University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 10/20/2023

Group Number: 38

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Michael Tham	36082469	s6f2k	mtham01@student.ubc.ca
Christopher Wong	78318987	k4g2g	chw2305@student.ubc.ca
Maureen Marthias	83217794	w5t9j	mmarthia@student.ubc.ca

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

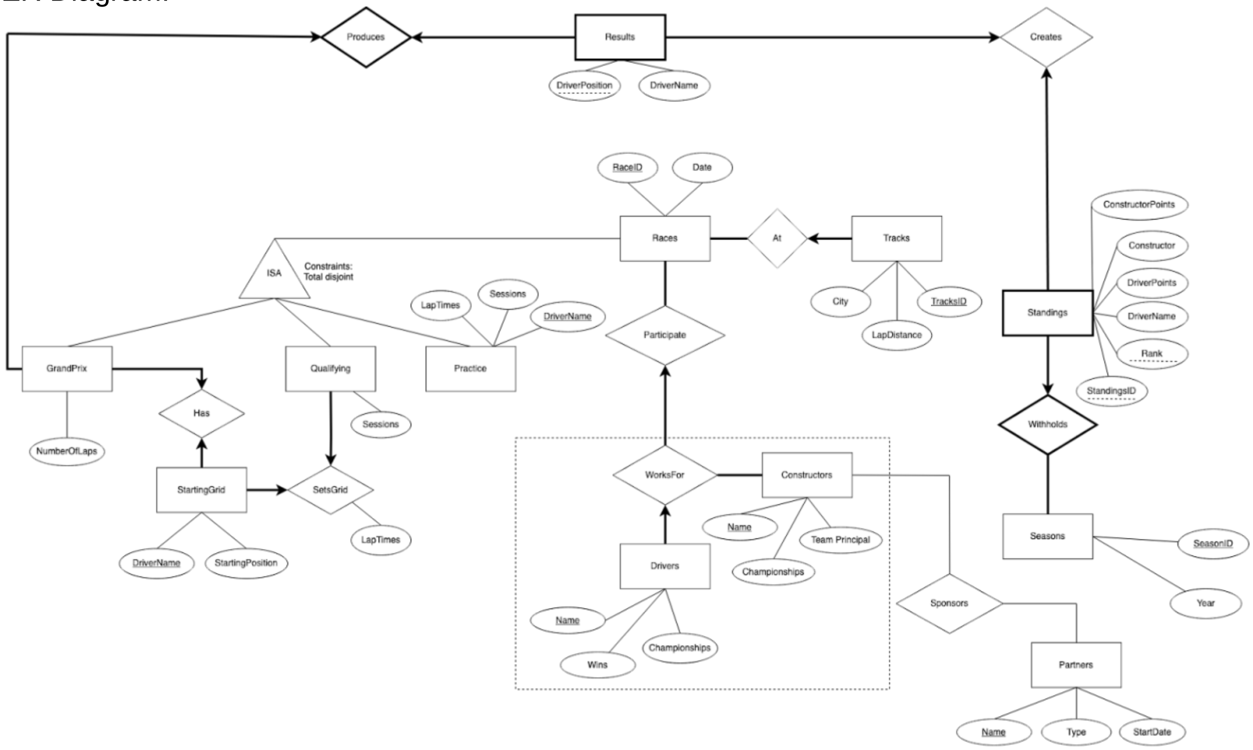
University of British Columbia, Vancouver

Department of Computer Science

Summary of project:

An application that displays relevant Formula 1 (F1) race data and standings. A user would be able to view each season described by the year the season was held (e.g. '2017') and view a multitude of data - races (location, drivers for that race, driver standings of that race), the drivers for that season and teams for that season just to name a few. Our application is targeted at avid F1 fans, just like us, who would not like the clutter of advertisements, news feeds, and different tabs.

ER Diagram:



Changes made:

- Renamed the “DateTime” attribute to “Date” and moved it to “Races” because it was redundant in the ISA and Races was missing a non-primary key attribute.
- Replaced all the IDs in the ISA (“GrandPrixID”, “QualifyingID”, “SprintID”, “PracticeID”) with “RaceID” key in “Races” to avoid redundancy and make the diagram more concise.
- Removed the “Sprint” entity because it was insignificant and inconsistent.
- Added a new entity to express the relationship between “Qualifying” and “GrandPrix” which we had overlooked in milestone 1. “Qualifying” now has a relationship “SetsGrid” to “StartingGrid”, which has a relationship to “GrandPrix”. These two relationships show that the lap times produced during a qualifying race will set the starting grid, with every driver and their respective positions, to start the Grand Prix race.
- Added “DriverName” attributes to “StartingGrid” and “Practice” to avoid storing “LapTimes” as an array and have them be assigned to each driver instead.
- Moved the “NumberOfLaps” attribute from “Tracks” to “GrandPrix” because we realized every “GrandPrix” has a specific number of laps and it is not important in the other race types. We also replaced it with “LapDistance” in “Tracks” instead.
- Removed the “Engineers” entity because it is insignificant to the main goal of our project.

- Moved the aggregation relationship to “Participate” instead of “Sponsors” because we realized that “Drivers” also participate in the races and therefore need to be in relationship with it, and “Partners” directly sponsor only the “Constructors”.
- Switched standings and seasons role, making standings the weak entity and also making the attribute ‘Rank’ a Foreign Key.

4. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:
a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.
b. Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.

- Drivers(Name: VARCHAR(50), Wins: INT(250), Championships: INT(100))
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: N/A
- Constructors(Name: VARCHAR(50), Team Principal: VARCHAR(50), Championships: INT)
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: Team Principal
- WorksFor(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**)
  - Primary Key: CName, DName
  - Foreign Key: CName, DName
  - Candidate Key: N/A
- Participate(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**, **RaceID: INT**)
  - Primary Key: CName, DName, RaceID
  - Foreign Key: CName, DName, RaceID
  - Candidate Key: N/A
- Partners(Name: VARCHAR(50), Type: VARCHAR(50), StartDate: DATE)
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: N/A
- Sponsors(**PName: VARCHAR(50), CName: VARCHAR(50)**)
  - Primary Key: PName, CName
  - Foreign Key: PName, CName
  - Candidate Key: N/A
- GrandPrix(**RaceID: INT**, NumberOfLaps: INT(100))
  - Primary Key: RaceID
  - Candidate Key: N/A
  - Foreign Key: RaceID
- Has(**RaceID: INT**, **DriverName:VARCHAR(50)**)
  - Primary Key: RaceID, DriverName
  - Candidate Key: N/A
  - Foreign Key: RaceID, DriverName
- StartingGrid(DriverName:VARCHAR(50), StartingPosition: INT(20))

    - Primary Key: DriverName
    - Candidate Key: StartingPosition
    - Foreign Key: N/A
- SetsGrid(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME)
    - Primary Key: RaceID, DriverName
    - Candidate Key: N/A
    - Foreign Key: RaceID, DriverName
- Qualifying(**RaceID: INT**, Sessions: INT(3))
    - Primary Key: RaceID
    - Candidate Key: N/A
    - Foreign Key: RaceID
- Practice(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME, Sessions: INT(3))
    - Primary Key: RaceID, DriverName
    - Candidate Key: N/A
    - Foreign Key: RaceID
- Races(RaceID: INT, Date: DATE)
    - Primary Key: RaceID
    - Candidate Key: N/A
    - Foreign Key: N/A
- At(**TracksID: INT**, **RaceID: INT**)
    - Primary Key: TracksID, RaceID
    - Candidate Key: N/A
    - Foreign Key: TracksID, RaceID
- Tracks(TracksID: INT, City: VARCHAR(50), LapDistance: INT(10))
    - Primary Key: TracksID
    - Candidate Key: N/A
    - Foreign Key: N/A
- GrandPrix_Results(**RaceID: INT**, DriverPosition: INT(30), DriverName: VARCHAR(50))
    - Primary Keys: RaceID, DriverPosition (Composite Primary Key)
    - Candidate Keys: DriverName
    - Foreign Key: RaceID
    - **Additional Information:** Results is a weak entity to GrandPrix so for the Results entity we combine both entities into 1 table
- Creates(**RaceID: INT**, **DriverPosition: INT(30), SeasonID: INT, StandingID: INT, Rank: INT(50))**
    - Primary Keys: RaceID, DriverPosition, SeasonID, StandingID, Rank (Composite Primary Key)
    - Candidate Keys: N/A
    - Foreign Key: RaceID, DriverPosition, SeasonID, StandingID, Rank
- Season_Standings(**SeasonID: INT**, StandingID: INT, Rank: INT(50), DriverName: VARCHAR(50), DriverPoints: INT, Constructor: VARCHAR(50), ConstructorPoints: INT)
    - Primary Key: SeasonID, StandingID, Rank
    - Candidate Keys: N/A
    - Foreign Key: SeasonID
    - **Additional Information:** Standings is a weak entity to Seasons so for the Standings entity, we combine both entities, Standings and Seasons into 1 table
- Seasons(SeasonID: INT, Year: YEAR)
    - Primary Keys: SeasonID
    - Candidate Keys: Year
    - Foreign Key: N/A

5. Functional Dependencies (FDs)
a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).
PKs and CKs are considered functional dependencies and should be included in the list of FDs. You do not need to include trivial FDs such as A → A.
Note: In your list of FDs, there must be some kind of valid FD other those identified by a PK or CK. If you observe that **no relations have FDs other than the PK and CK(s),** then you will have to **intentionally add some** (**meaningful**) **attributes** to show valid FDs. We want you to get a good normalization exercise.
Your design must go through a normalization process.

- Drivers(Name: VARCHAR(50), Wins: INT(250), Championships: INT(100))
    - FDs:
        - Name → Wins
        - Name → Championships
- Constructors(Name: VARCHAR(50), Team Principal: VARCHAR(50), Championships: INT)
    - FDs:
        - Name → Team Principal
        - Name → Championships
        - Team Principal → Name
- WorksFor(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**)
    - FDs:
        - CName → DName
        - DName → CName
- Participate(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**, **RaceID: INT**)
    - FDs:
        - CName → DName, RaceID
        - DName → CName, RaceID
        - RaceID → CName, DName
- Partners(Name: VARCHAR(50), Type: VARCHAR(50), StartDate: DATE)
    - FDs:
        - Name → Type
        - Name → StartDate
- Sponsors(**PName: VARCHAR(50), CName: VARCHAR(50)**))
    - FDs:
        - PName → CName
        - CName → PName
- GrandPrix(**RaceID: INT**, NumberOfLaps: INT(100))
    - FDs:
        - RaceID → NumberOfLaps
- Has(**RaceID: INT**, **DriverName:VARCHAR(50)**)
    - FDs:
        - RaceID → DriverName
- StartingGrid(DriverName:VARCHAR(50), StartingPosition: INT(20))
    - FDs:
        - DriverName → StartingPosition

    - StartingPosition → DriverName
- SetsGrid(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME)
  - FDs:
    - RaceID → DriverName
    - DriverName → LapTimes
- Qualifying(**RaceID: INT**, Sessions: INT(3))
  - FDs:
    - RaceID → Sessions
- Practice(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME, Sessions: INT(3))
  - FDs:
    - RaceID → DriverName, Sessions
    - DriverName → LapTimes
- Races(RaceID: INT, Date: DATE)
  - FDs:
    - RaceID → Date
- At(**TracksID: INT**, **RaceID: INT**)
  - FDs:
    - RaceID → TracksID
- Tracks(TracksID: INT, City: VARCHAR(50), LapDistance: INT(10))
  - FDs:
    - TracksID → City
    - City → LapDistance
- GrandPrix_Results(**RaceID: INT**, DriverPosition: INT, DriverName: VARCHAR(50))
  - FDs:
    - RaceID, DriverPosition → DriverName
- Creates(**RaceID: INT**, **DriverPosition: INT(30)**, **SeasonID: INT, StandingID: INT, Rank: INT(50))**
  - FDs:
    - RaceID → DriverPosition, SeasonID, StandingID, Rank
    - DriverPosition → RaceID, SeasonID, StandingID, Rank
    - StandingID → RaceID, DriverPosition, SeasonID, Rank
    - Rank → DriverPosition, RaceID, SeasonID, StandingID
- Season_Standings(**SeasonID: INT**, StandingID: INT, Rank: INT(50), DriverName: VARCHAR(50), DriverPoints: INT, Constructor: VARCHAR(50), ConstructorPoints: INT)
  - FDs:
    - SeasonID, StandingID, Rank → DriverName, DriverPoints, Constructor, ConstructorPoints
    - SeasonID, StandingID, DriverName → DriverPoints
    - SeasonID, StandingID, Constructor → ConstructorPoints
- Seasons(SeasonID: INT, Year: YEAR)
  - FDs:
    - SeasonID → Year

6. Normalization
a. Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization. You should show the steps taken for the decomposition. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown. The format should be the same as Step 3, with tables listed similar to Table1(attr1:domain1, attr2:domain2, ...). ALL Tables must be listed, not only the

ones post normalization.

- Drivers(Name: VARCHAR(50), Wins: INT(250), Championships: INT(100))
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: N/A
  - BCNF? **Yes**
- Constructors(Name: VARCHAR(50), Team Principal: VARCHAR(50), Championships: INT)
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: Team Principal
  - BCNF? **Yes**
- WorksFor(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**)
  - Primary Key: CName, DName
  - Foreign Key: CName, DName
  - Candidate Key: N/A
  - BCNF? **Yes**
- Participate(**CName: VARCHAR(50)**, **DName: VARCHAR(50)**, **RaceID: INT**)
  - Primary Key: CName, DName, RaceID
  - Foreign Key: CName, DName, RaceID
  - Candidate Key: N/A
  - BCNF? **Yes**
- Partners(Name: VARCHAR(50), Type: VARCHAR(50), StartDate: DATE)
  - Primary Key: Name
  - Foreign Key: N/A
  - Candidate Key: N/A
  - BCNF? **Yes**
- Sponsors(**PName: VARCHAR(50), CName: VARCHAR(50)**)
  - Primary Key: PName, CName
  - Foreign Key: PName, CName
  - Candidate Key: N/A
  - BCNF? **Yes**
- GrandPrix(**RaceID: INT**, NumberOfLaps: INT(100))
  - Primary Key: RaceID
  - Candidate Key: N/A
  - Foreign Key: RaceID
  - BCNF? **Yes**
- Has(**RaceID: INT**, **DriverName:VARCHAR(50)**)
  - Primary Key: RaceID, DriverName
  - Candidate Key: N/A
  - Foreign Key: RaceID, DriverName
  - BCNF? **Yes**
- StartingGrid(DriverName:VARCHAR(50), StartingPosition: INT(20))
  - Primary Key: DriverName
  - Candidate Key: StartingPosition
  - Foreign Key: N/A
  - BCNF? **Yes**
- SetsGrid(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME)

  - Primary Key: RaceID, DriverName
  - Candidate Key: N/A
  - Foreign Key: RaceID, DriverName
  - BCNF? **Yes**
- Qualifying(**RaceID: INT**, Sessions: INT(3))
  - Primary Key: RaceID
  - Candidate Key: N/A
  - Foreign Key: RaceID
  - BCNF? **Yes**
- Practice(**RaceID: INT**, **DriverName:VARCHAR(50)**, LapTimes: TIME, Sessions: INT(3))
  - Primary Key: RaceID, DriverName
  - Candidate Key: N/A
  - Foreign Key: RaceID
  - BCNF? **Yes**
- Races(RaceID: INT, Date: DATE)
  - Primary Key: RaceID
  - Candidate Key: N/A
  - Foreign Key: N/A
  - BCNF? **Yes**
- At(**TracksID: INT**, **RaceID: INT**)
  - Primary Key: TracksID, RaceID
  - Candidate Key: N/A
  - Foreign Key: TracksID, RaceID
  - BCNF? **Yes**
- Tracks(TracksID: INT, City: VARCHAR(50), LapDistance: INT(10))
  - Primary Key: TracksID
  - Candidate Key: N/A
  - Foreign Key: N/A
  - BCNF? **Yes**
- GrandPrix_Results(**RaceID: INT**, DriverPosition: INT(30, DriverName: VARCHAR(50))
  - FDs:
    - RaceID, DriverPosition → DriverName
  - BCNF? **Yes**
- Creates(**RaceID: INT**, **DriverPosition: INT(30), SeasonID: INT, StandingID: INT, Rank: INT(50))**
  - FDs:
    - RaceID → DriverPosition, SeasonID, StandingID, Rank
    - DriverPosition → RaceID, SeasonID, StandingID, Rank
    - StandingID → RaceID, DriverPosition, SeasonID, Rank
    - Rank → DriverPosition, RaceID, SeasonID, StandingID
  - BCNF? **Yes**
- Season_Standings(**SeasonID: INT**, StandingID: INT, Rank: INT(50), DriverName: VARCHAR(50), DriverPoints: INT, Constructor: VARCHAR(50), ConstructorPoints: INT)
  - FDs:
    - SeasonID, StandingID, Rank → DriverName, DriverPoints, Constructor, ConstructorPoints
    - SeasonID, StandingID, DriverName → DriverPoints
    - SeasonID, StandingID, Constructor → ConstructorPoints
  - BCNF? **No**

L	M	R
SeasonID StandingID Rank	DriverName DriverPoints Constructor ConstructorPoints	

- Closures:
    - SeasonID, StandingID, Rank$^+$: {SeasonID, StandingID, Rank, DriverName, DriverPoints, Constructor, ConstructorPoints}
    - SeasonID, StandingID, DriverName$^+$: {SeasonID, StandingID, DriverName, DriverPoints}
    - SeasonID, StandingID, Constructor$^+$: {SeasonID, StandingID, Constructor, ConstructorPoints}

- Therefore, SeasonID, SeasonID, StandingID, and Rank is the minimal key. However, Relation Season_Standings is not in BCNF/3NF.

- Find Minimal cover. Make RHS have only 1 attribute.
    - SeasonID, StandingID, Rank → DriverName
    - SeasonID, StandingID, Rank → DriverPoints
    - SeasonID, StandingID, Rank → Constructor
    - SeasonID, StandingID, Rank → ConstructorPoints
    - SeasonID, StandingID, DriverName → DriverPoints
    - SeasonID, StandingID, Constructor → ConstructorPoints

- Cannot Minimize LHS of any attributes, so next step, remove redundant FDs.
    - We can remove:
        - SeasonID, StandingID, Rank → DriverPoints
        - SeasonID, StandingID, Rank → ConstructorPoints
    - The SeasonID, StandingID, Rank$^+$ is still {SeasonID, StandingID, Rank, DriverName, DriverPoints, Constructor, ConstructorPoints} without considering those rules.

- Therefore, our minimal cover is:
    - SeasonID, StandingID, Rank → DriverName
    - SeasonID, StandingID, Rank → Constructor
    - SeasonID, StandingID, DriverName → DriverPoints
    - SeasonID, StandingID, Constructor → ConstructorPoints

- Decompose into 3NF using the synthesis method
    - Create a relation for reach FD in the minimal cover
        - R1(SeasonID, StandingID, Rank, DriverName)
        - R2 (SeasonID, StandingID, Rank, Constructor)

            - R3 (SeasonID, StandingID, DriverName, DriverPoints)
            - R4 (SeasonID, StandingID, Constructor, ConstructorPoints)
        - Since, our initial decomposition included the key, {SeasonID, StandingID, Rank}, we do not need to add an additional relation containing the key.

    - Final Answer:
        - Driver_Rank (SeasonID, StandingID, Rank, DriverName)
        - Constructor_Rank (SeasonID, StandingID, Rank, Constructor)
        - Driver_Points (SeasonID, StandingID, DriverName, DriverPoints)
        - Constructor_Points (SeasonID, StandingID, Constructor, ConstructorPoints)

- Seasons(<u>SeasonID: INT</u>, Year)
    - FDs:
        - SeasonID → Year
    - BCNF? **Yes**

7. The SQL DDL statements required to create all the tables from item #6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

-
  ```
  CREATE TABLE Drivers_WorksFor_Participate (
     DName VARCHAR(50) PRIMARY KEY,
     Wins INT,
     Championships INT,
     CName VARCHAR(50) NOT NULL,
     RaceID INT NOT NULL,
     FOREIGN KEY (CName) REFERENCES Constructors(Name)
                              ON DELETE CASCADE
                              ON UPDATE CASCADE,
     FOREIGN KEY (RaceID) REFERENCES Races(RaceID)
                              ON DELETE CASCADE
                              ON UPDATE CASCADE
  );
  ```

-
  ```
  CREATE TABLE Constructors (
     Name VARCHAR(50) PRIMARY KEY,
     TeamPrincipal VARCHAR(50),
     Championships INT
  );
  ```

-
  ```
  CREATE TABLE Partners (
     Name VARCHAR(50) PRIMARY KEY,
     Type VARCHAR(50),
     StartDate DATE
  );
  ```

-
  ```
  CREATE TABLE Sponsors (
     PName VARCHAR(50),
     CName VARCHAR(50),
     PRIMARY KEY (PName, CName),
     FOREIGN KEY (PName) REFERENCES Partners(Name),
     FOREIGN KEY (CName) REFERENCES Constructors(Name)
  );
  ```

-
  ```
  CREATE TABLE GrandPrix (
          RaceID INT PRIMARY KEY,
          NumberOfLaps INT,
          DriverName VARCHAR(50) NOT NULL,
          DriverPosition INT(30) NOT NULL,
          UNIQUE (DriverName, DriverPosition),
          FOREIGN KEY (RaceID) REFERENCES Races(RaceID)
  );
  ```

-
  ```
  CREATE TABLE StartingGrid (
          DriverName VARCHAR(50) PRIMARY KEY,
          StartingPosition INT,
          RaceID INT NOT NULL,
          LapTimes TIME NOT NULL,
          UNIQUE RaceID
  );
  ```

-
  ```
  CREATE TABLE Qualifying (
          RaceID INT PRIMARY KEY,
          Sessions INT(3),
          DriverName VARCHAR(50) NOT NULL,
          LapTimes TIME NOT NULL,
          UNIQUE DriverName,
          FOREIGN KEY (RaceID) REFERENCES Races(RaceID)
  );
  ```

-
  ```
  CREATE TABLE Practice (
          RaceID INT,
          DriverName VARCHAR(50),
          LapTimes TIME,
          Sessions INT(3),
          PRIMARY KEY (RaceID, DriverName),
          FOREIGN KEY (RaceID) REFERENCES Races(RaceID)
  );
  ```

-
  ```
  CREATE TABLE Races (
          RaceID INT PRIMARY KEY,
          Date DATE
  );
  ```

-
  ```
  CREATE TABLE At_Tracks (
          TracksID INT PRIMARY KEY,
          City VARCHAR(50),
          LapDistance INT(10)
          RaceID INT NOT NULL,
          FOREIGN KEY (RaceID) REFERENCES Races(RaceID)
                          ON DELETE CASCADE
                          ON UPDATE CASCADE,
  );
  ```

-
  ```
  CREATE TABLE Results(
          RaceID INT NOT NULL,
          DriverPosition INT(30),
          DriverName VARCHAR(50),
          SeasonID INT NOT NULL,
          StandingID INT NOT NULL,
          Rank INT(50) NOT NULL,
          UNIQUE (RaceID, SeasonID, StandingsID, Rank),
          PRIMARY KEY (RaceID, DriverPosition),
          FOREIGN KEY (RaceID) REFERENCES GrandPrix (RaceID)
                          ON DELETE CASCADE
                          ON UPDATE CASCADE,
  );
  ```

-
  ```
  CREATE TABLE Driver_Rank_Standings_Withholds (
          SeasonID INT,
          StandingID INT,
          Rank INT,
          DriverName VARCHAR(50),
          RaceID INT NOT NULL,
          DriverPosition INT(30) NOT NULL,
          UNIQUE(RaceID, DriverPosition),
          PRIMARY KEY (SeasonID, StandingID, Rank)
  );
  ```

-
  ```
  CREATE TABLE Constructor_Rank_ Standings_Withholds (
          SeasonID INT,
          StandingID INT,
          Rank INT,
          Constructor VARCHAR(50),
          RaceID INT NOT NULL,
          DriverPosition INT(30) NOT NULL,
          UNIQUE(RaceID, DriverPosition),
          PRIMARY KEY (SeasonID, StandingID, Rank)
  );
  ```

-
  ```
  CREATE TABLE Driver_Points_ Standings_Withholds (
          SeasonID INT,
          StandingID INT,
  ```

```
            DriverName VARCHAR(50),
            DriverPoints INT,
            RaceID INT NOT NULL,
            DriverPosition INT(30) NOT NULL,
            UNIQUE(RaceID, DriverPosition),
            PRIMARY KEY (SeasonID, StandingID)
    );
```

-
  ```
  CREATE TABLE Constructor_Points_ Standings_Withholds (
          SeasonID INT,
          StandingID INT,
          Constructor VARCHAR(50),
          ConstructorPoints INT,
          RaceID INT NOT NULL,
          DriverPosition INT(30) NOT NULL,
          UNIQUE(RaceID, DriverPosition),
          PRIMARY KEY (SeasonID, StandingID)
  );
  ```

-
  ```
  CREATE TABLE Seasons (
          SeasonID INT,
          Year INT,
          PRIMARY KEY (SeasonID)
  );
  ```

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later on.
Note: Be consistent with the names used in your ER diagram, schema, and FDs. Make a note if the name has been intentionally changed.

```
INSERT INTO Drivers (Name, Wins, Championships)
VALUES
   ('Lewis Hamilton', 103, 7),
   ('Charles Leclerc', 5, 0),
   ('Max Verstappen', 49, 3),
   ('Lando Norris', 0, 0),
   ('Daniel Ricciardo.', 8, 0);

INSERT INTO Constructors (Name, TeamPrincipal, Championships)
VALUES
   ('Mercedes-AMG Petronas F1 Team', 'Toto Wolff', 9),
   ('Red Bull Racing', 'Christian Horner', 6),
   ('Scuderia Ferrari', 'Mattia Binotto', 16),
   ('McLaren F1 Team', 'Zak Brown', 8),
   ('Scuderia AlphaTauri', 'Laurent Mekies', 0);

INSERT INTO WorksFor (CName, DName)
VALUES
   ('Mercedes-AMG Petronas F1 Team', 'Lewis Hamilton'),
```

```
   ('Red Bull Racing', 'Max Verstappen'),
   ('Scuderia Ferrari', 'Charles Leclerc'),
   ('Scuderia AlphaTauri', 'Daniel Ricciardo'),
   ('McLaren F1 Team', 'Lando Norris');

INSERT INTO Participate (CName, DName, RaceID)
VALUES
   ('Mercedes-AMG Petronas F1 Team', 'Lewis Hamilton', 1),
   ('Red Bull Racing', 'Max Verstappen', 1),
   ('Scuderia Ferrari', 'Charles Leclerc', 1),
   ('Scuderia AlphaTauri', 'Daniel Ricciardo', 1),
   ('McLaren F1 Team', 'Lando Norris', 1)
;

INSERT INTO Partners (Name, Type, StartDate)
VALUES
   ('SponsorX', 'Title Sponsor', '2022-01-01'),
   ('SponsorY', 'Official Partner', '2021-03-15'),
   ('SponsorZ', 'Technical Partner', '2022-05-20'),
   ('SponsorA', 'Official Supplier', '2020-12-10'),
   ('SponsorB', 'Official Sponsor', '2021-08-05');

INSERT INTO Sponsors (PName, CName)
VALUES
   ('SponsorX', 'Mercedes-AMG Petronas F1 Team'),
   ('SponsorY', 'Red Bull Racing'),
   ('SponsorZ', 'Scuderia Ferrari'),
   ('SponsorA', 'Scuderia AlphaTauri'),
   ('SponsorB', 'McLaren F1 Team');

INSERT INTO GrandPrix (RaceID, NumberOfLaps)
VALUES
  (1, 53),
  (2, 44),
  (3, 56),
  (4, 47),
  (5, 61);

INSERT INTO Has (RaceID, DriverName)
VALUES
  (1, 'Lewis Hamilton'),
  (1, 'Charles Leclerc'),
  (1, 'Max Verstappen’),
  (1, 'Lando Norris'),
  (1, 'Daniel Ricciardo');

INSERT INTO StartingGrid (DriverName, StartingPosition)
VALUES
  ('Lewis Hamilton', 1),
  ('Charles Leclerc', 2),
  ('Max Verstappen', 6),
  ('Lando Norris', 10),
  ('Daniel Ricciardo', 15);

INSERT INTO Qualifying (RaceID, Sessions)
VALUES
  (1, 1),
  (2, 3),
  (3, 2),
  (4, 3),
  (5, 2)
;

INSERT INTO Practice (RaceID, DriverName, LapTimes, Sessions)
VALUES
  (1, 'Lewis Hamilton', 123, ‘00:01:15’),
  (1, 'Charles Leclerc', 124, ‘00:01:45’),
  (1, 'Max Verstappen', 125, ‘00:01:05’),
  (1, 'Lando Norris', 126, ‘00:01:55’),
  (1, 'Daniel Ricciardo', 127, ‘00:01:32’);

INSERT INTO Races (RaceID, Date)
VALUES
  (1, '2023-03-31'),
  (2, '2023-04-02'),
  (3, '2023-06-16'),
  (4, '2023-06-17'),
  (5, '2023-08-20');



INSERT INTO At (TracksID, RaceID)
VALUES
  (1, 1),
  (1, 2),
  (2, 3),
  (2, 4),
  (3, 5);

INSERT INTO Tracks (TracksID, City, LapDistance)
VALUES
  (1, 'Melbourne', 5.278),
  (2, 'Montreal', 4.361),
  (3, 'Austin', 5.513),
  (4, 'Singapore', 4.94),
  (5, 'Barcelona', 4.657);

INSERT INTO GrandPrix_Results (RaceID, DriverPosition, DriverName)
```

```
VALUES
  (1, 1, 'Lewis Hamilton'),
  (1, 2, 'Max Verstappen'),
  (1, 3, 'Charles Leclerc'),
  (1, 4, 'Daniel Riccardo’),
  (1, 5, ‘Lando Norris’)
;

INSERT INTO Creates (RaceID, DriverPosition, SeasonID, StandingID, Rank)
VALUES
  (1, 1, 1, 1, 1),
  (1, 2, 1, 1, 2),
  (1, 3, 1, 1, 3),
  (1, 4, 1, 1, 4),
  (1, 5, 1, 1, 5)
;

INSERT INTO Driver_Rank (SeasonID, StandingID, Rank, DriverName)
VALUES
  (1, 1, 1, 'Lewis Hamilton'),
  (1, 1, 2, 'Max Verstappen'),
  (1, 1, 3, ‘Charles Leclerc’)
  (1, 1, 4, 'Daniel Riccardo’),
  (1, 1, 5, ‘Lando Norris’)
;

INSERT INTO Constructor_Rank (SeasonID, StandingID, Rank, Constructor)
VALUES
  (1, 1, 1, 'Mercedes-AMG Petronas F1 Team'),
  (1, 1, 2, 'Red Bull Racing'),
  (1, 1, 3, 'Scuderia Ferrari'),
  (1, 1, 3, ‘McLaren F1 Team’),
  (1, 1, 3, ‘Scuderia AlphaTauri’)
;

INSERT INTO Driver_Points (SeasonID, StandingID, DriverName, DriverPoints)
VALUES
VALUES
  (1, 1, 'Lewis Hamilton', 385),
  (1, 1, 'Max Verstappen', 265),
  (1, 1, ‘Charles Leclerc’, 200)
  (1, 1, 'Daniel Riccardo’, 190),
  (1, 1, ‘Lando Norris’, 150)
;

INSERT INTO Constructor_Points (SeasonID, StandingID, Constructor, ConstructorPoints)
VALUES
  (1, 1, 'Mercedes-AMG Petronas F1 Team', 300),
  (1, 1, 'Red Bull Racing', 261),
  (1, 1, 'Scuderia Ferrari', 200),
  (1, 1, ‘McLaren F1 Team’, 150),
  (1, 1, ‘Scuderia AlphaTauri’, 100)
;

INSERT INTO Seasons (SeasonID, Year)
VALUES
  (1, 2001)
  (2, 2002)
  (3, 2003)
  (4, 2004)
  (5, 2005)
;
```