NFL SEASON

Designed, Implemented, and Summarized by

Brian Ellis

For Dr. Nguyen
COM330 Database Concepts and Programming
Saint Leo University

PROJECT DESCRIPTION:
(Parts 1 & 2 Completed by Brian Ellis)

The goal of this project will be to relate the various components of an NFL season to one another in a well-designed database. This will allow any user of the database to easily and efficiently access information about the players, coaching staff, owners, leagues, games, statistics, or any other entities involved in the happenings of an NFL season, through one centralized location.

This database will be created with the names and positions of each person affiliated with an organization and show the season schedule for a team during the entire year. It will also contain numerous offensive, defensive, and special team stats for each team in the database. The data schema will be modeled on the relational structure of the NFL season. This will be represented in an ERD format using Crow's Foot Notation. This data schema will show the multiple dimensions of the entities associated with the NFL season.

In this database each table may be linked to one or many different tables by showing the correspondences between data fields in each table. These relationships will represent the business rules set forth by the NFL Leagues and the constraints associated with those rules. Incorporated in the database will be super-type entities, subtype entities, and dependent entities. The tables will be set up using the 1NF, 2NF, and 3NF methods. Certain employees and performance statistics of the database will share similar characteristics; therefore, they will be put into the most useful Normal Form to avoid redundancies in the information.

The database will be set up so that it can easily be updated and/or extended based on the needs of the NFL season. This will allow us to maintain data integrity of the stored information while allowing for us to put in new information.

DESIGN:
(Parts 3a, 3b, 3c, & 3d Completed by Brian Ellis)

The first steps in designing this database are recognizing and identifying entities that are associated with the National Football League as a whole, and identifying the Business Rules that are associated with an NFL Season. In order to do this, research was conducted on many NFL websites and reference materials to identify all of the entities involved. Then those entities were narrowed down to the primary divisions that would be used in the database. It was not feasible to use all of the entities in the NFL in this project, as the timeframe allotted did not allow for it.

The entities that were chosen for this paper were: (1) League, (2) Owner, (3) Team, (4) Employee, (5) Coach, (6) Player, (7) Schedule, (8) Game, (9) Season, (10) Performance, (11) Defense, (12) Offense, and (13) Special Teams.

After selecting the primary entities that would be used in this database, the next step was to determine the business rules involved with those entities and how they would relate to one another. In this step, the entities were given the necessary attributes associated with each one, and then based on those attributes, business rules were created to show the correlation between the entities. The business rules developed were kept relatively simple, in that there were only so many relations that were to be used for this project. In a real-world environment, the business rules and the entities

for this database would be much more complex. The Business Rules used in the database are as follows:

Each LEAGUE has many OWNERS,
Each OWNER belongs to one LEAGUE.

Each LEAGUE hosts many TEAMS,
Each TEAM belongs to one LEAGUE.

Each OWNER owns one or more TEAMS,
Each TEAM is owned by one OWNER.

Each TEAM hires many EMPLOYEES,
Each EMPLOYEE works for one TEAM.

Each COACH is a subtype of the EMPLOYEE super-type.

Each PLAYER is a subtype of the EMPLOYEE super-type.

Each TEAM plans many SCHEDULED_GAMES,
Each SCHEDULED_GAME is planned by one TEAM.

Each GAME is from more than one team's SCHEDULE,
Each SCHEDULE instance is for one GAME.

Each SEASON hosts many GAMES,
Each GAME takes place in one SEASON.

Each TEAM generates many PERFORMANCE_STATISTICS,
Each PERFORMANCE_STATISTIC is generated by one TEAM.

Each SEASON results in many PERFORMANCE_STATISTICS,
Each PERFORMANCE_STATISTIC is the result of one SEASON.

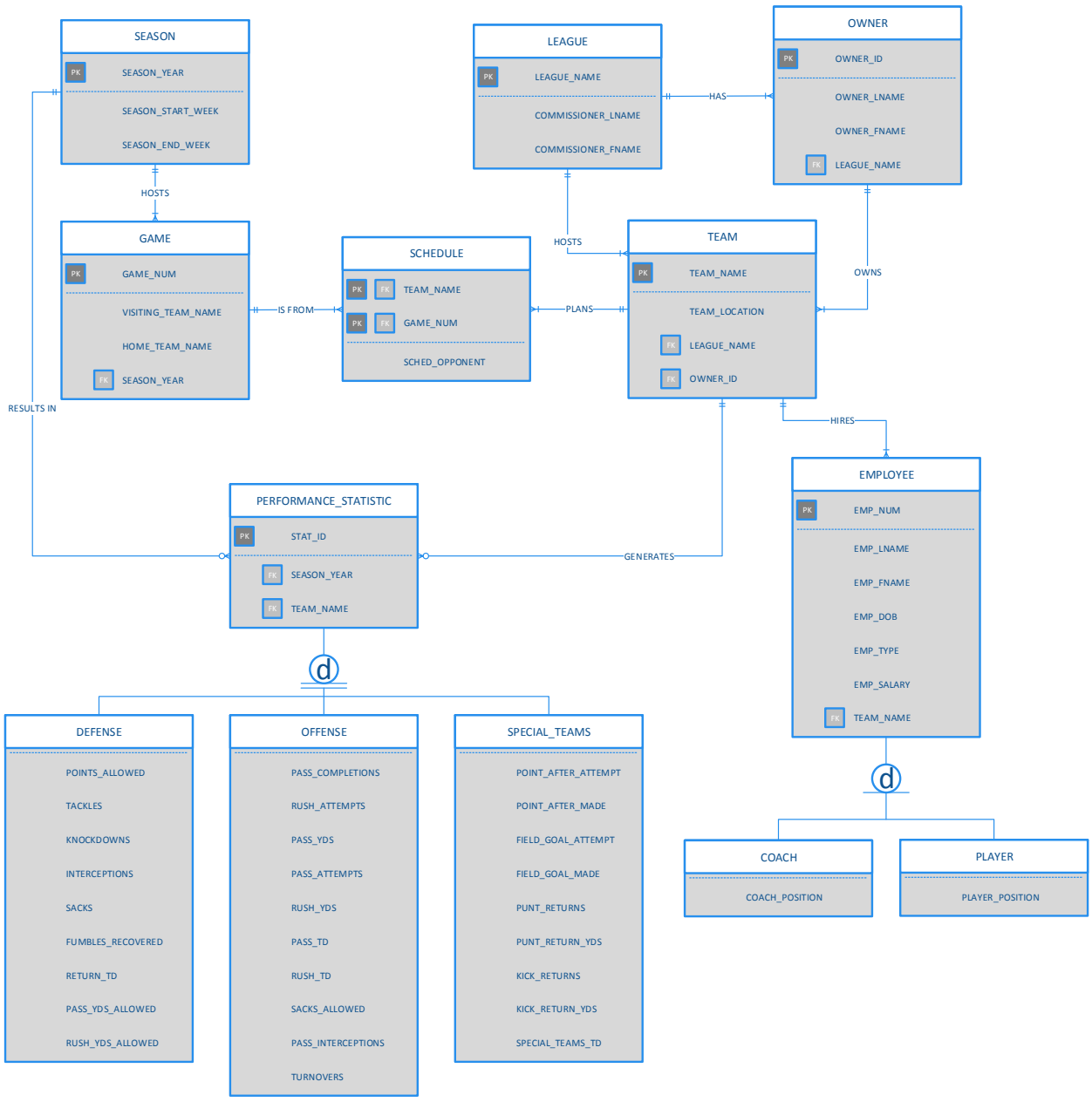
Each DEFENSE_PERFORMANCE is a subtype of the PERFORMANCE_STATISTIC super-type.

Each OFFENSE_PERFORMANCE is a subtype of the PERFORMANCE_STATISTIC super-type.

Each SPECIAL_TEAMS_PERFORMANCE is a subtype of the PERFORMANCE_STATISTIC super-type.

The next step in designing the database was drawing out the ERD for the selected entities and business rules. The design of the ERD for this project was relatively simple; however there were some hurdles to overcome. Early in the design, it was realized that there was a "many to many" relationship between the GAME and TEAM entities. In order to break this up, the SCHEDULE entity was designed to create two (2) one to many relationships among these tables. This allowed for the data to be sorted more effectively and efficiently by the end user of the database.

Some other relationships that needed to be looked at were the relationships of EMPLOYEES to COACHES and PLAYERS, as well as the relationships of PERFORMANCE to DEFENSE, OFFENSE, and SPECIAL TEAMS. It was determined that the EMPLOYEE table must become a super-type and the PLAYER and COACH tables must become its subtypes. Also, the PERFORMANCE table was to become a super-type as well, and the DEFENSE, OFFENSE, and SPECIAL TEAMS were to become its subtypes. Basically, these were the only challenges faced when designing the ERD for this database. All of the other entities were relatively straight-forward. The ERD created for this database is as follows:



The last step in the design process was the normalization of the tables. After separating the tables out and going through them one by one, it was relatively simple to get them into 3NF. Each table had been well thought out prior to this step, so there was very little that needed to be done to make this happen. However, after conducting this

process, it was determined that the database was as efficient as it could possibly be based on the data that was to be used to populate it.

IMPLEMENTATION:
(Parts 4a & 4b Completed by Brian Ellis)

The implementation process for this database was really started in the initial design of the database. When designing the tables, the business rules, and determining relations for this database, much thought was given to how it would work once it was put into SQL. By doing this, redundancy and null values were reduced as much as possible and the design was able to run as efficiently as possible. Once the design was set, and approved, it was time create the tables and types and populate them with data to see how well it worked in Oracle Express.

Initially, the three main tables of this database, the League, Owner, and Team tables were created and formatted. These entities are really the cornerstone of the design. Next, the Employee section of the database was implemented. This was a little more difficult with the limited experience that I had in SQL. The objective was to create the Employee table as a super-type and create the Coach and Player tables as subtypes. It looked simple enough on paper, but when it came to implementing this it took a lot of trial and error, and a lot of hours in Oracle. Ultimately, the EMP type, which is the super-type of the structure, was created. Then the COACH type under the EMP type and the PLAYER type under the EMP type were created to complete the EMP type section of the database. Now that the super-type and subtypes in place, the EMPLOYEE table was created to hold the values for all employees whether they were a

staff member, coach, or player. Next the data values were inserted into the table using the specific type for each individual.

After conquering that obstacle, the next steps were to create the Schedule, Game, and Season tables. These tables were relatively straight forward and only required the basic Create Table clause and setting up the Primary and Foreign keys for each entity. After that, the PERF type was created.

Much like the EMP type, the PERF type contained a super-type. However, the PERF type would contain three subtypes. With this, the PERF type was created containing common values to all of the subtypes. Then the DEFENSE, OFFENSE, and SPECIAL_TEAMS types were created under the PERF type. After completing the set-up of these specific types, the PERFORMANCE table was then created to house the data associated with these statistics. Once this was complete, the data values were inserted into this table by using the specified type for each instance.

Ultimately, at the end, the database is created exactly as it was originally planned, and it works as it is supposed to. Everything in the implementation phase of this project fell into place nicely and relatively easily since the original design was well thought out.

SAMPLE QUERIES AND RESULTS:
(Part 5 Completed by Brian Ellis)

Now, it was time for the ultimate challenge: Creating queries based on the data and design of the database. There was really no doubt that the database would respond to the queries as it was supposed to, given the great lengths that were put into place

during the design and implementation phase. So it was time to enter the first query and see what happens.

The first query was one in which the user could request the league, team, and schedule information for a team where the game number attribute in the schedule was less than 8500. This query worked perfectly. It showed that the possibility of selecting individual games and teams, or a range of games and teams was available to the user. The results were first ordered by team name and then game number to make them easier for the user to decipher.

The second query was used to join the schedule and game tables for games that took place in the 2005 season. This worked as it was supposed to. This would allow a user to see the specific schedule for a specific team during a specific season year. The results on this query were ordered by season year and then team name to make it easier for the user to read.

Query number three was designed to join the league, team, owner, and schedule tables for the team located in Hartford, Connecticut. This query worked perfectly. It shows that a user can find the information of scheduled games for any team based on their location. This query was naturally ordered by the team name and then the game number so it was not necessary to change the order for easier viewing.

The fourth query naturally joined the league and team tables and then joined the performance table using the team name. It displayed the data for the defensive performance statistics of the team named Flames. This query worked as it was supposed to. This query will allow the user to see specific statistics for a specific team

based on the team name. The natural order of the results was the team name and then the stat id number. Therefore it was not necessary to re-order this view.

In the fifth query, the user would be able to display the special team statistics for the season years 2005, 2006, & 2007 for the team named Tractors. This was accomplished by joining the season and game tables naturally and joining the performance table using the season year. Also, the "between" clause was used to specify the range of season years to be located. This query worked well. The results were naturally ordered by statistic number and then game number for easy viewing.

Query number six was used to show the owner, team, and employee information for the employee with the highest salary. This was done by using the "max" calculation. All of the employee salaries were compared to one another, and the highest was chosen. Then the database displayed his/her info as well as team and owner info associated with this person. The query worked perfectly. The results did not need to be ordered as there was only one selection made.

The seventh query was similar to number six; however it focused all the employees of the team Arches that had a salary that was higher than the average for all of the NFL employees. This was accomplished by using the "average" calculation to first find the average of all employees and then it displayed the employees of the Arches franchise who made a higher salary than that. This query worked well. There was no need to order it because it was already naturally ordered by employee number.

Query number eight joined the owner, team, and employee tables to display all of the NFL employees that were not coaches or players. This was done by using the

"not in" clause to find employee types that were not coach or player. The query worked well. The view was ordered by team name and then employee last name to make viewing the results easier.

The ninth query was used to display specific league, team, and owner attributes for each team in the database. This was done by selecting the specific columns, and then joining the tables together based on their relations. This query worked as it was supposed to. The results were ordered by league name and then team name to make viewing easier for the user.

Query number ten was a simple query to display league, team, and employee information for all of the employees of the team Desert. It showed how the user can obtain employee information about any specific team or a range of teams. This query worked perfectly. The results were ordered by the employee last name for easier viewing.

The last query, query number eleven. This was another simple query used to display the league, team, performance, and owner information for the team Wilderness. This showed how a user could pull all of the statistics for a specific team and relate them back to the league and owner of that team. The query worked well. The results were naturally easy to view so there was no need to order them.

SUMMARY:
(Part 6 Completed by Brian Ellis)

This is truly one of the best projects I have done during my three years of college. I was really involved in making it as great as I knew it could be. Many of the projects

that are done for classes are very ordinary, and really don't teach you anything that you will use in the real world after college. This one however does.

I feel that with this project I was able to see a glimpse of what life would be like for a database designer in the corporate world. It allowed me to get hands on training in a programming language that is necessary for the IT field in which my degree will be in. Basically, this project is very similar to some of the on the job training I would expect to encounter once I have finished with my collegiate studies.

Although many people said to not create such a complex database for this project, I wanted to push myself past the simple, mundane database that I could have created just to get a grade. I feel that this has helped me to understand the advanced parts of the SQL language at a more rapid pace than I originally would have. I now feel very confident in my base knowledge of this programming language, and hope to develop on this knowledge in the future.

I don't know if I will go into database design in the future, but it is great that I got to do this project to give me a base from which I can expand if that is a career path I decide to choose. I feel that this class as a whole was very important in the development of my skills in the IT industry.

APPENDIX A: TABLE OF ENTITIES AND ATTRIBUTES

NFL FOOTBALL DATABASE DATA DICTIONARY								
TABLE NAME	ATTRIBUTE NAME	CONTENTS	TYPE	FORMAT	RANGE	REQUIRED	PK OR FK	FK REFERENCED TABLE
LEAGUE	LEAGUE_NAME	League Name	varchar2(3)	XXX	NA	Y	PK	
	COMMISSIONER_LNAME	Commissioner Last Name	varchar2(20)	Xxxxxxx	NA	Y		
	COMMISSIONER_FNAME	Commissioner First Name	varchar2(20)	Xxxxxxx	NA	Y		
OWNER	OWNER_ID	Owner ID	integer	#####	NA	Y	PK	
	OWNER_LNAME	Owner Last Name	varchar2(20)	Xxxxxxxx	NA	Y		
	OWNER_FNAME	Owner First Name	varchar2(20)	Xxxxxxxx	NA	Y		
	LEAGUE_NAME	League Name	varchar2(3)	XXX	NA	Y	FK	LEAGUE
TEAM	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	PK	
	TEAM_LOCATION	City/State	varchar2(35)	Xxxx/Xxxx	NA	Y		
	LEAGUE_NAME	League Name	varchar2(3)	XXX	NA	Y	FK	LEAGUE
	OWNER_ID	Owner ID	integer	#####	NA	Y	FK	OWNER
EMPLOYEE	EMP_NUM	Employee Number	char(5)	XX###	NA	Y	PK	
	EMP_LNAME	Employee Last Name	varchar2(20)	Xxxxxxx	NA	Y		
	EMP_FNAME	Employee First Name	varchar2(20)	Xxxxxxx	NA	Y		
	EMP_DOB	Employee Date of Birth	date	MM/DD/YYYY	NA	Y		
	EMP_TYPE	Type of Employee	varchar2(20)	Xxxxxxx	NA	Y		
	EMP_SALARY	Employee Annual Salary	number(10,2)	####.##	0-99999999.99	Y		
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
COACH	COACH_POSITION	Coach Position	varchar2(20)	Xxxxxxx	NA	Y		
	EMP_NUM	Employee Number	char(5)	XX###	NA	Y	PK	EMPLOYEE
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
PLAYER	PLAYER_POSITION	Player Position	varchar2(20)	Xxxxxxx	NA	Y		
	EMP_NUM	Employee Number	char(5)	XX###	NA	Y	PK	EMPLOYEE
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
SEASON	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	PK	
	SEASON_START_WEEK	Season Start	date	MM/DD/YYYY	01/01-12/31	Y		
	SEASON_END_WEEK	Season End	date	MM/DD/YYYY	01/01-12/31	Y		
GAME	GAME_NUM	Game Number	number	###	NA	Y	PK	
	VISITING_TEAM_NAME	Visiting Team	varchar2(35)	Xxxxxxx	NA	Y		
	HOME_TEAM_NAME	Home Team	varchar2(35)	Xxxxxxx	NA	Y		
	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	FK	SEASON
SCHEDULE	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	PK,FK	TEAM
	GAME_NUM	Game Number	number	###	NA	Y	PK,FK	GAME
	SCHED_OPPONENT	Opponent	varchar2(35)	Xxxxxxx	NA	Y		

NFL FOOTBALL DATABASE DATA DICTIONARY								
TABLE NAME	ATTRIBUTE NAME	CONTENTS	TYPE	FORMAT	RANGE	REQUIRED	PK OR FK	FK REFERENCED TABLE
PERFORMANCE	STAT_ID	Stat Identifier	char(5)	XXXXX	NA	Y	PK	
	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	FK	SEASON
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
DEFENSE_PERFORMANCE	POINTS_ALLOWED	Points Allowed	integer	###	0-200	Y		
	TACKLES	Tackles Made	integer	##	0-99	Y		
	KNOCKDOWNS	Knockdowns	integer	##	0-99	Y		
	SACKS	Sacks Made	integer	##	0-99	Y		
	INTERCEPTIONS	Interceptions Made	integer	##	0-99	Y		
	FUMBLES_RECOVERED	Fumbles Recovered	integer	##	0-99	Y		
	RETURN_TD	Defensive Touchdowns	integer	##	0-99	Y		
	PASS_YDS_ALLOWED	Passing Yards Allowed	integer	##	0-9999.99	Y		
	RUSH_YDS_ALLOWED	Rushing Yards Allowed	integer	##	0-9999.99	Y		
	STAT_ID	Stat Identifier	char(5)	XXXXX	NA	Y	PK	
	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	FK	SEASON
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
OFFENSE_PERFORMANCE	PASS_ATTEMPTS	Passes Attempted	integer	###	0-999	Y		
	PASS_COMPLETIONS	Passes Completed	integer	###	0-999	Y		
	PASS_YDS	Passing Yards	integer	###	0-9999.99	Y		
	RUSH_ATTEMPTS	Rush Attempts	integer	###	0-999	Y		
	RUSH_YDS	Rushing Yards	integer	###	0-9999.99	Y		
	PASS_TD	Passing Touchdowns	integer	##	0-99	Y		
	RUSH_TD	Rushing Touchdowns	integer	##	0-99	Y		
	SACKS_ALLOWED	Sacks Allowed	integer	##	0-99	Y		
	PASS_INTERCEPTIONS	Offensive Interceptions	integer	##	0-99	Y		
	TURNOVERS	Turnovers	integer	##	0-99	Y		
	STAT_ID	Stat Identifier	char(5)	XXXXX	NA	Y	PK	
	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	FK	SEASON
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM
SPECIAL_TEAMS_PERFORMANCE	POINT_AFTER_ATTEMPT	Point After Attempts	integer	##	0-99	Y		
	POINT_AFTER_MADE	Point After Made	integer	##	0-99	Y		
	FIELD_GOAL_ATTEMPT	Field Goals Attempted	integer	##	0-99	Y		
	FIELD_GOAL_MADE	Field Goals Made	integer	##	0-99	Y		
	PUNT_RETURNS	Punt Returns	integer	###	0-99	Y		
	PUNT_RETURN_YDS	Punt Return Yards	integer	###	0-9999.99	Y		
	KICK_RETURNS	Kick Returns	integer	###	0-99	Y		
	KICK_RETURN_YDS	Kick Return Yards	integer	###	0-9999.99	Y		
	SPECIAL_TEAMS_TD	Special Teams Touchdowns	integer	##	0-99	Y		
	STAT_ID	Stat Identifier	char(5)	XXXXX	NA	Y	PK	
	SEASON_YEAR	Season Year	integer	YYYY	1966-2014	Y	FK	SEASON
	TEAM_NAME	Team Name	varchar2(35)	Xxxxxxx	NA	Y	FK	TEAM

APPENDIX B: NORMALIZATION PROCESS

NFL Season Database Normalization

PRIMARY KEY = FOREIGN KEY = PRIMARY & FOREIGN KEY =

TABLE NAME	ATTRIBUTES													NORMAL FORM
LEAGUE:														
	LEAGUE_NAME	COMMISSIONER_LNAME	COMMISSIONER_FNAME											3NF
OWNER:														
	OWNER_ID	OWNER_LNAME	OWNER_FNAME	LEAGUE_NAME										3NF
TEAM:														
	TEAM_NAME	TEAM_LOCATION	LEAGUE_NAME	OWNER_ID										3NF
EMPLOYEE:														
	EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_DOB	EMP_TYPE	EMP_SALARY	TEAM_NAME							3NF
COACH:														
	EMP_NUM	COACH_POSITION	TEAM_NAME											3NF
PLAYER:														
	EMP_NUM	PLAYER_POSITION	TEAM_NAME											3NF
SEASON:														
	SEASON_YEAR	SEASON_START_WEEK	SEASON_END_WEEK											3NF
GAME:														
	GAME_NUM	VISITING_TEAM_NAME	HOME_TEAM_NAME	SEASON_YEAR										3NF
SCHEDULE:														
	TEAM_NAME	GAME_NUM	SCHED_OPPONENT											3NF
PERFORMANCE_STATISTIC														
	STAT_ID	SEASON_YEAR	TEAM_NAME											3NF
DEFENSE_PERFORMANCE														
	STAT_ID	SEASON_YEAR	TEAM_NAME	POINTS_ALLOWED	TACKLES	KNOCKDOWNS	SACKS	INTERCEPTIONS	FUMBLES_RECOVERED	RETURN_TD	PASS_YDS_ALLOWED	RUSH_YDS_ALLOWED		3NF
OFFENSE_PERFORMANCE														
	STAT_ID	SEASON_YEAR	TEAM_NAME	PASS_ATTEMPTS	PASS_COMPLETIONS	PASS_YDS	RUSH_ATTEMPTS	RUSH_YDS	PASS_TD	RUSH_TD	SACKS_ALLOWED	PASS_INTERCEPTIONS	TURNOVERS	3NF
SPECIAL_TEAMS_PERFORMANCE														
	STAT_ID	SEASON_YEAR	TEAM_NAME	POINT_AFTER_ATTEMPT	POINT_AFTER_MADE	FIELD_GOAL_ATTEMPT	FIELD_GOAL_MADE	PUNT_RETURNS	PUNT_RETURN_YDS	KICK_RETURNS	KICK_RETURN_YDS	SPECIAL_TEAMS_TD		3NF

APPENDIX C: METADATA FOR TABLES

```
SQL> DESCRIBE LEAGUE;
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 LEAGUE_NAME               NOT NULL VARCHAR2(3)
 COMMISSIONER_LNAME                 VARCHAR2(20)
 COMMISSIONER_FNAME                 VARCHAR2(20)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'LEAGUE';

CONSTRAINT_NAME              C
------------------------------ -
SYS_C007183                  P

SQL> DESCRIBE OWNER;
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OWNER_ID                  NOT NULL NUMBER(38)
 OWNER_LNAME                        VARCHAR2(20)
 OWNER_FNAME                        VARCHAR2(20)
 LEAGUE_NAME                        VARCHAR2(3)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'OWNER';

CONSTRAINT_NAME              C
------------------------------ -
SYS_C007184                  P
FK_OWNER                     R

SQL> DESCRIBE TEAM;
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TEAM_NAME                 NOT NULL VARCHAR2(35)
 TEAM_LOCATION                      VARCHAR2(35)
 LEAGUE_NAME                        VARCHAR2(3)
 OWNER_ID                           NUMBER(38)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'TEAM';

CONSTRAINT_NAME              C
------------------------------ -
SYS_C007186                  P
FK_TEAM                      R
FK_TEAM2                     R

SQL> DESCRIBE EMPLOYEE;
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_NUM                   NOT NULL CHAR(5)
 EMP_LNAME                          VARCHAR2(20)
 EMP_FNAME                          VARCHAR2(20)
 EMP_DOB                            DATE
 EMP_TYPE                           VARCHAR2(20)
 EMP_SALARY                         NUMBER(10,2)
 TEAM_NAME                          VARCHAR2(35)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'EMPLOYEE';

CONSTRAINT_NAME              C
------------------------------ -
SYS_C007189                  U
SYS_C007190                  P
FK_EMPLOYEE                  R

SQL> DESCRIBE EMP;
 EMP is NOT FINAL
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_NUM                            CHAR(5)
 EMP_LNAME                          VARCHAR2(20)
 EMP_FNAME                          VARCHAR2(20)
 EMP_DOB                            DATE
 EMP_TYPE                           VARCHAR2(20)
 EMP_SALARY                         NUMBER(10,2)
 TEAM_NAME                          VARCHAR2(35)

SQL> DESCRIBE COACH;
 COACH extends SYSTEM.EMP
 COACH is NOT FINAL
 Name                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMP_NUM                            CHAR(5)
 EMP_LNAME                          VARCHAR2(20)
```

```
EMP_FNAME                                 VARCHAR2(20)
EMP_DOB                                   DATE
EMP_TYPE                                  VARCHAR2(20)
EMP_SALARY                                NUMBER(10,2)
TEAM_NAME                                 VARCHAR2(35)
COACH_POSITION                            VARCHAR2(20)

SQL> DESCRIBE PLAYER;
PLAYER extends SYSTEM.EMP
PLAYER is NOT FINAL
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
EMP_NUM                                   CHAR(5)
EMP_LNAME                                 VARCHAR2(20)
EMP_FNAME                                 VARCHAR2(20)
EMP_DOB                                   DATE
EMP_TYPE                                  VARCHAR2(20)
EMP_SALARY                                NUMBER(10,2)
TEAM_NAME                                 VARCHAR2(35)
PLAYER_POSITION                           VARCHAR2(20)

SQL> DESCRIBE SEASON;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
SEASON_YEAR                               NOT NULL NUMBER(38)
SEASON_START_WEEK                                  DATE
SEASON_END_WEEK                                    DATE

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'SEASON';

CONSTRAINT_NAME              C
---------------------------- -
SYS_C007192                  P

SQL> DESCRIBE GAME;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
GAME_NUM                                  NOT NULL NUMBER
VISITING_TEAM                                      VARCHAR2(35)
HOME_TEAM                                          VARCHAR2(35)
SEASON_YEAR                                        NUMBER(38)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'GAME';

CONSTRAINT_NAME              C
---------------------------- -
SYS_C007193                  P
FK_GAME                      R

SQL> DESCRIBE SCHEDULE;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
TEAM_NAME                                 NOT NULL VARCHAR2(35)
GAME_NUM                                  NOT NULL NUMBER
SCHED_OPPONENT                                     VARCHAR2(35)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'SCHEDULE';

CONSTRAINT_NAME              C
---------------------------- -
PK_SCHEDULE                  P
FK_SCHEDULE1                 R
FK_SCHEDULE2                 R

SQL> DESCRIBE PERFORMANCE;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
STAT_ID                                   NOT NULL CHAR(5)
SEASON_YEAR                                        NUMBER(38)
TEAM_NAME                                          VARCHAR2(35)

SQL> SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE FROM USER_CONSTRAINTS WHERE TABLE_NAME = 'PERFORMANCE';

CONSTRAINT_NAME              C
---------------------------- -
SYS_C007198                  U
SYS_C007199                  P
FK_PERFORMANCE1              R
FK_PERFORMANCE2              R

SQL> DESCRIBE PERF;
PERF is NOT FINAL
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
STAT_ID                                   CHAR(5)
SEASON_YEAR                               NUMBER(38)
```

```
 TEAM_NAME                                 VARCHAR2(35)

SQL> DESCRIBE DEFENSE;
 DEFENSE extends SYSTEM.PERF
 DEFENSE is NOT FINAL
 Name                           Null?    Type
 ------------------------------------- -------- ----------------------------
 STAT_ID                                CHAR(5)
 SEASON_YEAR                            NUMBER(38)
 TEAM_NAME                              VARCHAR2(35)
 POINTS_ALLOWED                         NUMBER(38)
 TACKLES                                NUMBER(38)
 KNOCKDOWNS                             NUMBER(38)
 SACKS                                  NUMBER(38)
 INTERCEPTIONS                          NUMBER(38)
 FUMBLES_RECOVERED                      NUMBER(38)
 RETURN_TD                              NUMBER(38)
 PASS_YDS_ALLOWED                       NUMBER(38)
 RUSH_YDS_ALLOWED                       NUMBER(38)

SQL> DESCRIBE OFFENSE;
 OFFENSE extends SYSTEM.PERF
 OFFENSE is NOT FINAL
 Name                           Null?    Type
 ------------------------------------- -------- ----------------------------
 STAT_ID                                CHAR(5)
 SEASON_YEAR                            NUMBER(38)
 TEAM_NAME                              VARCHAR2(35)
 PASS_ATTEMPTS                          NUMBER(38)
 PASS_COMPLETIONS                       NUMBER(38)
 PASS_YDS                               NUMBER(38)
 RUSH_ATTEMPTS                          NUMBER(38)
 RUSH_YDS                               NUMBER(38)
 PASS_TD                                NUMBER(38)
 RUSH_TD                                NUMBER(38)
 SACKS_ALLOWED                          NUMBER(38)
 PASS_INTERCEPTIONS                     NUMBER(38)
 TURNOVERS                              NUMBER(38)

SQL> DESCRIBE SPECIAL_TEAMS;
 SPECIAL_TEAMS extends SYSTEM.PERF
 SPECIAL_TEAMS is NOT FINAL
 Name                           Null?    Type
 ------------------------------------- -------- ----------------------------
 STAT_ID                                CHAR(5)
 SEASON_YEAR                            NUMBER(38)
 TEAM_NAME                              VARCHAR2(35)
 POINT_AFTER_ATTEMPT                    NUMBER(38)
 POINT_AFTER_MADE                       NUMBER(38)
 FIELD_GOAL_ATTEMPT                     NUMBER(38)
 FIELD_GOAL_MADE                        NUMBER(38)
 PUNT_RETURNS                           NUMBER(38)
 PUNT_RETURN_YDS                        NUMBER(38)
 KICK_RETURNS                           NUMBER(38)
 KICK_RETURN_YDS                        NUMBER(38)
 SPECIAL_TEAMS_TD                       NUMBER(38)

SQL> SPOOL OFF;
```