Alonso\_Project 2 Deliverable

Martin Alonso

June 27, 2018

# 1. Summary and Business Case

This project looks to create a pitch usage database that tracks the types of pitches thrown by professional baseball players as well as the events surrounding their decisions to use certain pitches, while also monitoring the movement of these pitches, starting from the release point out of the pitchers hand up to the moment the pitch reaches the plate. The importance of creating such a database will let us better understand the physics behind each pitch thrown by major league pitchers, as well as understand the effects of pitch sequencing and game theory regarding individual pitcher-batter matchups and decisions made.  
As such, we first need to track every game that has been played. Every game will have multiple events, but each event must be assigned to one game. Each event will involve two teams, a fielding team with a pitcher and a batting team with a batter involved. Only one batter and one pitcher can be involved in one single event, though there can be multiple separate events involving the same pitcher and batter. Because of this, each event must occur sequentially, tracking the inning, outs, plate appearance, pitch used, pitch of the sequence, result of the pitch (ball, foul, or strike), outcome (strike out, hit, walk, etc.), men on base (first, second, third), the score, and the plate appearance of the inning. This must be done for every pitch thrown in every single game.  
Each pitch must be assigned to a pitch type and each pitch can only have one pitch type, but there can be multiple pitches with different pitch types. Each pitch must also be related to a single pitcher, though a pitcher can throw multiple pitches of different pitch types. Each pitch thrown must also be linked to one and only one event. However, the pitch must also be linked to the pitch of the next event, so we can analyze pitch sequencing. Each pitch within the first event must have a distinct release point, trajectory, and movement (both on the x and z plane). Likewise, each pitch will end in one result, but many results can be traced backed to each pitch thrown.  
On the player side, each game will feature players from two teams. Only two teams may play in any single game, and at least nine players from each team must be present. However, if the home team is from the American League, at least ten players will have to appear (given the DH-rule of the American League.) We must also know, for each team, what position each player has on the field (using baseball’s position guide, 1 through 10.) Each position must be played by a player on each team, but, given pitching changes, pinch hitters, pinch runners, defensive replacements, and injuries, many players could end up playing a single position in any given game.

# 2. Conceptual Model

The database will need at least six tables that will work together to provide the necessary information for each plate appearance and every pitch that has been thrown throughout the seasons.  
The Game table will provide the GameID, the date a game was played, and the Home and Away teams.  
Both the HomeTeamID and the AwayTeamID will be fed from the Team table, which will have the TeamID, TeamCity, TeamName, and LeagueID, identifying the different teams that have played.  
The GameID will work as the first foreign key that will help identify the different events that have occurred within a game. These events will be found in the Events table. Within this table, we will have the pitcherID, batterID, inning, half\_inning (top or bot), and plate\_appearance, provinding us with the context of every plate appearance that has occurred within the game. Similarly, the event will also have the pitchID of every pitch thrown in the plate appearance, along with the pitch\_type, the pitch\_of\_pa, the result of the pitch (whether it was a ball or a strike), and the outcome (which will be ball and strike for every pitch that doesn’t end the plate appearance; for the rest, it will show what happened). The events table will also show, through the use of boolean values, whether there were men on first, second, or third, the count (by adding the previous pitches thrown), the outs, and the scores for both home and away teams.  
Both the pitcherID and batterID will be linked to the Player table, which will provide the information for each player: his first\_name, last\_name, birth\_date, height\_in (in inches), weight\_lbs (in pounds), TeamID (which will be linked to the Team table), season, and positionID for the main position the player played.  
The positionID will be linked to the Position table where the names of each position will be taken from. PositionID will be an integer that goes from one to ten, using the standard notation for every baseball position (1 – pitcher, 2 – catcher, 3 – first base, 4 – second base, 5 – third base, 6 – shortstop, 7 – leftfield, 8 – centerfield, 9 – rightfield, 10 – designated hitter).  
Going back to the events table, this table also features pitchID, which is a foreign key that will merge the Event table to the Pitch table. Within the Pitch table, we will find the pitchID and the pitch\_type of the pitch thrown. We will also find the pitcherID (which will link from the Player table) and the eventID (which will link from the Event table). Finally, we will have the information regarding the movement of the pitch: release\_point, velocity, x\_movement, z\_movement, max\_plate\_dist, max\_plate\_time, and next\_pitch (which will be the pitchID of the next pitch thrown in the sequence). This will help us identify pitch sequencing used by the different pitchers.

# 3. Normalization

# 4. Physical Database Design

/\*  
 Author: Martin Alonso  
 Course: IST659 Project  
 Term: April 2018  
 \*/  
   
 --Create the League table  
 CREATE TABLE League (  
 --Columns for the League table  
 LeagueID char(2) not null,  
 LeagueName char(30) not null,  
 CONSTRAINT PK\_League PRIMARY KEY (LeagueID)  
 );  
 --End Creating the League table   
   
 --Create the Team table  
 CREATE TABLE Team (  
 --Columns for the Team table  
 TeamID char(3) not null,  
 TeamCity varchar(20) not null,   
 TeamName varchar(20) not null,  
 LeagueID char(2) not null,  
 --Constraints on the Team table  
 CONSTRAINT PK\_Team PRIMARY KEY (TeamID),  
 CONSTRAINT FK1\_Team FOREIGN KEY (LeagueID) REFERENCES League(LeagueID)  
 );  
 --End Creating the Team table  
   
 --Create the Game table  
 CREATE TABLE Game (  
 --Columns for the Game table  
 GameNumber int identity,   
 GameID char(25) not null,  
 Game\_Date datetime not null,   
 HomeTeamID char(3) not null,  
 AwayTeamID char(3) not null,  
 --Constraints on the Game table  
 CONSTRAINT PK\_Game PRIMARY KEY (GameID),  
 CONSTRAINT FK1\_Game FOREIGN KEY (HomeTeamID) REFERENCES Team(TeamID),  
 CONSTRAINT FK2\_Game FOREIGN KEY (AwayTeamID) REFERENCES Team(TeamID)  
 );  
 --End Creating the Game table  
   
 --Create the Positions table  
 CREATE TABLE Positions (  
 --Columns for the Positions table  
 positionID int not null,  
 position\_short char(2) not null,  
 position\_name char(20) not null,  
 --Constraints on the Positions table  
 CONSTRAINT PK\_Positions PRIMARY KEY (positionID)  
 );  
 --End Creating the Positions table  
   
 --Create the Player table  
 CREATE TABLE Player (  
 --Columns for the Player table  
 playerID int identity,   
 first\_name varchar(30) not null,   
 last\_name varchar(30) not null,  
 birth\_date date not null,  
 height\_in int,  
 weight\_lbs int,  
 TeamID char(3),  
 season int,   
 positionID int,  
 --Constraints on the Player table  
 CONSTRAINT PK\_Player PRIMARY KEY (playerID),  
 CONSTRAINT FK1\_Player FOREIGN KEY (TeamID) REFERENCES Team(TeamID),  
 CONSTRAINT FK2\_Player FOREIGN KEY (positionID) REFERENCES Positions(positionID)  
 );  
 --End Creating the Player table  
   
 --Create the Plate\_Appearance table  
 CREATE TABLE Plate\_Appearance (  
 --Columns for the Plate\_Appearance table  
 plate\_appearance\_id int identity,   
 GameID char(25) not null,  
 pitcherID int not null,   
 batterID int not null,  
 inning int not null,  
 half\_inning char(3) not null,  
 plate\_appearance int not null,  
 runner\_1b bit,  
 runner\_2b bit,   
 runner\_3b bit,  
 outs int not null,   
 home\_score int not null,  
 away\_score int not null,  
 --Constraints on the Plate\_Appearance table  
 CONSTRAINT PK\_Plate\_Appearance PRIMARY KEY (plate\_appearance\_id),  
 CONSTRAINT FK1\_Plate\_Appearance FOREIGN KEY (GameID) REFERENCES Game(GameID),  
 CONSTRAINT FK2\_Plate\_Appearance FOREIGN KEY (pitcherID) REFERENCES Player(playerID),  
 CONSTRAINT FK3\_Plate\_Appearance FOREIGN KEY (batterID) REFERENCES Player(playerID)  
 );  
 --End Creating the Plate\_Appearance table  
   
 --Create the Pitch\_event table  
 CREATE TABLE Pitch\_event (  
 --Columns for the Pitch\_event table  
 pitchID int identity,   
 GameID char(25) not null,  
 plate\_appearance\_id int not null,  
 pitch\_of\_pa int not null,   
 pitch\_type char(2) not null,  
 result varchar(10) not null,  
 outcome varchar(20),  
 balls int not null,  
 strikes int not null,  
 play\_description varchar(100),  
 --Constraints on the Pitch\_event table  
 CONSTRAINT PK\_Pitch\_event PRIMARY KEY (pitchID),  
 CONSTRAINT FK1\_Pitch\_event FOREIGN KEY (GameID) REFERENCES Game(GameID),  
 CONSTRAINT FK2\_Pitch\_event FOREIGN KEY (plate\_appearance\_id) REFERENCES Plate\_Appearance(plate\_appearance\_id)   
 );  
 --End Creating the Pitch\_event table  
   
 --Create the Pitch\_type table  
 CREATE TABLE Pitch\_type (  
 --Columns for Pitch\_type table  
 Pitch\_type char(2) not null,   
 Pitch\_name char(20) not null,  
 --Constraints on the Pitch\_type table  
 CONSTRAINT PK\_Pitch\_type PRIMARY KEY (Pitch\_type)  
 );  
 --End Creating the Pitch\_type table  
   
 --Create the Pitch table  
 CREATE TABLE Pitch (  
 --Columns for the Pitch table  
 pitchID int not null,  
 pitch\_type char(2) not null,  
 release\_point float not null,   
 velocity float not null,  
 x\_movement float not null,  
 z\_movement float not null,  
 max\_plate\_dist float not null,  
 max\_plate\_time float not null,  
 next\_pitchID int not null,  
 --Constraints on the Pitch table  
 CONSTRAINT PK\_Pitch PRIMARY KEY (pitchID),  
 CONSTRAINT FK1\_Pitch FOREIGN KEY (pitchID) REFERENCES Pitch\_event(pitchID),  
 CONSTRAINT FK2\_Pitch FOREIGN KEY (pitch\_type) REFERENCES Pitch\_type(Pitch\_type)  
 );  
 --End Creating the Pitch table  
   
 /\*  
 drop table pitch;  
 drop table pitch\_type;  
 drop table Pitch\_event;  
 drop table Plate\_Appearance;  
 drop table Player;  
 drop table Positions;  
 drop table Game;  
 drop table Team;  
 drop table League;  
 \*/  
   
 /\*Removed GameEvent table as it was the same as Pitch\_event table sans the GameID column.   
 Game, Pitch, and Plate\_Appearance now reference each other without GameEvent table.\*/

# 5. Data Creation

--Insert data into League table   
 INSERT INTO League  
 (LeagueID, LeagueName)  
 VALUES  
 ('AL', 'American League'),  
 ('NL', 'National League');  
   
 --Verify that the data has been inserted successfully  
 SELECT \* FROM League;  
   
 --Insert Team data into Team table  
 INSERT INTO Team  
 (TeamID, TeamCity, TeamName, LeagueID)  
 VALUES  
 ('ARI', 'Arizona', 'Diamondbacks', 'NL'),  
 ('ATL', 'Atlanta', 'Braves', 'NL'),  
 ('BAL', 'Baltimore', 'Orioles', 'AL'),  
 ('BOS', 'Boston', 'Red Sox', 'AL'),  
 ('CHA', 'Chicago', 'White Sox', 'AL'),  
 ('CHN', 'Chicago', 'Cubs', 'NL'),  
 ('CIN', 'Cincinnati', 'Reds', 'NL'),  
 ('CLE', 'Cleveland', 'Indians', 'AL'),  
 ('COL', 'Colorado', 'Rockies', 'NL'),  
 ('DET', 'Detroit', 'Tigers', 'AL'),  
 ('HOU', 'Houston', 'Astros', 'AL'),  
 ('KCA', 'Kansas City', 'Royals', 'AL'),  
 ('LAA', 'Los Angeles', 'Angels', 'AL'),  
 ('LAN', 'Los Angeles', 'Dodgers', 'NL'),  
 ('MIA', 'Miami', 'Marlins', 'NL'),  
 ('MIL', 'Milwaukee', 'Brewers', 'NL'),  
 ('MIN', 'Minnesota', 'Twins', 'AL'),  
 ('OAK', 'Oakland', 'Athletics', 'AL'),  
 ('PHI', 'Philadelphia', 'Phillies', 'NL'),  
 ('PIT', 'Pittsburgh', 'Pirates', 'NL'),  
 ('NYA', 'New York', 'Yankees', 'AL'),  
 ('NYN', 'New York', 'Mets', 'NL'),  
 ('SDN', 'San Diego', 'Padres', 'NL'),  
 ('SEA', 'Seattle', 'Mariners', 'AL'),  
 ('SFN', 'San Francisco', 'Giants', 'NL'),  
 ('SLN', 'St. Louis', 'Cardinals', 'NL'),  
 ('TBA', 'Tampa Bay', 'Rays', 'AL'),  
 ('TEX', 'Texas', 'Rangers', 'AL'),  
 ('TOR', 'Toronto', 'Blue Jays', 'AL'),  
 ('WAS', 'Washington', 'Nationals', 'NL');  
   
 --Verify that the data has been inserted  
 SELECT \* FROM Team;  
   
 --Insert data into Game table  
 INSERT INTO Game  
 (GameID, Game\_Date, HomeTeamID, AwayTeamID)  
 VALUES  
 /\*Opening Day games.   
 PIT @ DET and WAS @ CIN not included on account of games being postponed.\*/  
 ('2018\_03\_29\_MIA\_CHN\_0', '2018-03-29', 'MIA', 'CHN'),  
 ('2018\_03\_29\_NYN\_SLN\_0', '2018-03-29', 'NYN', 'SLN'),  
 ('2018\_03\_29\_BAL\_MIN\_0', '2018-03-29', 'BAL', 'MIN'),  
 ('2018\_03\_29\_TEX\_HOU\_0', '2018-03-29', 'TEX', 'HOU'),  
 ('2018\_03\_29\_TOR\_NYA\_0', '2018-03-29', 'TOR', 'NYA'),  
 ('2018\_03\_29\_TBA\_BOS\_0', '2018-03-29', 'TBA', 'BOS'),  
 ('2018\_03\_29\_OAK\_LAA\_0', '2018-03-29', 'OAK', 'LAA'),  
 ('2018\_03\_29\_SDN\_MIL\_0', '2018-03-29', 'SDN', 'MIL'),  
 ('2018\_03\_29\_ATL\_PHI\_0', '2018-03-29', 'ATL', 'PHI'),  
 ('2018\_03\_29\_KCA\_CHA\_0', '2018-03-29', 'KCA', 'CHA'),  
 ('2018\_03\_29\_LAN\_SFN\_0', '2018-03-29', 'LAN', 'SFN'),  
 ('2018\_03\_29\_SEA\_CLE\_0', '2018-03-29', 'SEA', 'CLE'),  
 ('2018\_03\_29\_ARI\_CLE\_0', '2018-03-29', 'ARI', 'COL');  
   
 --View Game table  
 SELECT \* FROM Game;  
   
 --Insert data into Positions table  
 INSERT INTO Positions  
 (positionID, position\_short, position\_name)  
 VALUES  
 ('1', 'P', 'Pitcher'),  
 ('2', 'C', 'Catcher'),  
 ('3', '1B', 'First Base'),  
 ('4', '2B', 'Second Base'),  
 ('5', '3B', 'Third Base'),  
 ('6', 'SS', 'Shortstop'),  
 ('7', 'LF', 'Leftfield'),  
 ('8', 'CF', 'Centerfield'),  
 ('9', 'RF', 'Rightfield'),  
 ('10', 'DH', 'Designated Hitter');  
   
 --View Positions table  
 SELECT \* FROM Positions;  
   
 --Insert data into Player table  
 INSERT INTO Player  
 (first\_name, last\_name, birth\_date, height\_in, weight\_lbs, TeamID, season, positionID)  
 VALUES  
 --MIA Opening Day lineup (only starting 9; no relievers, pinch hitters, or pinch runners included.)  
 ('Lewis', 'Brinson', '1994-05-08', 75, 195, 'MIA', 2018, '8'),  
 ('Derek', 'Dietrich', '1989-07-18', 72, 205, 'MIA', 2018,'7'),  
 ('Starlin', 'Castro', '1990-05-07', 74, 230, 'MIA', 2018, '4'),  
 ('Justin', 'Bour', '1988-05-28', 75, 265, 'MIA', 2018, '3'),  
 ('Brian', 'Anderson', '1993-05-19', 75, 185, 'MIA', 2018, '5'),  
 ('Garrett', 'Cooper', '1990-12-25', 78, 230, 'MIA', 2018, '9'),  
 ('Miguel', 'Rojas', '1989-02-24', 71, 195, 'MIA', 2018, '6'),  
 ('Chad', 'Wallach', '1991-11-04', 75, 230, 'MIA', 2018, '2'),  
 ('Jose', 'Urena', '1991-09-12', 74, 200, 'MIA', 2018, '1'),  
 --CHN Opening Day lineup (only starting 9; no relievers, pinch hitters, or pinch runners included.)  
 ('Ian', 'Happ', '1994-08-12', 72, 205, 'CHN', 2018, '8'),  
 ('Kris', 'Bryant', '1992-01-04', 75, 230, 'CHN', 2018,'5'),  
 ('Anthony', 'Rizzo', '1989-08-08', 75, 240, 'CHN', 2018, '3'),  
 ('Willson', 'Contreras', '1992-05-13', 73, 210, 'CHN', 2018, '2'),  
 ('Kyle', 'Schwarber', '1993-03-05', 72, 235, 'CHN', 2018, '7'),  
 ('Addison', 'Russell', '1994-01-23', 72, 200, 'CHN', 2018, '6'),  
 ('Jason', 'Heyward', '1989-08-09', 77, 240, 'CHN', 2018, '9'),  
 ('Javier', 'Baez', '1992-12-01', 72, 190, 'CHN', 2018, '4'),  
 ('Jon', 'Lester', '1984-01-07', 76, 240, 'CHN', 2018, '1');  
   
 --View Player table  
 SELECT \* FROM Player;  
   
 --Insert data into Plate\_Appearance table  
 INSERT INTO Plate\_Appearance  
 (GameID, pitcherID, batterID, inning, half\_inning, plate\_appearance, runner\_1b, runner\_2b, runner\_3b, outs, home\_score, away\_score)  
 VALUES  
 --CHN top of the first inning  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 10, 1, 'top', 1, 0, 0, 0, 0, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 11, 1, 'top', 2, 1, 0, 0, 0, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 12, 1, 'top', 3, 1, 1, 0, 0, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 13, 1, 'top', 4, 1, 1, 0, 1, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 14, 1, 'top', 5, 1, 1, 0, 2, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 15, 1, 'top', 6, 0, 1, 1, 2, 0, 1),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 16, 1, 'top', 7, 1, 1, 1, 2, 0, 2),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 17, 1, 'top', 8, 1, 1, 1, 2, 0, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 18, 1, 'top', 9, 1, 1, 1, 3, 0, 3),  
 --MIA bot of the first inning   
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 1, 1, 'bot', 1, 0, 0, 0, 1, 0, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 2, 1, 'bot', 2, 0, 0, 0, 2, 0, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 3, 1, 'bot', 3, 1, 0, 0, 2, 0, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 4, 1, 'bot', 4, 1, 1, 0, 2, 0, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 5, 1, 'bot', 5, 1, 1, 0, 2, 1, 3),  
 ('2018\_03\_29\_MIA\_CHN\_0', 18, 6, 1, 'bot', 6, 1, 1, 0, 3, 1, 3);  
   
 --View Plate\_Appearance table  
 SELECT \* FROM Plate\_Appearance;  
   
 --Insert data into Pitch\_event table  
 INSERT INTO Pitch\_event  
 (GameID, plate\_appearance\_id, pitch\_of\_pa, pitch\_type, result, outcome, balls, strikes, play\_description)  
 VALUES  
 --Top of the first inning, Cubs at bat.  
 ('2018\_03\_29\_MIA\_CHN\_0', 1, 1, 'FA', 'strike', 'home run', 0, 0, 'Ian Happ homers(1).'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 1, 'FA', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 2, 'FT', 'ball', 'ball', 2, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 3, 'FT', 'strike', 'strike', 2, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 4, 'CH', 'ball', 'ball', 3, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 5, 'FA', 'strike', 'strike', 3, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 2, 6, 'CH', 'ball', 'walk', 4, 2, 'Kris Bryant walks.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 3, 1, 'FA', 'strike', 'foul', 0, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 3, 2, 'FT', 'strike', 'foul', 0, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 3, 3, 'FT', 'ball', 'ball', 1, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 3, 4, 'FT', 'ball', 'hit by pitch', 1, 2, 'Anthony Rizzo hit by pitch. Bryant to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 1, 'FT', 'strike', 'foul', 0, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 2, 'SL', 'strike', 'strike', 0, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 3, 'FA', 'strike', 'foul', 0, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 4, 'CH', 'ball', 'ball', 1, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 5, 'SL', 'ball', 'ball', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 6, 'SL', 'strike', 'foul', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 4, 7, 'SL', 'strike', 'strikeout', 2, 3, 'Willson Contreras strikes out swinging.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 5, 1, 'CH', 'strike', 'strike', 0, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 5, 2, 'FA', 'strike', 'foul', 0, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 5, 3, 'FT', 'ball', 'ball', 1, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 5, 4, 'FA', 'ball', 'ball', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 5, 5, 'CH', 'strike', 'ground out', 2, 2, 'Kyle Schwarber grounds out. Bryant to 3rd. Rizzo to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 6, 1, 'SL', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 6, 2, 'FT', 'strike', 'foul', 1, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 6, 3, 'FA', 'ball', 'hit by pitch', 1, 1, 'Addison Russell hit by pitch.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 7, 1, 'CH', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 7, 2, 'FT', 'ball', 'ball', 2, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 7, 3, 'CH', 'ball', 'ball', 3, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 7, 4, 'FA', 'ball', 'walk', 4, 0, 'Jason Heyward walks. Bryant scores. Rizza to 3rd. Russell to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 8, 1, 'FA', 'ball', 'hit by pitch', 0, 0, 'Javier Baez hit by pitch. Rizzo scores. Russell to 3rd. Heyward to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 1, 'FA', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 2, 'FA', 'ball', 'ball', 2, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 3, 'FA', 'strike', 'strike', 2, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 4, 'FA', 'strike', 'foul', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 9, 5, 'FT', 'strike', 'ground out', 2, 2, 'Lester grounds out.'),  
 --Bottom of the first inning, Marlins at bat.;  
 ('2018\_03\_29\_MIA\_CHN\_0', 10, 1, 'FA', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 10, 2, 'FA', 'strike', 'strike', 1, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 10, 3, 'SI', 'strike', 'ground out', 1, 1, 'Lewis Brison grounds out.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 11, 1, 'FA', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 11, 2, 'SI', 'strike', 'strike', 1, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 11, 3, 'SI', 'strike', 'ground out', 1, 0, 'Derek Dietrich grounds out.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 1, 'FA', 'strike', 'strike', 0, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 2, 'CU', 'ball', 'ball', 1, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 3, 'SI', 'ball', 'ball', 2, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 4, 'FC', 'strike', 'strike', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 5, 'FA', 'ball', 'ball', 3, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 12, 6, 'FA', 'strike', 'single', 3, 2, 'Starlin Castro singles.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 1, 'FA', 'ball', 'ball', 1, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 2, 'FA', 'ball', 'ball', 2, 0, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 3, 'SI', 'strike', 'strike', 2, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 4, 'FA', 'strike', 'strike', 2, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 5, 'FC', 'ball', 'ball', 3, 2, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 13, 6, 'FC', 'ball', 'ball', 4, 2, 'Justin Bour walks. Castro to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 14, 1, 'FC', 'strike', 'strike', 0, 1, NULL),  
 ('2018\_03\_29\_MIA\_CHN\_0', 14, 2, 'SI', 'strike', 'single', 0, 1, 'Brian Anderson singles. Castro scores. Bour to 2nd.'),  
 ('2018\_03\_29\_MIA\_CHN\_0', 15, 1, 'FA', 'strike', 'force out', 0, 1, 'Garrett Cooper grounds into a force out.');  
   
 --View Pitch\_event table  
 SELECT \* FROM Pitch\_event;  
   
 --Insert data into Pitch\_type table  
 INSERT INTO Pitch\_type  
 (Pitch\_type, Pitch\_name)  
 VALUES  
 ('FA', 'Four Seam Fastball'),  
 ('FS', 'Splitter'),  
 ('SI', 'Sinker'),  
 ('CU', 'Curveball'),  
 ('CH', 'Change Up'),  
 ('SL', 'Slider'),  
 ('KN', 'Knuckleball'),  
 ('CS', 'Slow Curve'),  
 ('FC', 'Cutter'),  
 ('FT', 'Two Seam Fastball');  
   
 --View Pitch\_type table  
 SELECT \* FROM Pitch\_type;   
   
 --Insert data into Pitch table  
 INSERT INTO Pitch   
 (pitchID, pitch\_type, release\_point, velocity, x\_movement, z\_movement, max\_plate\_dist, max\_plate\_time, next\_pitchID)  
 VALUES  
 (1, 'FA', 127.2, 95.8, 0.48, 2.38, 0.69, 0.00, 2),  
 (2, 'FA', 124.4, 95.6, -2.40, 2.25, 0.80, 0.00, 3),  
 (3, 'FT', 117.8, 95.3, -1.70, 1.96, 0.60, 0.00, 4),  
 (4, 'FT', 124.6, 94.8, -1.21, 1.04, 0.66, 0.02, 5),  
 (5, 'CH', 124.8, 89.8, -0.72, 1.42, 0.85, 0.03, 6),  
 (6, 'FA', 125.9, 94.6, -0.15, 2.56, 0.75, 0.02, 7),  
 (7, 'CH', 117.5, 89.7, -1.41, 2.56, 0.85, 0.03, 8),  
 (8, 'FA', 124.6, 95.3, 0.75, 2.03, 0.80, 0.00, 9),  
 (9, 'FC', 123.2, 96.0, 0.51, 2.44, 0.60, 0.00, 10),  
 (10, 'FT', 126.6, 95.8, 1.08, 2.69, 0.60, 0.00, 11),  
 (11, 'FT', 125.5, 95.2, 2.02, 2.90, 0.60, 0.00, 12),  
 (12, 'FT', 126.2, 95.6, -0.98, 1.74, 0.95, 0.04, 13),  
 (13, 'SL', 112.8, 84.3, 0.33, 1.38, 0.72, 0.04, 14),  
 (14, 'FA', 109.3, 95.2, -1.18, 1.43, 0.75, 0.02, 15),  
 (15, 'CH', 123.4, 87.9, -1.32, 1.81, 0.87, 0.01, 16),  
 (16, 'SL', 108.9, 81.0, -1.10, 4.08, 0.70, 0.01, 17),  
 (17, 'SL', 110.9, 83.6, -0.07, 1.93, 0.70, 0.01, 18),  
 (18, 'SL', 103.8, 82.8, 0.09, 2.13, 1.11, 0.01, 19),  
 (19, 'CH', 123.4, 88.0, -0.44, 2.71, 0.85, 0.03, 20),  
 (20, 'FA', 125.5, 94.9, -0.05, 1.82, 0.80, 0.00, 21),  
 (21, 'FT', 117.3, 95.3, 0.50, 3.24, 0.78, 0.00, 22),  
 (22, 'FA', 115.4, 96.1, 2.67, 2.83, 0.75, 0.02, 23),  
 (23, 'CH', 126.3, 90.5, -0.65, 1.23, 0.87, 0.01, 24),  
 (24, 'SL', 123.9, 85.6, 0.47, 3.88, 1.17, 0.04, 25),  
 (25, 'FT', 125.0, 95.9, -0.73, 2.62, 0.78, 0.00, 26),  
 (26, 'FA', 111.8, 96.0, -2.78, 3.36, 0.75, 0.02, 27),  
 (27, 'CH', 125.2, 89.4, -1.14, 1.55, 0.69, 0.03, 28),  
 (28, 'FT', 114.8, 95.1, -1.10, 2.79, 0.66, 0.02, 29),  
 (29, 'CH', 122.2, 90.2, 0.27, 0.21, 0.85, 0.03, 30),  
 (30, 'FA', 114.4, 95.6, 1.14, 1.46, 0.69, 0.00, 31),  
 (31, 'FA', 124.3, 95.6, -1.72, 3.47, 0.69, 0.00, 32),  
 (32, 'FA', 122.6, 94.1, -1.41, 3.28, 0.69, 0.00, 33),  
 (33, 'FA', 120.5, 93.4, 1.74, 2.94, 0.69, 0.00, 34),  
 (34, 'FA', 123.6, 93.9, 0.78, 2.47, 0.69, 0.00, 35),  
 (35, 'FA', 121.6, 94.1, -0.27, 2.34, 0.69, 0.00, 36),  
 (36, 'FT', 122.8, 94.3, -0.02, 2.18, 0.78, 0.00, 58); /\*Pitch 58 of the game would be the next Jose Urena pitch.\*/  
 /\*Pitches 37 through 57 were thrown by Jon Lester.\*/  
   
 --View Pitch table  
 SELECT \* FROM Pitch;

# 6. Data Manipulation

/\*   
 We want to create functions that are able to calculate the following data for pitchers:   
 1. Number of innings pitched  
 2. Number of batters faced  
 3. RA/9  
 4. K/9  
 5. BB/9  
 6. H/9  
 7. HR/9  
 8. Opp BA/OBP/SLG  
 9. Average pitch velocity for individual pitchers by pitch type  
 10. Tunnel point size   
 11. Average release point for individual pitchers by pitch type  
   
 Views we want to make would be summaries of games and innings: number of batters faced, pitches thrown, hits allowed, runs scored, and the   
 metrics that were calculated with the previous functions.  
 \*/  
   
 --This function will calculate total innings pitched per pitcher  
 CREATE FUNCTION dbo.innings\_pitched(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @ip float  
 SELECT @ip = sum(T.inning)  
 FROM (SELECT GameID, pitcherID, max(inning) as inning   
 FROM Plate\_Appearance   
 WHERE Plate\_Appearance.pitcherID = @pitcher  
 GROUP BY GameID, pitcherID) as T  
 RETURN @ip  
 END;  
   
 --The following function will calculate the number of batters faced per pitcher  
 CREATE FUNCTION dbo.batters\_faced(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @bf float  
 SELECT @bf = sum(T.batters)  
 FROM (SELECT GameID, pitcherID, count(batterID) as batters   
 FROM Plate\_Appearance   
 WHERE Plate\_Appearance.pitcherID = @pitcher  
 GROUP BY GameID, pitcherID) as T  
 RETURN @bf  
 END;  
   
 --Calculates runs allowed per 9 innings (RA \* 9/IP)  
 CREATE FUNCTION dbo.RA9(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @ra9 float  
 SELECT @ra9 = sum(T.runs) \* 9 / sum(T.inning)  
 FROM (SELECT GameID, pitcherID, max(inning) as inning,   
 case when half\_inning = 'top' then max(away\_score) else max(home\_score) end as runs   
 FROM Plate\_Appearance   
 WHERE Plate\_Appearance.pitcherID = @pitcher  
 GROUP BY GameID, pitcherID, half\_inning) as T  
 RETURN @ra9  
 END;  
   
 --Calculate rest of the per9 stats (K, BB, H, HR)  
 CREATE FUNCTION dbo.K9(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @k9 float  
 SELECT @k9 = sum(T.k) \* 9 / sum(T.inning)  
 FROM (SELECT pitch\_event.GameID, pitcherID, max(inning) as inning,   
 count(outcome) as k  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE --Plate\_Appearance.pitcherID = @pitcher AND  
 outcome = 'strikeout'  
 GROUP BY pitch\_event.GameID, pitcherID, outcome) as T  
 RETURN @k9  
 END;  
   
 CREATE FUNCTION dbo.BB9(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @bb9 float  
 SELECT @bb9 = sum(T.bb) \* 9 / sum(T.inning)  
 FROM (SELECT pitch\_event.GameID, pitcherID, max(inning) as inning,   
 count(outcome) as bb  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome = 'walk'  
 GROUP BY pitch\_event.GameID, pitcherID, outcome) as T  
 RETURN @bb9  
 END;  
   
 CREATE FUNCTION dbo.H9(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @h9 float  
 SELECT @h9 = sum(T.h) \* 9 / sum(T.inning)  
 FROM (SELECT pitch\_event.GameID, pitcherID, max(inning) as inning,   
 count(outcome) as h  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run')  
 GROUP BY pitch\_event.GameID, pitcherID, outcome) as T  
 RETURN @h9  
 END;  
   
 CREATE FUNCTION dbo.HR9(@pitcher int)  
 RETURNS float AS   
 BEGIN  
 DECLARE @hr9 float  
 SELECT @hr9 = sum(T.hr) \* 9 / sum(T.inning)  
 FROM (SELECT pitch\_event.GameID, pitcherID, max(inning) as inning,   
 count(outcome) as hr  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome = 'home run'  
 GROUP BY pitch\_event.GameID, pitcherID, outcome) as T  
 RETURN @hr9  
 END;  
 --End of per9 stats  
   
 --Opposing batter slash-line stats (Batting Average/On-Base Percentage/Slugging)  
 CREATE FUNCTION dbo.opp\_ba(@pitcher int)  
 RETURNS decimal(4,3) AS   
 BEGIN  
 DECLARE @ba decimal(4,3)  
 SELECT @ba = cast(S.h as float) / cast(S.ab as float)  
 FROM (SELECT Plate\_Appearance.pitcherID, count(pitch\_event.outcome) as h, max(ab) as ab  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN (SELECT pitcherID, count(outcome) as ab  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run', 'strikeout', 'ground out', 'fly out', 'force out', 'double play', 'pop out')  
 GROUP BY pitcherID) as T on Plate\_Appearance.pitcherID = T.pitcherID  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run')  
 GROUP BY Plate\_Appearance.pitcherID) as S   
 RETURN @ba  
 END;  
   
 CREATE FUNCTION dbo.opp\_obp(@pitcher int)  
 RETURNS decimal(4,3) AS   
 BEGIN  
 DECLARE @obp decimal(4,3)  
 SELECT @obp = cast(S.ob as float) / cast(S.pa as float)  
 FROM (SELECT Plate\_Appearance.pitcherID, count(pitch\_event.outcome) as ob, max(pa) as pa  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN (SELECT pitcherID, count(distinct(Plate\_appearance.plate\_appearance\_id)) as pa  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher  
 GROUP BY pitcherID) as T on Plate\_Appearance.pitcherID = T.pitcherID  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run', 'walk', 'hit by pitch')  
 GROUP BY Plate\_Appearance.pitcherID) as S   
 RETURN @obp  
 END;  
   
 CREATE FUNCTION dbo.opp\_slg(@pitcher int)  
 RETURNS decimal(4,3) AS   
 BEGIN  
 DECLARE @slg decimal(4,3)  
 SELECT @slg = (cast(S.sgl as float) + cast(S.dbl as float) \* 2 + cast(S.trp as float) \* 3 + cast(S.hr as float) \* 4) / cast(S.ab as float)  
 FROM (SELECT Plate\_Appearance.pitcherID, max(ab) as ab,  
 case when outcome = 'single' then count(outcome) else 0 end as sgl,  
 case when outcome = 'double' then count(outcome) else 0 end as dbl,  
 case when outcome = 'triple' then count(outcome) else 0 end as trp,  
 case when outcome = 'home run' then count(outcome) else 0 end as hr  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN (SELECT pitcherID, count(outcome) as ab  
 FROM pitch\_event  
 JOIN Plate\_Appearance ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run', 'strikeout', 'ground out', 'fly out', 'force out', 'double play', 'pop out')  
 GROUP BY pitcherID) as T on Plate\_Appearance.pitcherID = T.pitcherID  
 WHERE Plate\_Appearance.pitcherID = @pitcher AND  
 outcome in ('single', 'double', 'triple', 'home run')  
 GROUP BY Plate\_Appearance.pitcherID, outcome) as S   
 RETURN @slg  
 END;  
 --End of slash-line stats  
   
 --Create view detailing pitcher performance  
 CREATE VIEW PitcherPerformance AS (  
 SELECT Player.last\_name, Player.first\_name, Player.TeamID, Player.season,  
 dbo.innings\_pitched(Player.playerID) as IPs,  
 dbo.batters\_faced(Player.playerID) as BF,  
 dbo.RA9(Player.playerID) as RA9,  
 dbo.K9(Player.playerID) as K9,  
 dbo.BB9(Player.playerID) as BB9,  
 dbo.H9(Player.playerID) as H9,  
 dbo.HR9(Player.playerID) as HR9,   
 dbo.opp\_ba(Player.playerID) as OppBA,  
 dbo.opp\_obp(Player.playerID) as OppOBP,  
 dbo.opp\_slg(Player.playerID) as OppSLG  
 FROM Player  
 WHERE positionID = 1  
 );  
   
 --Create view detailing pitch velocity and release points. Because we're just looking at averages, we don't need to create any extra functions  
 CREATE VIEW PitchPerformance AS (  
 SELECT Player.last\_name, Player.first\_name, Player.season, Player.TeamID, Pitch.pitch\_type,  
 COUNT(Pitch\_event.pitchID) AS Pitch\_count,  
 ROUND(AVG(pitch.velocity), 2) AS Pitch\_velo,  
 ROUND(AVG(pitch.release\_point), 2) AS Pitch\_relpoint  
 FROM Player  
 JOIN Plate\_Appearance ON Player.playerID = Plate\_Appearance.pitcherID  
 JOIN Pitch\_event ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN Pitch ON Pitch\_event.pitchID = Pitch.pitchID  
 GROUP BY last\_name, first\_name, season, teamID, Pitch.pitch\_type  
 );  
   
 --Check PitcherPerformance View  
 SELECT \* FROM PitcherPerformance ORDER BY last\_name;  
   
 --Check PitchPerformance View  
 SELECT \* FROM PitchPerformance ORDER BY last\_name, Pitch\_count;

# 7. Answering Data Questions

/\*We want to know how many pitches, grouped by pitch type, did Jose Urena throw in the first inning against the Cubs and   
 what his average velocity, release point, and movement was. \*/  
   
 -- Let's first check how many pitches he threw  
 SELECT playerID, concat(first\_name, ' ', last\_name) as player\_name, TeamID, inning, COUNT(pitch\_of\_pa) as pitches\_thrown  
 FROM Player  
 JOIN Plate\_Appearance ON Player.playerID = Plate\_Appearance.pitcherID  
 JOIN Pitch\_event ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 WHERE last\_name = 'Urena'  
 GROUP BY playerID, first\_name, last\_name, TeamID, inning;  
 -- Jose Urena threw 36 pitches in the first inning

## Number of pitches thrown by Jose Urena

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| playerID | player\_name | TeamID | inning | pitches\_thrown |
| 9 | Jose Urena | MIA | 1 | 36 |

-- How many pitches did he throw by pitch type?  
 SELECT playerID, concat(first\_name, ' ', last\_name) as player\_name, TeamID, inning, Pitch\_name, COUNT(pitch\_of\_pa) as pitches\_thrown  
 FROM Player  
 JOIN Plate\_Appearance ON Player.playerID = Plate\_Appearance.pitcherID  
 JOIN Pitch\_event ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN Pitch\_type ON Pitch\_event.pitch\_type = Pitch\_type.Pitch\_type  
 WHERE last\_name = 'Urena'  
 GROUP BY playerID, first\_name, last\_name, TeamID, inning, Pitch\_name  
 ORDER BY pitches\_thrown;  
 --Jose Urena threw 5 sliders, 7 changeups, 10 two-seam fastballs, and 14 four-seam fastballs.

## Number of pitches thrown by Jose Urena by pitch type

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| playerID | player\_name | TeamID | inning | Pitch\_name | pitches\_thrown |
| 9 | Jose Urena | MIA | 1 | Slider | 5 |
| 9 | Jose Urena | MIA | 1 | Change Up | 7 |
| 9 | Jose Urena | MIA | 1 | Two Seam Fastball | 10 |
| 9 | Jose Urena | MIA | 1 | Four Seam Fastball | 14 |

-- Now let's check the average release point, velocity, and movement for these pitch types  
 SELECT playerID, CONCAT(first\_name, ' ', last\_name) AS player\_name, TeamID, inning, Pitch\_name, COUNT(pitch\_of\_pa) AS pitches\_thrown,  
 ROUND(AVG(release\_point), 1) AS avg\_release\_point,   
 ROUND(AVG(velocity), 1) AS avg\_velo,   
 ROUND(AVG(x\_movement), 2) AS avg\_x\_movement,   
 ROUND(AVG(z\_movement), 2) AS avg\_z\_movement  
 FROM Player  
 JOIN Plate\_Appearance ON Player.playerID = Plate\_Appearance.pitcherID  
 JOIN Pitch\_event ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN Pitch\_type ON Pitch\_event.pitch\_type = Pitch\_type.Pitch\_type  
 JOIN Pitch ON Pitch\_event.pitchID = Pitch.pitchID  
 WHERE last\_name = 'Urena'  
 GROUP BY playerID, first\_name, last\_name, TeamID, inning, Pitch\_name  
 ORDER BY pitches\_thrown;  
 /\*   
 Things that pop out:   
 1. The changeup and fastballs have similar release points but there is a notorious drop in arm slot when he goes to the slider.  
 2. Urena's two-seamer is a bit faster than his four-seamer. The difference in movement is also negligible, which might indicate that it is the same pitch.  
 3. The changeup has the most x\_movement (vertical drop) than the other pitches.  
 4. The other three pitches have more horizontal movement along the pitching plane, meaning that they have more of a cut movement.   
 \*/

## Average release points, velocity, and movement for Urena’s pitches

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| playerID | player\_name | TeamID | inning | Pitch\_name | pitches\_thrown | avg\_release\_point | avg\_velo | avg\_x\_movement | avg\_z\_movement |
| 9 | Jose Urena | MIA | 1 | Slider | 5 | 112.1 | 83.5 | -0.06 | 2.68 |
| 9 | Jose Urena | MIA | 1 | Change Up | 7 | 123.3 | 89.4 | -0.77 | 1.64 |
| 9 | Jose Urena | MIA | 1 | Two Seam Fastball | 10 | 122.4 | 95.3 | -0.16 | 2.36 |
| 9 | Jose Urena | MIA | 1 | Four Seam Fastball | 14 | 120.8 | 95.0 | -0.17 | 2.47 |

-- A last question that I find interesting is how the pitches were split between balls and strikes, and which pitch was more prone to being outside the strike zone.  
 SELECT playerID, CONCAT(first\_name, ' ', last\_name) AS player\_name, TeamID, inning, Pitch\_name, result, COUNT(pitch\_of\_pa) AS pitches\_thrown  
 FROM Player  
 JOIN Plate\_Appearance ON Player.playerID = Plate\_Appearance.pitcherID  
 JOIN Pitch\_event ON Plate\_Appearance.plate\_appearance\_id = Pitch\_event.plate\_appearance\_id  
 JOIN Pitch\_type ON Pitch\_event.pitch\_type = Pitch\_type.Pitch\_type  
 WHERE last\_name = 'Urena'  
 GROUP BY playerID, first\_name, last\_name, TeamID, inning, Pitch\_name, result  
 ORDER BY pitch\_name, result, pitches\_thrown;  
 /\* The changeup was the likeliest pitch to be called a ball in this sample. The rest of the pitches were 50/50 meaning that Urena's command was probably off   
 during this start. \*/

## Ball/Strike pitch split

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| playerID | player\_name | TeamID | inning | Pitch\_name | result | pitches\_thrown |
| 9 | Jose Urena | MIA | 1 | Change Up | ball | 5 |
| 9 | Jose Urena | MIA | 1 | Change Up | strike | 2 |
| 9 | Jose Urena | MIA | 1 | Four Seam Fastball | ball | 7 |
| 9 | Jose Urena | MIA | 1 | Four Seam Fastball | strike | 7 |
| 9 | Jose Urena | MIA | 1 | Slider | ball | 2 |
| 9 | Jose Urena | MIA | 1 | Slider | strike | 3 |
| 9 | Jose Urena | MIA | 1 | Two Seam Fastball | ball | 5 |
| 9 | Jose Urena | MIA | 1 | Two Seam Fastball | strike | 5 |

# 8. Implementation

# 9. Reflection

After working extensively on this project, I realize that creating and upkeeping a database is hard work - not as simple as I had assumed going in. I thought I would be able to simplify a series of database tables already in existence, but to my chagrin, I only managed to reduce two tables from the original database tables.  
If there are things that I would do differently, it would probably be on the events table. Though the information is useful to provide context, it could probably be omitted and merged with another of the tables using simplified notation.  
Another thing that I would do differently would be the data insertion. The amount of data in existence is too much and needs to be fed into the tables another way (Excel could be an option, though also limited.)  
Regarding information approach, in future events, the lessons learned lead me to think that this approach is better served by going bottom to top. Once the needs are established, understanding the purpose of the data and how it interacts between itself will help to create better, more robust, and easier to read data tables that - hopefully - will not need extensive joins or many lines of SQL code.