Alexandra Speer
Nicolas Henzel
Gruppe 2
Hochschule der Medien
Sommersemester 23

HOCHSCHULE
DER MEDIEN

Data Warehouse Workshop

Prof. Dr. Klaus Freyburger

Inhaltsverzeichnis

Daten

Exploration

Die Datensätze stammen von Kaggle[1]. Inhalt sind die Weltmeisterschaften der Formel 1 im Zeitraum von 1950 bis 2022. Die Daten sind durch die Ergast Developer API[2] zusammengestellt worden, die Renndaten für nicht kommerzielle Zwecke zur Verfügung stellt.

Die Formel 1 (oft auch F1) ist eine Formelserie, die durch den Automobil Dachverband Fédération Internationale de l'Automobile (FIA) autorisiert ist. Formelserie bedeutet hierbei, dass bestimmte Regeln (Formeln) auf technischer Ebene für die Leistungsfähigkeit der Fahrzeuge festgelegt wurden, um einen Wettkampf unter gleichbleibenden Bedingungen zu ermöglichen. Die Formel 1 Weltmeisterschaft fand erstmals 1950 statt und besteht pro Saison aus bis zu 23 Grand Prix (französisch für „Großer Preis) Rennen. Dies sind Einzelrennen auf ausgewählten Rennstrecken in jeweils unterschiedlichen Ländern. Dabei sammeln die Fahrer abhängig von ihrer Endposition bei diesen Rennen Punkte. Am Ende der Saison gewinnt der Fahrer mit den meisten Punkten. Außerdem erhalten die Konstrukteure der Wagen Punkte, die ebenfalls am Ende der Saison ausgewertet werden.[3]

Die Daten beinhalten alle Informationen der Formel 1 Weltmeisterschaften seit 1950:
Rennen, Fahrer, Konstrukteure, Qualifizierungen, Rennstrecken, Rundenzeiten, Boxenstopps und Ergebnisse. Diese Informationen sind in 14 Datensätze aufgeteilt. Eine Übersicht der Daten in Python findet sich im DataExploration.ipynb. Die wichtigsten Erkenntnisse aus dieser Übersicht sind:

- Die Daten sind bereits „tidy“:
  Jede Zeile beinhaltet eine Beobachtung,
  jede Spalte beschreibt eine Variable,
  in jeder Zelle ist genau ein Wert (und nicht mehrere).[4]
  Ausnahme hierzu bilden die Tabellen „races“ und „qualifying“, in der einige Werte fehlen. Das wird im Data Wrangling behoben
- Die Datensätze besitzen mindestens eine Id Spalte (Primärschlüssel). wodurch sich die Daten leicht referenzieren lassen. Manche Datensätze haben weitere Id Spalten aufgelistet, die als Fremdschlüssel verwendet werden können.

Zusätzlich haben wir eine Tabelle zum Mapping der Länder und Nationalitäten genutzt.
Die Daten dazu haben wir von einem öffentlichen GitHub Repository heruntergeladen:
https://github.com/Imagin-io/country-nationality-list/blob/master/countries.csv

[1] https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020

[2] http://ergast.com/mrd/

[3] https://de.wikipedia.org/wiki/Formel_1

[4] https://cran.r-project.org/web/packages/tidyr/vignettes/tidy-data.html

Data Wrangling

Die Daten wurden im ersten Schritt im DataExploration.ipynb untersucht und in einer flachen Form zusammengeführt. Dabei haben wir uns auf die Fahrer Daten fokussiert und die Id Spalten genutzt, um die unterschiedlichen Datensätze miteinander zu joinen. Die Konstrukteur Daten sind im ER-Modell enthalten. Die Struktur dient zur ersten Übersicht, sowie beispielhaften Verbindung der Datensätze zu einem finalen Datensatz. Damit wurden beispielhafte Auswertungen im Python Skript umgesetzt (siehe Anhang).

```
Int64Index: 9631 entries, 0 to 9630
Data columns (total 32 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   number                 9631 non-null   object
 1   forename               9631 non-null   object
 2   surname                9631 non-null   object
 3   dob                    9631 non-null   object
 4   nationality            9631 non-null   object
 5   points                 9631 non-null   float64
 6   wins                   9631 non-null   int64
 7   avgMillisecondsLap     9631 non-null   float64
 8   pitStop                9631 non-null   int64
 9   lapOfPitStop           9631 non-null   int64
 10  timeOfPitStop          9631 non-null   object
 11  MillisecondsOfPitStop  9631 non-null   int64
 12  startingPos            9631 non-null   int64
 13  finalPos               9631 non-null   int64
 14  totalLaps              9631 non-null   int64
 15  timeToFinish           9631 non-null   object
 16  millisecondsToFinish   9631 non-null   object
 17  fastestLap             9631 non-null   object
 18  fastestLapRank         9631 non-null   object
 19  fastestLapTime         9631 non-null   object
...
 30  lng                    9631 non-null   float64
 31  alt                    9631 non-null   object
```

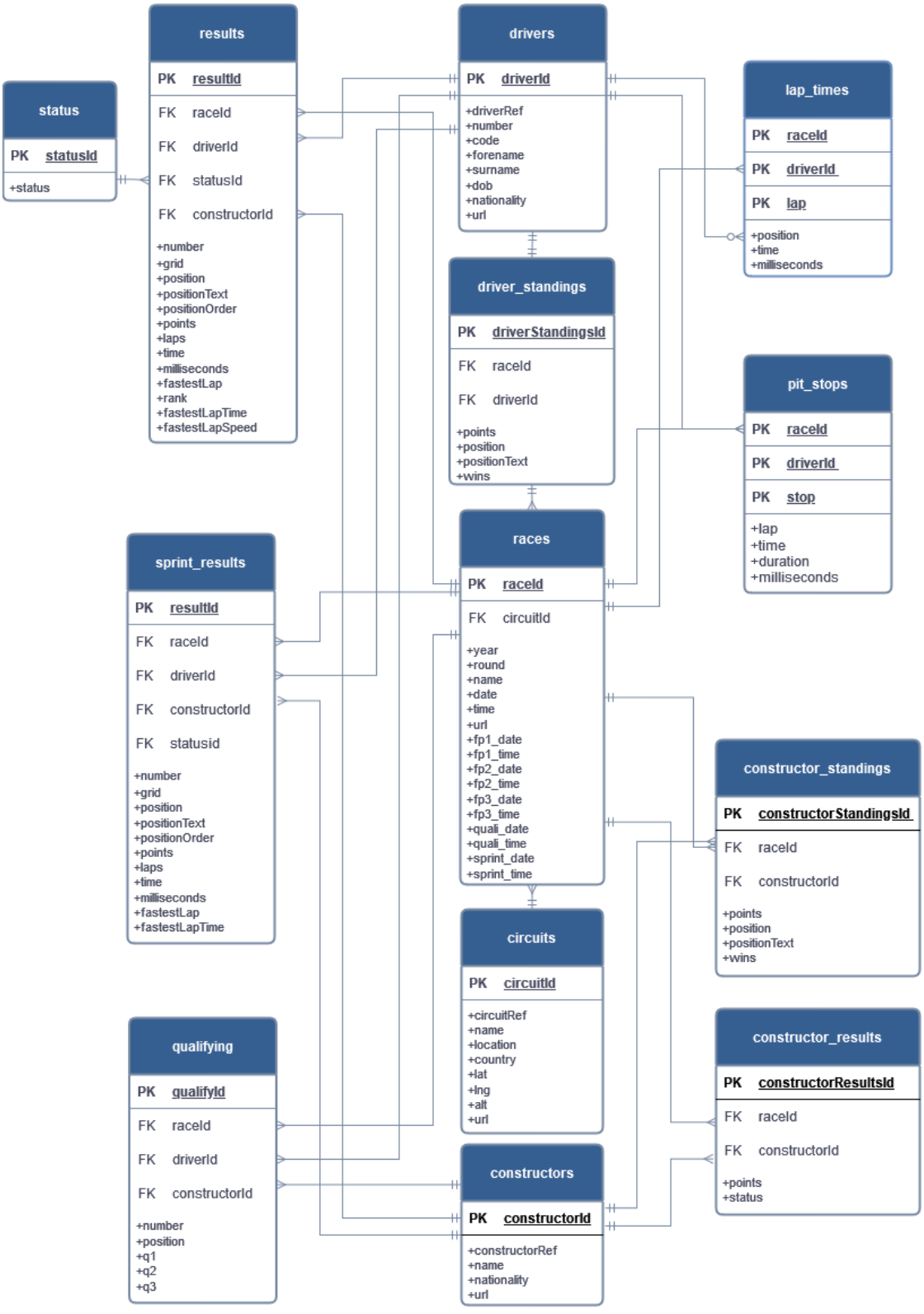
Die Datensätze, die genutzt wurden, sind:

- circuits.csv
- driver_standings.csv
- drivers.csv
- lap_times.csv
- pit_stops.csv
- results.csv
- races.csv
- status.csv
- constructors.csv
- constructor_standings.csv
- constructor_results.csv
- qualifying.csv
- sprint_results.csv

Der seasons.csv Datensatz wurde nicht genutzt, da hier keine neuen Informationen enthalten sind.

Logisches Modell

Die hauptsächlichen Tabellen für die Auswertungen sind races, drivers, und constructors. Hier werden alle anderen Inhalte gejoint.



Implementierung

Zum initialen Laden der Tabellen in DWC wurde eine JSON-Datei erstellt (180_FormulaOne.json), die die Tabellen und Verbindungen anlegt.
Folgende Anpassungen haben wir vorgenommen:

- Driver_standings.csv - positionText Spalte entfernt
- Pit_stops.csv - duration Spalte entfernt
- Results.csv - position, positionText Spalte entfernt
- Constructor_standings.csv - positionText Spalte entfernt
- Sprint_results.csv - position, positionText Spalte entfernt
- Alle „URL“ Spalten in jedem Datensatz entfernt (für die Auswertung nicht relevant)

Die entfernten Spalten haben keinen Mehrwert geboten, da alle Informationen bereits in anderen Spalten vorhanden waren. Die konkreten Schritte sind im DataWrangling.ipynb nachzuvollziehen.

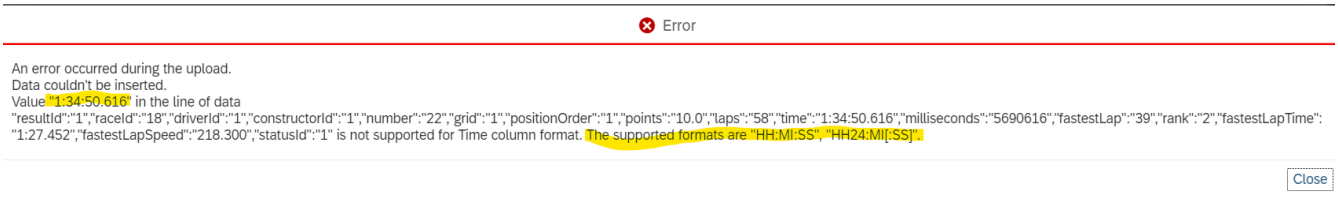
Die initiale Implementierung der Tabellen wurde mit der manuell erstellten JSON-Datei durchgeführt, wodurch das E/R Modell per Import in DWC angelegt wurde. Alle weiteren Anpassungen sind in DWC direkt durchgeführt worden.

Datenqualität für die Analyse

Im Abschnitt Daten haben wir angemerkt, dass in der Tabelle „races“ (in DWC 180_FormulaOneRaces) einige Werte fehlen. Konkret sind die Spalten I – R (fp1_date, fp1_time, …, sprint_time) nur mit 67 von 1103 Werten oder weniger befüllt. Aus diesem Grund haben wir hier keine Auswertungen erstellt, da nicht ausreichend Daten vorhanden sind.
Ähnliches gilt für die Tabelle „qualifying“ und „sprint_results“ – da hier viele Datenpunkte fehlen, oder zu wenige vorhanden sind (sprint_results) werden mit diesen Daten keine Auswertungen durchgeführt.

In der Tabelle „results“ ist die Zeit in zwei unterschiedlichen Formaten enthalten. Für den ersten Platz eines Rennens ist in Zeitformat in der Form HH:MM:SS.MS angegeben, alle weiteren Zeiten des gleichen Rennens in der Form +S.MS. Da die Transformation dieser Daten innerhalb von DWC für uns nicht möglich war, haben wir uns entschieden diese Daten nicht zu verwerten (siehe Anhang).

Beim Importieren der Daten sind einige Inhalte in Spalten, die einen Zeitwert enthalten, nicht korrekt erkannt worden, da diese nicht im H:MM:SS Format vorliegen, sondern M:SS.MS. Deshalb wurden diese im String Format importiert.



Transformationen und Views

Die Spalten mit Zeitwerten im String Format wurden anschließend in DWC in das Format H:MM:SS transformiert, indem eine Berechnung in den jeweiligen genutzt wird:

TO_TIME(CONCAT('00:0',SUBSTRING(SPALTENNAME,1,4)))

Die Zeitwerte sind zusätzlich als Millisekunden in den entsprechenden Spalten in Integer Werten vorhanden. Diese konnten auch nach mehrfachen Versuchen nicht in das Format H:MM:SS transformiert werden, da die in DWC vorhandenen Funktionen[5] nicht ausreichend waren, um die Transformation durchzuführen:

Message
invalid name of function or procedure: TIME_FORMAT: line 2 col 3 (at pos 19), Code: 328, SQL State: HY000

Die konkreten Schritte sind im Anhang aufgelistet.

Es gibt Spalten mit Zahlen, die auf den ersten Blick wie Integer aussehen, nach dem Import jedoch Float sind, was wir auch anpassen mussten (z.B. points Spalte in results.csv).

Zuerst haben wir die SQL View 180_FormulaOneAVGLapTimeAnalysis und erstellt. In dieser haben wir pro Fahrer je Rennen die einzelnen Rundenzeiten und Rundenpositionen als Mittelwert aggregiert:

```
select "raceId","driverId","year","fullname",
CAST(ROUND(AVG("milliseconds")) AS INTEGER) AS avg_milliseconds,
 ROUND(AVG("position")) AS avg_position,
 "nationality", "dob","round","name","date","grid","positionOrder","points","laps","startingTime","fastestLap","rank","fastestLapTime","fastestLapSpeed","statusId"
from "180_LapRaceAnalysis"
group by "raceId","driverId","fullname","nationality","dob","year","round","name","date","grid","positionOrder","points","laps","startingTime","fastestLap","rank"
    ,"fastestLapTime","fastestLapSpeed","statusId"
```

*raceId	*driverId	year	fullname	avg_milliseconds	avg_position	nationality
1	1	2,009	Lewis Hamilton	97,564	10	British
1	2	2,009	Nick Heidfeld	97,636	15	German
1	3	2,009	Nico Rosberg	97,612	7	German

Hierbei konnten wir nur die durchschnittlichen Rundenzeiten in Millisekunden angeben, nicht im Format HH:MM:SS, da die Transformation nicht erfolgreich war (siehe oben).

5 https://help.sap.com/docs/SAP_HANA_PLATFORM/4fe29514fd584807ac9f2a04f6754767/20a61f29751910149f99f0300dd95cd9.html

Anschließend haben wir folgende Views für die Erstellung von Stories angelegt:
Dimensional

- 180_FormulaOneStatusHierarchy– Dimensional View, bildet die Hierarchie von Status zu Constructor, sowie Driver und Contsructor zum Land ab.
- 180_FormulaOneGeoDim – Dimensional View, zur Erstellung der Location Dimension.

Diese wurden zu den jeweiligen analytischen Views assoziiert, um eine Konsumierung in SAC zu ermöglichen.

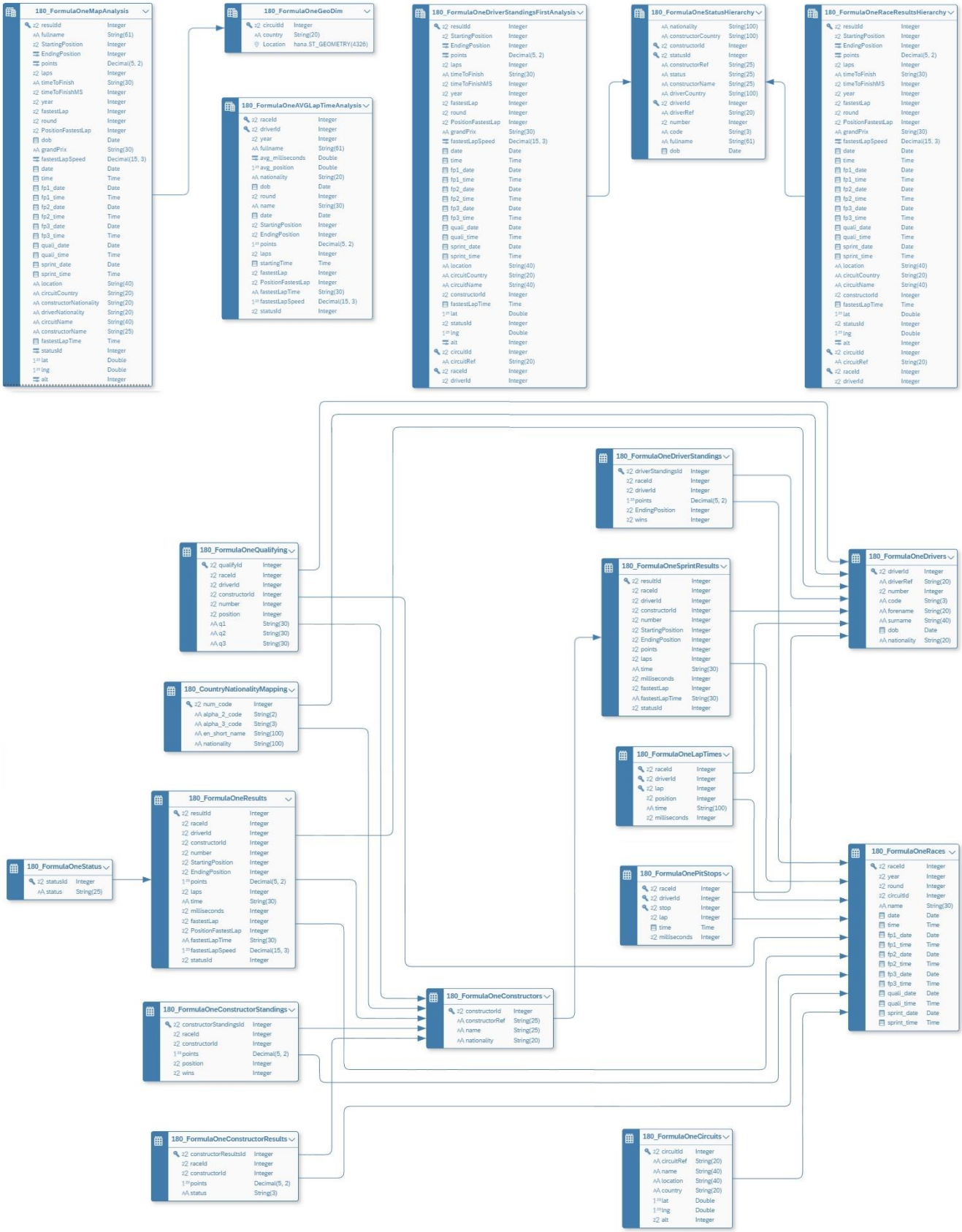
Analytical

- 180_FormulaOneDriverStandingsFirstAnalysis – Eingrenzung auf FinalPosition = 1 (Fahrer, die erster wurden im jeweiligen Rennen). Hierarchien sind eingebaut.
- 180_FormulaOneRaceResultsAnalysis – Analyse aller Rennergebnisse pro Fahrer und Konstrukteur auf jeder Strecke ab 1950 bis 2022.
- 180_FormulaOneAVGLapTimeAnalysis – Analyse der Durchschnittlichen Rundenzeit pro Fahrer pro Strecke. Auflistung der Jahre, um Werte vergleichbar zu machen.
- 180_FormulaOneMapAnalysis – Assoziation von 180_FormulaOneGeoDim zur Erstellung einer Karte.

Zu Beginn haben wir Views erstellt, ohne die Hierarchien abzubilden. Nachdem wir die Hierarchien eingebauten haben, mussten wir einige Auswertungen neu erstellen. Dieser Punkt hat einiges an Zeit gekostet, da die Änderungen von den assoziierten Dimensional Views jedes Mal neu deployed werden mussten, damit sie in SAC angezeigt wurden.

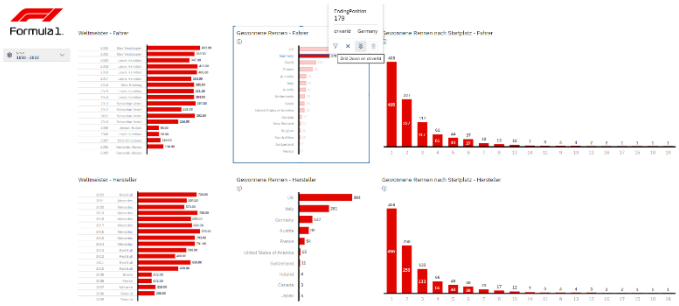
E/R Modell

Unser E/R Modell in DWC bildet das Logische Modell und die erstellen Views ab (180_FormulaOne):



Auswertung

Es wurde sich dazu entschieden in der Story (FormulaOne) mit mehreren Seiten zu arbeiten. Die erste Seite soll die gängigsten Fragen in Bezug zur Formel 1 beantworten. Auf der zweiten bis vierten Seite werden zwei der drei Haupttabellen (Fahrer und Rennstrecke) näher beleuchtet. Für den Hersteller könnte äquivalent zur Fahrertabelle ebenfalls eine Seite angelegt werden. In Anbetracht der zur Verfügung stehenden Zeit bei der Präsentation wurde hierauf verzichtet. Auf der letzten Seite wird auf ein unserer Meinung besonders interessanter Measure, den Status, eingegangen.

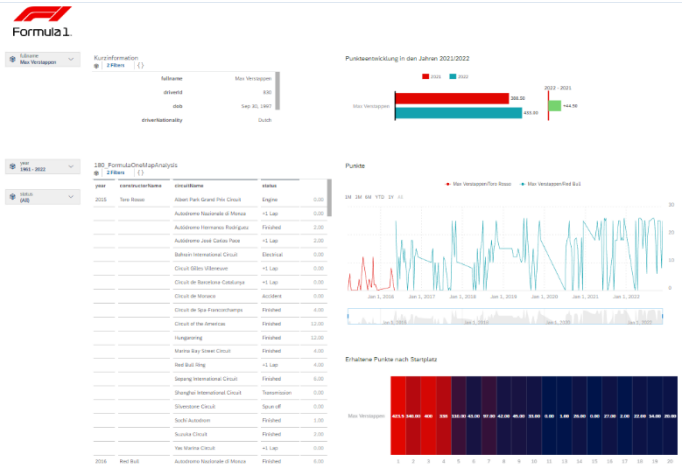


Überblick

Diese Seite besteht aus zwei Reihen an Diagrammen, wobei sich die Erste auf die Fahrer und die Zweite auf die Hersteller bezieht. In den beiden linken Diagrammen werden die Weltmeister abgebildet. Hierbei handelt es sich um eine Aufsummierung der erreichten Punktezahl pro Jahr. Durch ein Ranking wurde jeweils die Person/der Hersteller mit der höchsten Summe abgebildet. Zum Schluss wurde nach absteigenden Jahren sortiert. Da unsere Daten mehr als 70 Jahre umfassen wurde auf der linken Seite, unterhalb des Formel 1 Logos ein Input eingebunden, um nur eine gewünschte Zeitspanne abzubilden.

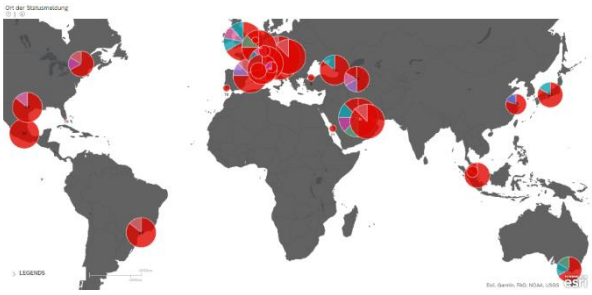
Im Vergleich dazu beziehen sich die restlichen Diagramme dieser Seite auf die Anzahl der gewonnenen Rennen. Die Diagramme in der Mitte zeigen die Anzahl der Rennen nach Nationalität. Dabei ist es möglich sich durch ein Dill-Down die einzelnen Fahrer/Hersteller anzeigen zu lassen.

In den zwei rechten Diagrammen wird auf der x-Achse die Startposition abgebildet. Es stellt dar, wie oft von der jeweiligen Position ein Sieg errungen wurde. Auch hierbei ist ein Drill-Down möglich. Um die Ergebnisse auf dem nächsten Level übersichtlich darstellen zu können wurde sich für ein Stagged Bar Chart entschieden.



Fahrer

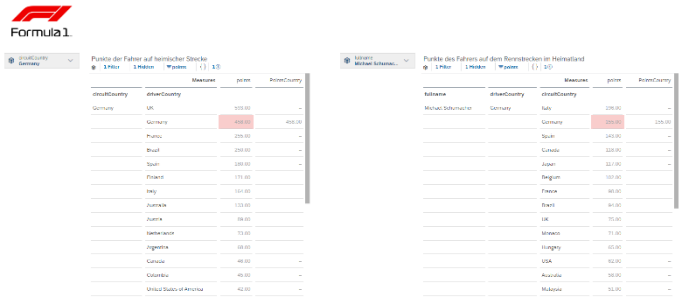
Auf dieser Seite wurde sich mit einer der Haupttabellen, den Fahrern, befasst. Mit Hilfe des ersten Input „fullname“ kann genau ein Fahrer ausgewählt werden. Auf diesen beziehen sich dann alle Auswertungen auf dieser Seite. Rechts davon befindet sich eine Tabelle mit wichtigen Informationen, bspw. dem Geburtsdatum und der Nationalität. Wiederrum rechts davon ist ein kleiner Überblick der Punkteentwicklung über die letzten beiden Jahre dargestellt. In unserem Fall 2021 und 2022. Diese Werte stammen in dieser Form nicht aus der Datasphere, sondern wurden durch einen Restricted Measure aus den Punkten



und der Dimension „date“ berechnet. Durch die hinzugefügte Varianz ist schnell ersichtlich, wie sich die Punkte in den letzten zwei Jahren entwickelt haben. Ein Nachteil dieses Diagramms ist, dass keine Daten abgebildet werden können, wenn ein Fahrer ausgewählt wird, welcher in den letzten zwei Jahren keine Rennen gefahren ist.

Darunter befinden sich zwei weitere Eingaben, zum einen das Jahr, zum anderen der Status. Die dort getätigte Auswahl hat keinen Einfluss auf die zwei darüber liegenden Darstellungen, jedoch auf alle anderen auf dieser Seite. Es gibt eine ausführliche Ergebnistabelle des Fahrers, in welcher der Hersteller sowie alle Rennen samt Punkte ausgegeben werden. Je nach Fahrer kann es mühsam sein alle Hersteller aus der Tabelle zu entnehmen. Um diese auf einen Blick erkennen zu können wurde die danebenliegende Time Series Chart mit den Herstellern eingefärbt. Diese gibt des Weiteren die erhaltenen Punkte wieder. Leider ist es nicht möglich bei diesem Diagrammtyp eine Beschriftung für die y-Achse hinzuzufügen.

In der Heatmap werden die Punkte nach Startplatz abgebildet. Als letztes wurde eine Karte eingebunden, welche die Status in einem Kuchendiagramm an den dazugehörigen Rennstrecken abbildet.



Heimvorteil

Es war ursprünglich geplant diese Tabelle ebenfalls auf der Seite der Fahrer einzubinden, sie wurde jedoch aus Gründen der Performance ausgelagert.

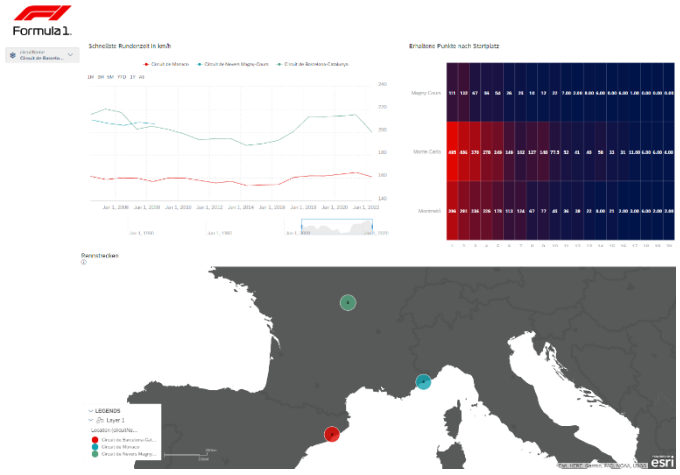
Auf dieser Seite soll dargestellt werden, ob ein Fahrer auf einer Strecke im eigenen Land einen Heimatvorteil hat. Um die Ergebnisse des Fahrers mit anderen Nationalitäten vergleichen zu können wurden zwei Tabellen mit jeweils einem eigenen Input angelegt. Auf der linken Seite wird das Land einer Rennstrecke ausgewählt. Es werden alle Nationalitäten von Fahren ausgegeben, die dort Punkte erreicht haben. Der Punktestand der Nationalität, welche eventuell einen Heimatvorteil hat wird rot hinterlegt. Auf der rechten Seite gibt es eine ähnliche Tabelle, jedoch wird dort ein Fahrer ausgewählt und es werden die Punkte dieses Fahrers in unterschiedlichen Ländern abgebildet. Wie zuvor ist der Punktestand im Heimatland rot hinterlegt.

Zu Beginn war beabsichtig durch die If-Funktion in einer Calculated Measure Strings ausgeben zu lassen, durch welche ersichtlich sein sollte, wann ein Fahrer Punkte auf einer heimischen Strecke erhalten hat. Diese Funktion akzeptiert jedoch nur nummerische Werte. Daher wurde sich für folgende Lösung entschieden und die Spalte „PointsCountry“ erstellt:

IF([d/"180_FormulaOneMapAnalysis":country]=[d/"180_FormulaOneMapAnalysis":driverCountry]
,["180_FormulaOneMapAnalysis":points])

Diese Funktion gibt die Punktezahl aus, wenn das Herkunftsland des Rennfahrers mit dem Land der Rennstrecke übereinstimmt. Im Vergleich zu dieser Spalte wurde im Conditional Formatting die

Spalte „points“ rot hinterlegt, sobald ein Fahrer auf einer heimatlichen Strecke fuhr. Somit sollte ein eventueller Heimvorteil leichter ersichtlich sein. Um das Vorgehen einfacher erklären zu können wurde die Spalte „PointsCountry“ nicht ausgeblendet.

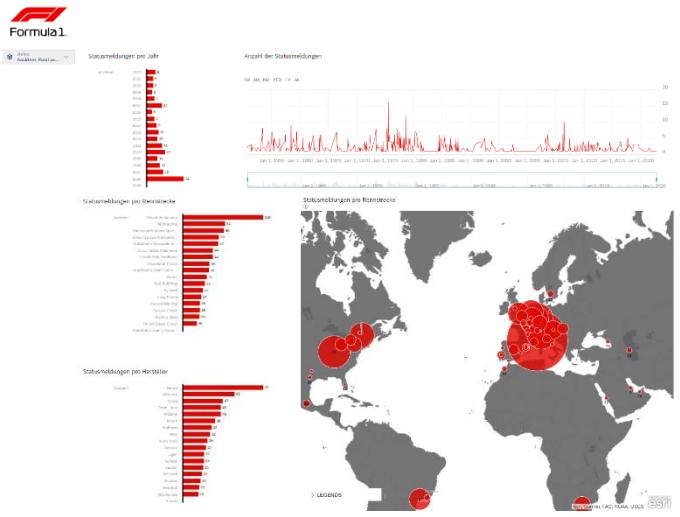


Rennstrecke

Auf dieser Seite wird sich mit der zweiten Haupttabelle, den Rennstrecken befasst. Es kann durch den Input eine oder mehrere Rennstrecken ausgewählt werden. Die Auswahl wirkt sich auf alle Diagramme auf dieser Seite aus.

Im ersten Diagramm werden die schnellsten Rundenzeiten pro Rennstrecke und Rennen abgebildet. Wie zuvor beschrieben ist es bei Time Series Charts nicht möglich eine Beschriftung auf der Y-Achse anzubringen. Daher wurde in der Überschrift auf die Einheit aufmerksam gemacht. Ein weiteres Problem war, dass nicht die richtige Zeitspanne (2000 – 2022) abgebildet wurde. Es wurde entweder nur das letzte Jahr oder der gesamte Zeitraum ab 1950, für welchen keine Geschwindigkeiten hinterlegt sind, dargestellt. Um die gewünschte Zeitspanne dauerhaft angezeigt zu bekommen wurde diese einmal richtig eingestellt und danach „Open in last saved view“ ausgewählt.

Daneben befindet sich eine Heatmap die darstellt, von welchen Startplätzen aus wie viele Punkte erreicht wurden. Damit soll Einblick gewährt werden, ob es bei manchen Rennstrecken vielleicht besser ist von einem Platz weiter hinten zu starten, da diese Position einen Überholvorgang in der ersten Kurve begünstigt etc. Darunter befindet sich erneut eine Karte, auf welcher die einzelnen Rennstrecken zu finden sind.



Status

Auf der letzten Seite wurde ein Measure abgebildet, den Status, welcher durch ein Input ausgewählt werden kann.

Zu Beginn gibt es zwei verschiedene Diagrammarten, welche ähnliches abbilden, die Anzahl und das Datum der Statusmeldungen. Eine Aggregierte Form wird im Balkendiagramm dargestellt, während das danebenliegende Time Series Diagramm den genauen Verlauf wiedergibt.

Darunter folgen ein Diagramm sowie eine Karte, die die Status in Zusammenhang mit den Rennstrecken bringen. Zuletzt gibt es ein Balkendiagramm für die Statusmeldungen in Bezug auf die Hersteller.

Quellen

https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020
Zuletzt besucht am 30.03.2023 19 Uhr

http://ergast.com/mrd/
Zuletzt besucht am 30.03.2023 19 Uhr

https://de.wikipedia.org/wiki/Formel_1
Zuletzt besucht am 30.03.2023 19 Uhr

https://cran.r-project.org/web/packages/tidyr/vignettes/tidy-data.html
Zuletzt besucht am 30.03.2023 19 Uhr

Anhang

Nicht genutzte Views

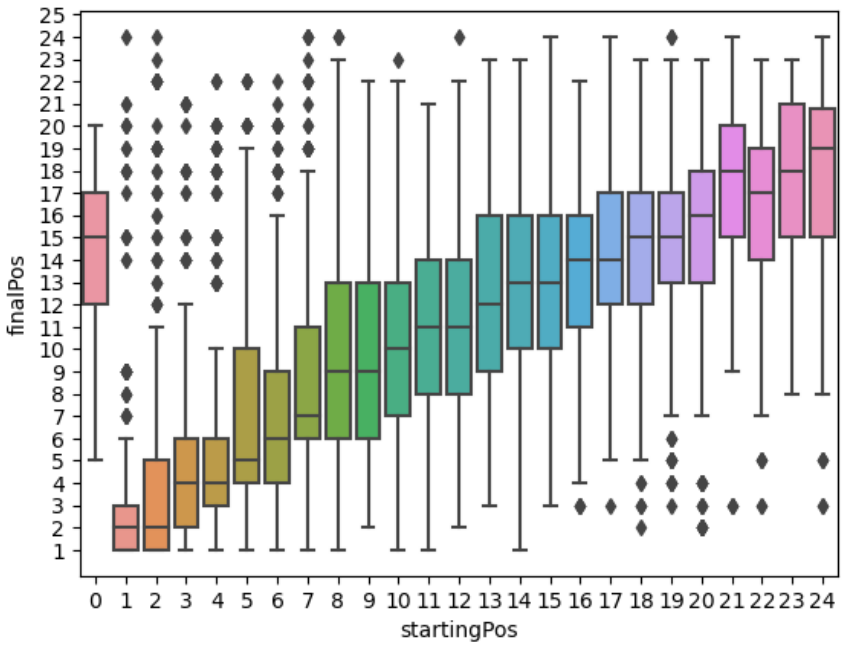
- 180_FormulaOneLapTimeAnalysis – Auflistung der einzelnen Rundenzeiten und Positionen für jeden Fahrer je Rennen
- 180_FormulaOnePitStopAnalysis - Auflistung der einzelnen Pit Stops und Stop Dauer für jeden Fahrer je Rennen
- 180_FormulaOneDriverStandingsAnalysis – Auflistung der finalen Auswertungen der Rennergebnisse für jeden Fahrer je Rennen von 1950 bis 2022
- 180_FormulaOneSpeedTimeAnalysis – SQL View, um eine Zeitreihenanalyse der Geschwindigkeiten pro Strecke zu ermöglichen.

Es wurden einige weitere Views erstellt (in DWC unter Views zu finden, beginnend mit 180_FormulaOne). Diese wurden zu Testzwecken gebaut z.B. für die Versuche die Zeitformate zu transformieren (180_FormulaOneConverted…), oder um weiter Informationen in den Daten zu prüfen (180_FormulaOneQualifying, bzw. 180_FormulaOnePitStops).

Nicht erstellte Auswertungen

Verhältnis von Start zu Endposition:
Welche Auswirkung hat die Startposition auf die Endposition in einem Rennen? Die Darstellung von Boxplots bei zwei kategorialen Attributen ist uns in SAC nicht gelungen. Wir haben immer den Hinweis erhalten, dass ein Measure notwendig ist, welches numerisch sein muss. Sobald jedoch ein Measure eingebunden wurde, waren die Boxplots so gedrungen, dass diese nicht mehr erkennbar waren.

Zum Vergleich die Abbildung auf den gleichen Daten in Python. In dieser Form konnten wir die Abbildung in SAC nicht darstellen.



Welche Strecken werden mit den kürzesten, welche mit den längsten Zeiten gefahren?

Umwandlung von Time Format in Millisekunden konnte nicht durchgeführt werden (siehe Probleme bei Umwandlung von Daten). Dadurch konnte diese Auswertung nicht angefertigt werden.

Probleme bei Umwandlung von Daten

Versuch zur Umwandlung von Millisekunden in HH:MM:SS Format per SQL View:

1. SELECT CONVERT(varchar(8), DATEADD(ms, "milliseconds", CAST('00:00:00' as time)), 108) as mytime
   FROM "180_FormulaOneLapTimes"

Message
FORMATTED CSN ERRORS ## CDS compilation failed csn.json:1:1-1: Info: CSN input had to be recompiled Warning: With option 'deprecated', many newer features are disabled <recompile>.csn:29: Error: Element "ms" has not been found (in entity:"View_1%%END%%"/column:"mytime") ## ORIGINAL CSN ERRORS ## CDS compilation failed csn.json:1:1-1: Info: CSN input had to be recompiled Warning: With option 'deprecated', many newer features are disabled <recompile>.csn:29: Error: Element "ms" has not been found (in entity:"View_1%%END%%"/column:"mytime")

2. SELECT TIME_FORMAT(SEC_TO_TIME("milliseconds" / 1000), '%H:%i:%s') AS my_time
   FROM "180_FormulaOneLapTimes"

Message
invalid name of function or procedure: TIME_FORMAT: line 2 col 3 (at pos 19), Code: 328, SQL State: HY000

Versuch Umwandlung Time Format in Millisekunden, anschließende Aggregation und rückwärts Transformation per SQL View:

1. SELECT "driverId", "raceId",
   AVG(UNIX_TIMESTAMP(TO_TIME(CONCAT('00:', SUBSTRING("time", 1, 2), ':', SUBSTRING("time", 4, 2), '.000')))) AS "lapTime"
   FROM "180_FormulaOneLapTimes"
   GROUP BY "driverId", "raceId"

Message
invalid name of function or procedure: UNIX_TIMESTAMP: line 4 col 7 (at pos 103), Code: 328, SQL State: HY000

2. SELECT "driverId", "raceId",
   SEC_TO_TIME(AVG(TIME_TO_SEC(TO_TIME("lapTime")))*1000) AS "AVGlapTime"
FROM "180_FormulaOneConvertedLapTimes"
GROUP BY "driverId", "raceId"

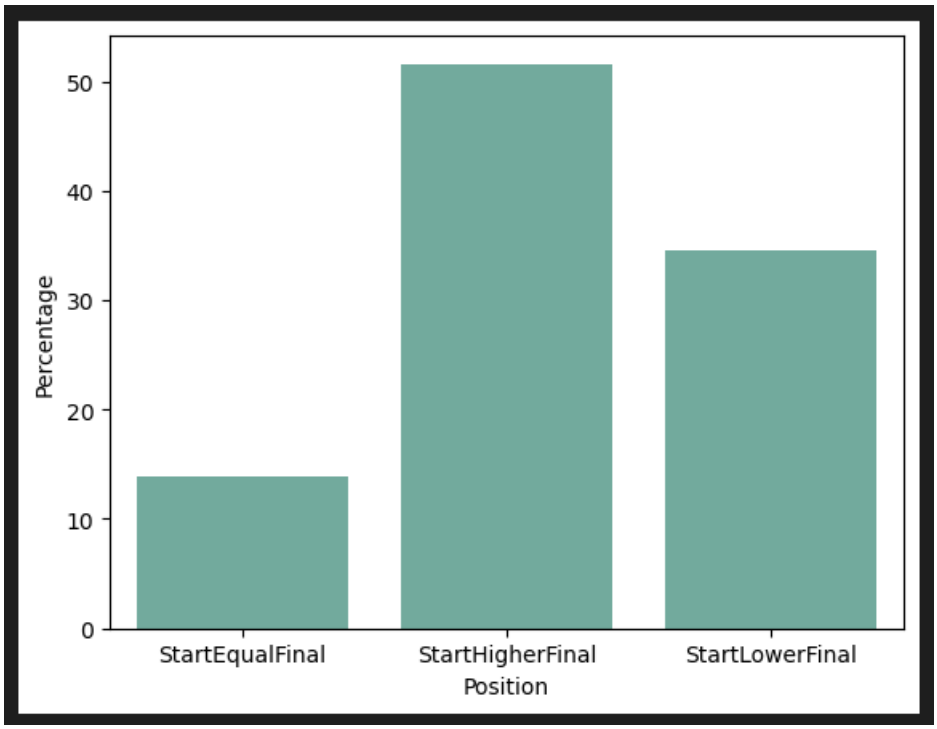
Message
invalid name of function or procedure: SEC_TO_TIME: line 4 col 3 (at pos 117), Code: 328, SQL State: HY000

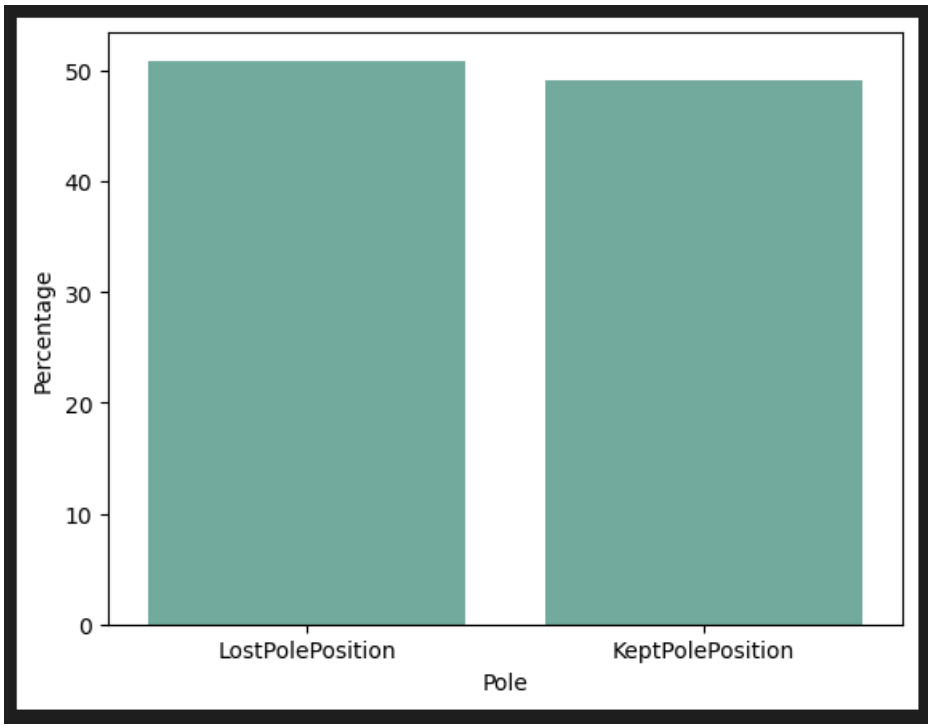
Beispielhafte Auswertungen

Zum Vergleich sind hier die Auswertungen und Fragen zu finden, die wir uns zu Beginn gestellt haben. Abweichungen davon sind aufgrund von notwendigen Anpassungen / Problemen bei der Datenaggreagation geschehen.

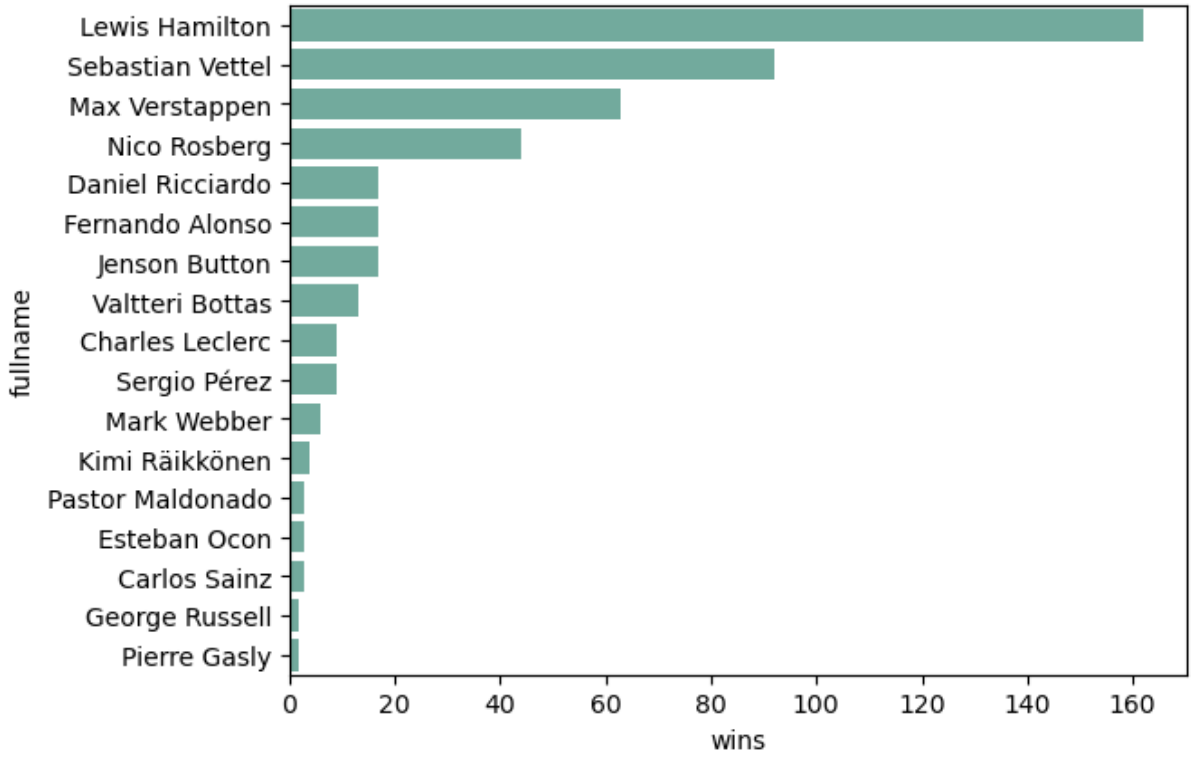
Vorschläge zur Auswertung finden sich in der DataExploration.ipynb Datei.

- Wie ist das Verhältnis zwischen Start und Endposition?





- Welcher Fahrer hat die meisten Grand Prix Rennen gewonnen?



- Welcher Hersteller hat die meisten Weltmeisterschaften gewonnen?
- Welcher Fahrer hat die meisten Weltmeisterschaften gewonnen?
- In welcher Saison sind die meisten Unfälle passiert?
- Welcher Hersteller hat die meisten Defekte / Ausfälle / Unfälle?
- Welche Strecken werden mit den kürzesten, welche mit den längsten Zeiten gefahren?