven    weight       height       bats         throws
## [21] debut        finalGame    retroID      bbrefID
## <0 rows> (or 0-length row.names)
```

```
db_Parks[(duplicated(db_Parks["park.key"])
       | duplicated(db_Parks["park.key"], fromLast = TRUE)), ]
```

```
## [1] park.key   park.name  park.alias city       state      country
## <0 rows> (or 0-length row.names)
```

```
db_Pitching[(duplicated(db_Pitching[c("playerID", "yearID", "stint")])
          | duplicated(db_Pitching[c("playerID", "yearID", "stint")], fromLast = TRUE)), ]
```

```
##  [1] playerID yearID   stint    teamID   lgID     W        L
##  [8] G        GS       CG       SHO      SV       IPouts   H
## [15] ER       HR       BB       SO       BAOpp    ERA      IBB
## [22] WP       HBP      BK       BFP      GF       R        SH
## [29] SF       GIDP
## <0 rows> (or 0-length row.names)
```

```
db_PitchingPost[(duplicated(db_PitchingPost[c("playerID", "yearID", "round")])
              | duplicated(db_PitchingPost[c("playerID", "yearID", "round")], fromLast = TRUE)), ]
```

```
##  [1] playerID yearID   round    teamID   lgID     W        L
##  [8] G        GS       CG       SHO      SV       IPouts   H
## [15] ER       HR       BB       SO       BAOpp    ERA      IBB
## [22] WP       HBP      BK       BFP      GF       R        SH
## [29] SF       GIDP
## <0 rows> (or 0-length row.names)
```

```
db_Salaries[(duplicated(db_Salaries[c("playerID", "yearID", "teamID")])
          | duplicated(db_Salaries[c("playerID", "yearID", "teamID")], fromLast = TRUE)), ]
```

```
## [1] yearID   teamID   lgID     playerID salary
## <0 rows> (or 0-length row.names)
```

```
db_Schools[(duplicated(db_Salaries["schoolID", ])
         | duplicated(db_Salaries["schoolID", ], fromLast = TRUE)), ]
```

```
## [1] schoolID  name_full city      state     country
## <0 rows> (or 0-length row.names)
```

```
db_SeriesPost[(duplicated(db_SeriesPost[c("yearID", "round")])
       | duplicated(db_SeriesPost[c("yearID", "round")], fromLast = TRUE)), ]
```

```
## [1] yearID       round        teamIDwinner lgIDwinner   teamIDloser
## [6] lgIDloser    wins         losses       ties
## <0 rows> (or 0-length row.names)
```

```
db_Teams[(duplicated(db_Teams[c("yearID", "teamID")])
                 | duplicated(db_Teams[c("yearID", "teamID")], fromLast = TRUE)), ]
```

```
##  [1] yearID         lgID           teamID         franchID
##  [5] divID          Rank           G              Ghome
##  [9] W              L              DivWin         WCWin
## [13] LgWin          WSWin          R              AB
## [17] H              2B             3B             HR
## [21] BB             SO             SB             CS
## [25] HBP            SF             RA             ER
## [29] ERA            CG             SHO            SV
## [33] IPouts         HA             HRA            BBA
## [37] SOA            E              DP             FP
## [41] name           park           attendance     BPF
## [45] PPF            teamIDBR       teamIDlahman45 teamIDretro
## <0 rows> (or 0-length row.names)
```

```
db_TeamsFranchises[(duplicated(db_TeamsFranchises[c("franchID")])
                 | duplicated(db_TeamsFranchises[c("franchID")], fromLast = TRUE)), ]
```

```
## [1] franchID   franchName active     NAassoc
## <0 rows> (or 0-length row.names)
```

```
db_TeamsHalf[(duplicated(db_TeamsHalf[c("teamID", "Half")])
             | duplicated(db_TeamsHalf[c("teamID", "Half")], fromLast = TRUE)), ]
```

```
##  [1] yearID lgID   teamID Half   divID  DivWin Rank   G      W      L
## <0 rows> (or 0-length row.names)
```

```
glimpse(db_AllstarFull)
```

```
## Observations: 5,148
## Variables: 8
## $ playerID    <chr> "gomezle01", "ferreri01", "gehrilo01", "gehrich01"...
## $ yearID      <int> 1933, 1933, 1933, 1933, 1933, 1933, 1933, 1933, 19...
## $ gameNum     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ gameID      <chr> "ALS193307060", "ALS193307060", "ALS193307060", "A...
## $ teamID      <chr> "NYA", "BOS", "NYA", "DET", "CHA", "WS1", "NYA", "...
## $ lgID        <chr> "AL", "AL", "AL", "AL", "AL", "AL", "AL", "AL", "A...
## $ GP          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0,...
## $ startingPos <chr> "1", "2", "3", "4", "5", "6", "7", "8", "9", "", "...
```