Final Project — Iron Will Sportsbook

DATA 5330/6330

# Introduction

In this project, you’ll apply what you’ve learned about data manipulation, pipelining, structured data repositories, and database testing within the context of a “real world” data engineering problem.

You will assume the role of a data engineer tasked to create a Postgres data mart for the use of a database-competent data scientist. The data scientist wants to perform a number of analyses on the data and to have the freedom to identify and complete even more as-yet-unidentified analyses based on the results of the initial wave of analysis. As such, creating (and testing) a relational database data mart allows the data scientist to recombine data as needed for this.

## Company and Dataset Information

The project and its data belong to Iron Will Sportsbook (motto: *feel the compulsion!*; alternate motto: *no credit check required!*), a sports gambling company. The company’s customers can place bets three ways: in-person (locally), online, or through the company’s call-center.

The data included refer to National Football League games as well as bets placed by a sample of the company’s customers. To understand the data, you need to understand how sports betting works. Specifically, there are two different types of bets within this dataset: betting line bets and over-under bets.

### Betting Line Bets

Here’s a very brief example of a betting line.

Team A plays Team B. Team A is expected to win by 6 points. As such, the betting line is described as “Team A -6”. That means that, if the bettor bets on Team A, Team A must win the game by *more than* six points for the bettor to win. If the bettor bets on Team B, then the bettor wins if Team B wins the game or loses by *fewer than* six points. If Team A wins the game by exactly six points, that’s referred to as a “push” — the bettor neither wins nor loses, but instead gets refunded the money of the bet (not including the commission — the commission is never refunded).

### Over-Under Bets

Another way to bet on a game is to bet on the “over-under”. For each game, the sports book sets an over-under line, which relates to the total number of points scored in the game (i.e., Team A’s score plus Team B’s score). If the bettor bets on the over, then the bettor wins if the two teams combine to score more points than the over-under line specifies. If the bettor bets on the under, then the bettor wins if they combine to score fewer points than the over-under line specifies. If the two teams combine to score the exact number of points as set in the over-under by the sports book, then the bet is refunded to the bettor (but the commission is retained by the sports book).

As with betting line bets, the sports book hopes to improve its profitability by better understanding factors related to bets placed by customers and factors related to customer value.

## How a Sports Book Makes Money

A sports book like Iron Will expects that the number of bets on either side of the line (whether the betting line or the over-under) will be very similar — this is the general intent when setting these lines, to encourage similar amounts of bets placed on either side. Most of a sports book’s income, then, comes not from encouraging more losing bets (although they’d do that if they could), but rather through commissions collected on bets. A usual commission to collect is 10% — in this case, a customer who places a $100 bet also pays a $10 commission for the bet, regardless of whether they win or lose (or “push”). Therefore, a $100 bet actually costs the customer $110, and, if they win, then they’ll get $200 back (their original $100 plus the $100 of winnings).

### Iron Will’s Commission Schedule

To encourage more high-stakes betting (based on gambling commission guidelines, the maximum bet that can be placed through Iron Will is $20,000), Iron Will has a graduated commission schedule whereby the higher the bet, the lower the net commission charged. Here’s the schedule:

* 10% commission on the first $1,000 bet.
* 8% commission on the next $4,000 bet.
* 6% commission on any additional dollars bet.

Here are some examples of calculating the commission on bets.

* Customer A places a $500 bet. Since this is under $1,000, the commission is 10%, so the cost of the $500 bet is 10% \* $500 = $50. In this case, Customer A pays Iron Will $550 up-front. If they win, then they get $1,000 back.
* Customer B places a $3,000 bet. The first $1,000 of this bet is charged a 10% commission, and the remaining $2,000 is charged an 8% commission. The net commission charged, then, is 10% \* $1,000 + 8% \* $2,000 = $260.
* Customer C places a $15,000 bet. For this bet, the commission is 10% on the first $1,000, 8% on the next $4,000, then 6% for the remaining $10,000, so: 10% \* $1,000 + 8% \* $4,000 + 6% \* $10,000 = $1,020.

Again, **the customer does not win the commission back if they win the bet.** When the bet amount is shown in the data set, it **does not include the commission value**.

### The Other Way They Make Money

Iron Will also makes money when a customer makes losing bets. While random chance would suggest that customers should generally win bets 50% of the time, some customers are inherently better or worse at betting — perhaps because they understand football better/worse, have insider knowledge, or don’t pay attention to available information about injuries, etc. When understanding which customers are “more valuable” to Iron Will, all else equal, a customer that loses more often than they win is therefore more valuable to the company.

## Data Sources

For this project, you’ll be accessing and manipulating data from two different data sources: (1) a set of three .csv files with information regarding NFL football games from 2001 through the 2023 season, all the NFL teams that have existed in that time period, and all the stadiums that have been used in that time period and (2) a two-table database that contains information regarding a sample of 2,000 high-frequency sports betting customers of Iron Will Sportsbook as well as the bets they placed on NFL games during the 2023 football season.

The .csv files include:

* spread\_scores.csv
* nfl\_teams.csv
* nfl\_stadiums.csv

The database is hosted on a SQL Server database server and can be accessed using the following credentials:

Server: stairwaytoheaven.usu.edu

Username: 5330user

Password: pipelinesnow

Database: ironwill

The database’s tables include:

* customer\_table (includes information about the customers included in the sample)
* betting\_log (information about those customers and the bets they placed during the season)

## Customer and Betting Columns

To help you out a bit, particularly when it comes to making calculations and completing the eventual linear regression models, here are some further details regarding columns in both the customer\_table table and the betlog table.

### customer\_table

* customer\_id: The primary key (integer).
* customer\_name: The customer’s first and last name.
* customer\_age: The customer’s age as of January 1, 2023.
* customer\_type: Whether the customer places bets online, via phone (phone), or in person (local).
* customer\_since: The year the customer created an account with Iron Will.
* customer\_income: The customer’s self-reported income for 2023.
* household\_size: The customer’s self-reported household size (number of individuals living in the same household).
* mode\_color: Based on a personality survey, the color mode that best describes the customer (red, yellow, blue, orange, green, purple, black, white).

### betlog

* bet\_id: Primary key (integer).
* customer\_id: Foreign key, references customer\_table.
* game\_id: The game’s unique identifier, indicates the season (i.e., 2023), the week of the season, the home team (abbreviated), and the away team (abbreviated).
* bet\_amount: The amount of the best placed in US$. This value EXCLUDES the commission paid to Iron Will.
* bet\_on: What the customer bet on — either the name of the team if it was a line bet or the word over or under if it was an over-under wager.

## Note from the Data Scientist

Hey Team!

So glad you’re setting this up for me. Here’s what I need: food, water, shelter, sunlight. As far as the project goes, I’d like you to deliver me a Postgres database that includes all of the data provided (i.e., don’t leave any of it out — it should all find a proper home in one of the tables). Since I want to be able to track the customers’ bets and data regarding the game on which they placed the bet, I need to be able to relate data from the ironwill database with data from the flat files.

Please be sure that all data originating from the sources are retained. In addition to setting up the database, I’d like you to provide me an ERD and a data dictionary that you used in setting up the physical structure of the database — that’ll be helpful for me to understand how the whole thing is supposed to work. I’d also ask you to test the usefulness of the database by performing certain queries and a regression analysis and verifying that the database can provide effective, accurate output.

If you have any questions, I’ll be on an arctic expedition for the next month or so and outside cell range, so hopefully you can answer them yourself.

Thanks!

# Deliverable

Complete this project within a Jupyter Notebook .ipynb file called fp\_*firstnamelastname*.ipynb (e.g., fp\_briandunn.ipynb) — with three exceptions:

* You can run the CREATE DATABASE command outside of Jupyter (e.g., in a database client).
* For the ERD, use LucidChart and submit a .pdf of the ERD. Be sure that the ERD is big enough to be easily viewed (it should take up the full page).
* You’ll need to provide a data dictionary Excel document.

## ERD Requirements

Your ERD should show:

* Each table included in the database.
* Each relationship.
* For each relationship, which side is the one side and which is the many side.
* Within each table, the names of all fields.
* Labels for all primary keys (PK) and foreign keys (FK).

## Data Dictionary Requirements

Address the database’s physical design by creating a data dictionary showing, for each table in the database and for each field within each table, the field name, whether it’s a primary or foreign key (and, if it’s a foreign key, the name of the table it references), the data type to be used, whether the field should be specified NOT NULL, the default value (if any), and a very brief description of the field.

* Create this within an Excel file.
* List each of the tables within the same tab, organizing the tables alphabetically.
* Skip one line between tables.
* For each table listed in your spreadsheet, repeat the headers.
* Provide names of columns in the same order as shown in your ERD.
* Use the data in the provided datasets as well as information given above to determine appropriate data types, null value status, and description.
  + Your data types should accommodate all the data provided in the tables.
  + However, bear in mind “efficiency” and don’t be \*too\* permissive (e.g., when setting VARCHAR length).
  + Also keep in mind whether values should include only a certain, specific quantity of characters.

## Database Requirements

* Name your database iwdm (for Iron Will Datamart).
* You should create and populate the database all within your Jupyter notebook file. Use the provided .csv files and SQL Server database for the source data.
* Be sure that your database tables are related appropriately — you may need to identify/create suitable primary and foreign keys to enable this.
* The table that includes information about individual games should also include a dummy field called winner\_ou that indicates whether the total score was greater than the over-under value (“over”), below the over-under (“under”), or a push (“push”). You’ll need to add this.
* The individual game table should also indicate in a column called winner\_line whether the winning bet would have been on the home team (“home”), the away team (“away”), or if the game was a push (“push”).
* The individual game table should only include games played starting in the 2015 season.
* In the Placed\_Bet table that appears in the data mart, add a column that indicates the result for the bettor — whether the bet resulted in a win, a loss, or a push (win, loss, push). To do this, you’ll need to consider the team for which the bettor bet, the final score of the game, **and the betting line for the game** (or the over-under bet, the final score, and the over-under line).
* Also in that Placed\_Bet table, add a column for the commission value paid by the bettor (see above for the commissions schedule).
* The customer table should include separate columns for each customer’s first and last names.

## Test Queries

For each of the queries, output the results of the query to a dataframe, then print the content of the dataframe. In your notebook, number these queries and include them in their own cells.

1. In separate queries, output the first five rows for data in each table.
2. The company would like to send customers who paid over $20,000 in commissions during the season a gift basket with dried fruit and cheeses. In one query, output the number of customers who paid over $20,000 in commissions, the total number of customers in the data mart, and the percent that would receive a gift basket. In a separate query, output the names and total amount paid in commissions of the 20 customers who paid the most in commissions during the season (so long as at least 20 paid over $20,000 — otherwise, only output those that paid over $20,000 in commission).
3. Among those who placed at least six bets, who are the ten luckiest bettors (in terms of percentage of bets won)? For each, show the first name, last name, number of bets placed, number of bets won, winning percentage, and total amount won. Order results first by winning percentage, then by amount won.
4. Who were the 20 costliest customers for the sports book[[1]](#footnote-1)? In other words, find the 20 customers for which the company had the greatest net loss in terms of dollars (i.e., compare what the company won from the customer versus what the company paid out). For each, note the customer’s first and last name, the number of bets placed by the customer, the number of bets won by the customer, and the net amount lost by the company.
5. For each week of the 2023 season, **ignoring commissions,** what percentage of games were “winners” for the sports book and what percentage were “losers” (i.e., in what percentage did they win money and in what percentage did they lose money)? For each week, show the week number, the total number of games played, the number of “winner” games (for the sports book), the number of “loser” games (for the sports book), the percentage that were winners, and the percentage that were losers. Order weeks chronologically from earliest to latest.
6. For each team during the 2023 season, how many times has the team been bet *for* and how many times *against*? For each team, show the team name, the number of wins for the team in the 2023 season (i.e., the number of times they scored more points than their opponent), the number of losses for the team in the 2022 season (scored fewer points than their opponent), the number of times they beat the spread in the 2023 season (i.e., won by more than the spread if the favorite, lost by less than the spread or won the game outright if the underdog), the number of bets placed in favor of the team, and the number of bets placed against the team. Order results alphabetically by team.

## Test Analysis

1. Finally, test the data mart by developing a regression model that predicts a customer’s value for the 2023 season. A customer’s value is defined as the total amount bet by the customer plus the commission paid by the customer minus the total amount paid out by the company. You’ll submit a correlation matrix verifying lack of collinearity, then provide two regression models (see below). For each regression model, output in your notebook the OLS Regression Results table.
   1. Run a correlation matrix with all numerical variables (as well as dummy variable versions of the “customer type” variable), then output the results of the matrix to the console. In a print statement below the matrix, answer these questions:
      1. Are there any collinearity issues among the independent variables?
      2. If so, between which variables (report the correlation values)?”
   2. Run a regression that predicts customer value using age, customer type, customer income, and household size as independent variables.
   3. Run a second regression using variables of your choosing. This second regression should result in a higher adjusted-*R2* value than in the original regression. In creating this new regression:
      1. Assuming the dataset used is correct (e.g., there weren’t problems with loading data with customer value calculations), the submission with the highest adjusted-*R2* value will get **four extra credit points**.
      2. This second regression needs to use at least one variable not included in the first model (or you can use a different version of the same variable, for instance, by making a continuous variable categorical or creating an interaction effect).
      3. Your regression needs to use “exogenous-enough” variables as independent variables in your analysis — avoid total amount bet, number of bets placed, winning percentage, etc.
      4. Consider feature-engineering some variables — does name length matter? Does their personality color matter? Are there potentially interaction effects?
      5. Hint: To increase adjusted-*R2*, removing insignificant predictor variables is usually helpful.

1. For example, if a customer places a $100 bet and wins, then the amount bet is $100, the commission is $10, and the amount paid out by the company is $200 — this would register as -$90 in customer value. If a customer places a $500 bet and loses, then this counts as +$550 in customer value (inclusive of the 10% commission charged). [↑](#footnote-ref-1)