American Football Special Teams Database Data Dictionary

Purpose

The purpose of this database was to have a streamlined, all in one place for Kent State Football Special Teams Data. Previously, special teams data was fragmented across a Business Intelligence (BI) application and multiple Microsoft Excel workbooks. While these sources provided the raw data, this approach made it difficult to efficiently query select data elements, generate reports, and derive meaningful insights. This relational database was developed to overcome those limitations.

The database contains comprehensive, relevant data on special teams performance, including specific metrics such as Hash Location, Punt Distance, Snap Time, and Weather conditions. To optimize performance and ensure data integrity, the database is structured in Third Normal Form (3NF), eliminating both transitive and partial dependencies.

The complete database structure is detailed in the Entity-Relationship Diagram (ERD) found on page 13. Additionally, each view present in the database is listed on pages 11-12 along with their corresponding columns.

TABLES:

The tables included in this database are:

- Conditions: Describes weather, game, practice, field conditions.
- Field_Goal: Contains information regarding every field goal attempt.
- Game: Holds each game played and its members.
- Hash: Relates to the Field Goal table and shows what each HashID represents.
- Kickoff: Contains information regarding every kickoff.
- Location: Relates to the Kickoff table representing what each locationID means. (See page 16 for diagram)
- Norm_Values_KO: Used to normalize values in the kickoff table allowing for ease of creating the ‘kickoff_view’.
- Norm_Values_Punt: Used to normalize values in the punt table allowing for ease of creating the ‘punt_view’.
- Norm_Values_Snap:Used to normalize values in the Punt table related to snapping allowing for ease of creating the ‘snap_view’.
- Player: Contains player demographic and positional information.
- Position: Related to the Player table and holds what each Position is.
- Punt: Contains information regarding every punt attempt.

- Teams: Related to the Game table, contains additional team information.

Table Columns:

Conditions

Column Name	**Datatype**	**Nullable?**	**Context**
Date (PK)	date	NN	Acts as the primary key for the conditions table. Used to be able to represent and differentiate dates on which different practices/games occurred.
Practice	tinyint(1)	N	Boolean value representing 1 for practice 0 for non-practice.
Game	int	N	Boolean value representing 1 for game 0 for non-game.
Precipitation	float(4,2)	N	Amount of Precipitation on which the event took place. Collected from weatherunderground.com.
Wind	int	N	Wind on which the event took place. Collected from weatherunderground.com.
Grass	tinyint(1)	N	Boolean value representing 1 for grass 0 for non-grass.
Turf	tinyint(1)	N	Boolean value representing 1 for turf

			0 for non-turf.
Temp	float(5,2)	N	Temperature on which the event took place. Collected from weatherunderground.com.

Field_Goal

Column Name	**Datatype**	**Nullable?**	**Context**
Result	tinyint(1)	N	Boolean value representing 1 for make and 0 for miss.
Distance	int	N	Distance in yards the kick was.
OP	float(3,2)	N	The operation time in seconds from the time the short snapper moves until the ball is kicked.
Field_GoalID (PK)	int	NN	Acts as the primary key of the field_goal table. Used to uniquely identify each field goal instance.
HashID (FK)	varchar(10)	NN	Location on the field in which the kick was attempted. Either “L, R, M, LM, RM”
PlayerIDFG (FK)	int	NN	Player Number of the kicker attempting the field goal.
PlayerIDSS (FK)	int	NN	Player Number of the short snapper.
FGDATE (FK)	date	NN	Date of which field

			goal instance took place.

Game

Column Name	**Datatype**	**Nullable?**	**Context**
GameID (PK)	int	NN	Acts as the primary key of the Game table. Used to represent each game as an auto-incrementing number.
TeamH (FK)	int	NN	Represents the Home team of each game. Represented by their Team ID.
TeamA (FK)	int	NN	Represents the Away team of each game. Represented by their Team ID.
Date (FK)	date	NN	Date when the game took place.

Hash

Column Name	**Datatype**	**Nullable?**	**Context**
HashID (PK)	varchar(10)	NN	Acts as the primary key of the Hash Table. Uniquely differs between the different hashes.
Description	varchar(20)	N	Description of what each HashID represents.

Kickoff

Column Name	Datatype	Nullable?	Context
HangTime	float(3,2)	N	Duration in seconds from when the kicker's foot hits the ball until it hits the ground and/or is fielded.
Distance	int	N	Distance the kick travels.
KOID (PK)	int	NN	Acts as the primary key of the Kickoff table. Uniquely represents each Kickoff instance.
KOLocID (FK)	int	NN	Horizontal location where kick is placed.
PlayerIDKO (FK)	int	NN	Player number of the kicker attempting the kickoff.
KODate (FK)	date	NN	Date of Kickoff.
Returned	tinyint(1)	N	Is Kick returned?
Return_Yards	int	N	How many yards is kick returned for.

Location

Column Name	Datatype	Nullable?	Context
Description	varchar(30)	N	Describes what each LocationID means.
LocationID (PK)	int	NN	Acts as Primary Key for the Location table. Represents each unique

			locationID.

Norm_Values_KO

Column Name	**Datatype**	**Nullable?**	**Context**
KOID (PK)	int	NN	Uniquely Identifies each kickoff instance
MIN_KO_DIST	float	N	Minimum kickoff distance
MAX_KO_DIST	float	N	Maximum kickoff distance
KO_DIST_NORM	float	N	Normalized kickoff distance value
MAX_KO_HANG	float	N	Maximum kickoff hang
MIN_KO_HANG	float	N	Minimum kickoff hang
KO_HANG_NORM	float	N	Normalized kickoff hang
MIN_KO_LOC	float	N	Minimum kickoff location
MAX_KO_LOC	float	N	Maximum kickoff location
KO_LOC_NORM	float	N	Normalized kickoff location

Norm_Values_Punt

Column Name	**Datatype**	**Nullable?**	**Context**
PUNTID (PK)	int	NN	Uniquely identifies each punt instance.
MIN_P_DIST	float	N	Minimum punt

			distance
MAX_P_DIST	float	N	Maximum punt distance
P_DIST_NORM	float	N	Normalized punt distance value
MIN_P_HANG	float	N	Maximum punt hang
MAX_P_HANG	float	N	Minimum punt hang
P_HANG_NORM	float	N	Normalized punt hang
MIN_P_LOC	float	N	Minimum punt location
MAX_P_LOC	float	N	Maximum punt location
P_LOC_NORM	float	N	Normalized punt location

Norm_Values_Snap

Column Name	**Datatype**	**Nullable?**	**Context**
PUNTID (PK)	int	NN	Uniquely identifies each punt instance for each snap.
MIN_SNAPTIME	float	N	Minimum snaptime
MAX_SNAPTIME	float	N	Maximum snaptime
SNAPTIME_NORM	float	N	Normalized snaptime
MIN_SNAPLOCID	float	N	Minimum snaplocID
MAX_SNAPLOCID	float	N	Maximum snaplocid
SNAPLOCID_NORM	float	N	Normalized snaplocid

Player

Column Name	Datatype	Nullable?	Context
First_Name	varchar(100)	N	Player first name.
Last_Name	varchar(100)	N	Player last name.
Position (FK)	varchar(100)	NN	Player Position.
PlayerID (PK)	int	NN	Acts as Primary Key for the player table. The player jersey number acts as a unique ID.
Height	int	N	Player height.
Weight	int	N	Player Weight

Position

Column Name	Datatype	Nullable?	Context
Position (PK)	varchar(100)	NN	Acts as the primary key for the Position Table. Each position is given its own unique ID.
Description	varchar(100)	N	Describes what each PositionID represents.

Punt

Column Name	Datatype	Nullable?	Context
Snaptime	float(3,2)	N	Time in seconds from the long snapper's initial movement to when it reaches the

			punter's hands.
OP	float(3,2)	N	Time in seconds from long snappers initial movement until it is kicked by the punter.
H2F	float(3,2)	N	Difference in seconds between OP and Snaptime.
Hang	float(3,2)	N	Duration in seconds from when the punter's foot hits the ball until it hits the ground and/or is fielded.
Distance	int	N	Distance the punt travels in yards.
PuntID (PK)	int	NN	Acts as the primary key in the punt table. Auto-incrementing number used to identify each punt instance.
PlayerIDP (FK)	int	NN	Punter Jersey Number.
PLocID (FK)	int	NN	Horizontal location where the punt lands.
PlayerIDLS (FK)	int	NN	Long Snapper Jersey Number.
SnapLocID (FK)	int	NN	Horizontal and Vertical Location where the snap goes to. See additional documentation.
PDate (FK)	date	NN	Date the punt occurred.
Returned	tinyint(1)	N	Was punt returned

Column Name	Datatype	Nullable?	Context
Return_Yards	int	N	How many yards was punt returned for.

Teams

Column Name	Datatype	Nullable?	Context
TeamID (PK)	int	NN	Acts as the primary key of the Teams table. Unique auto-incrementing number identifying each team.
TeamName	varchar(100)	N	Team name
Conference	varchar(100)	N	Conference name
Division	varchar(100)	N	Division name

VIEWS:

The views included in this database are:

- Kickoff_View: Provides important metrics and the resulting performance score
    - WeightedSum_Score = 0.50 * KOLOCID + 0.35 * Distance + 0.15 * HangTime
- Punt_View: Provides important metrics and the resulting performance score
    - WeightedSum_Score = 0.38 *PLocID + 0.35 * Distance + 0.27 * Hang
- Snap_View:
    - WeightedSum_Score = 0.65 * SnapLocID + 0.35 * Snaptime

Views Columns:

Kickoff_View

Column Name	**Datatype**	**Context**
KO_ID	int	Uniquely identifies each kickoff instance.
HangTime	float	Hangtime (in seconds) for that particular KO instance.
Distance	int	Distance (in yards) for that particular KO instance.
KOlocID	int	KOlocID for that particular KO instance.
WeightedSum_Score	Double	WeightedSum_Score for that particular KO instance using the formula above.

Punt_View

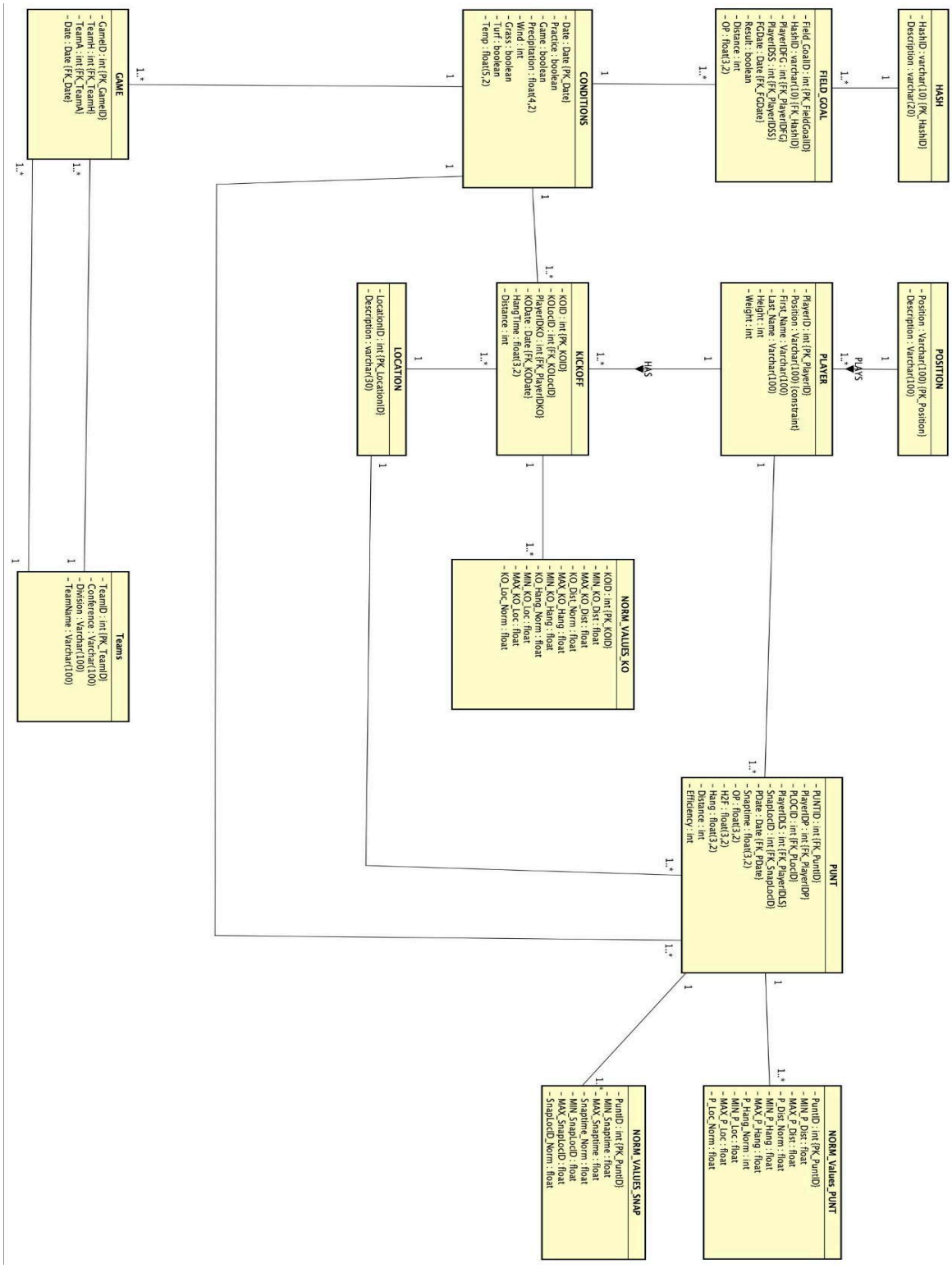
Column Name	**Datatype**	**Context**
PuntID	int	Uniquely identifies each punt instance.
Hang	float	Hangtime (in seconds) for that particular Punt instance.
Distance	int	Distance (in yards)for that particular Punt instance.
PLocID	int	PLocID for that particular Punt instance.
WeightedSum_Score	Double	WeightedSum_Score for that particular Punt instance using the formula above.

Snap_View

Column Name	**Datatype**	**Context**
PuntID	int	Uniquely identifies each punt and each punt only has one snap.
SnapLocID	int	SnapLocID of the punt instance.
Snaptime	float	Snaptime (in seconds) of the punt instance
OP	float	Operation time (in seconds) of the punt instance.
WeightedSum_Score	double	WeightedSum_Score for the snap related to the particular Punt instance using the formula above.

Class Diagram

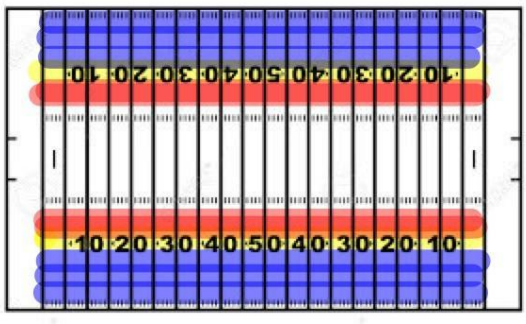
*NOTE THIS DOES NOT INCLUDE VIEWS! See pages 11-12 regarding views.



LOCATION DIAGRAMS AND CALCULATIONS:

Kickoff

- Divide the Field Up and have aiming points
    - Aiming Point Bottom Numbers
        - 3=Bottom Numbers
        - 2=On Numbers
        - 1=Between Middle Hash and Numbers
        - 0.5=Hash
        - The goal is to have the highest location score.
        - See diagram below:



Red = 1
Yellow = 2
Blue = 3

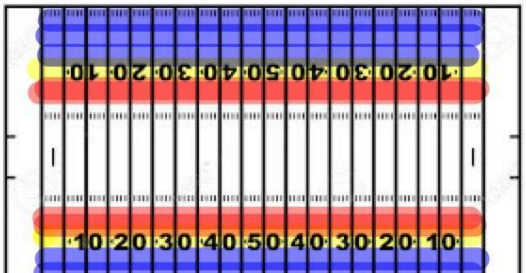
- Each value (Location, Distance and HangTime) will become normalized by using min/max normalization:

$$X_{\text{normalized}} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- WeightedSum_Score = 0.50 * KOLOCID + 0.35 * Distance + 0.15 * HangTime
    - Multiply each normalized value by the features' weight and sum.

Punt

- Divide Field Up (similar to KO)
    - Aiming Point Between Bottom Numbers & Sideline
        - 3=Between bottom numbers and sideline
        - 2=On numbers
        - 1=Between Middle Hash and Numbers
        - 0.5=Hash
        - The goal is to have the highest location score.
        - See Diagram Below



Red = 1, Yellow = 2, Blue = 3

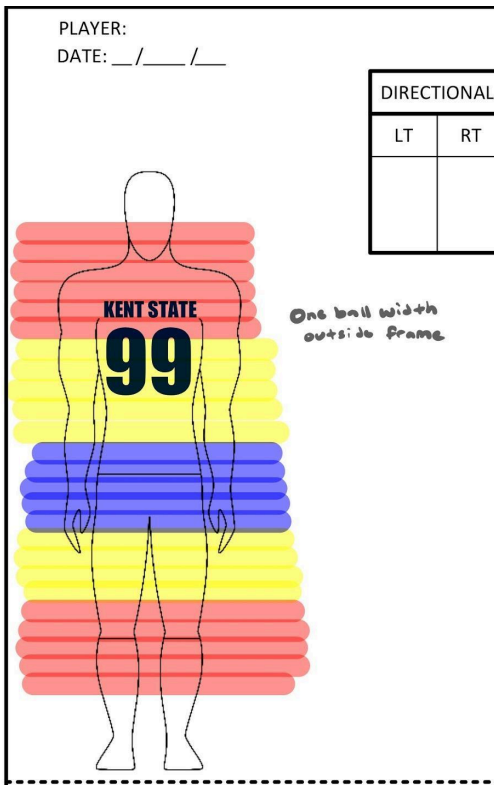
- Each value (PLocID, Distance, and Hang) will become normalized by using min/max normalization:

$$X_{\text{normalized}} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- WeightedSum_Score = 0.38 *PLocID + 0.35 * Distance + 0.27 * Hang
    - Multiply each normalized value by the features' weight and sum.

Long Snaps

- Divide the Punters Body Up
    - Aiming Point is Punter's belt line.
        - 1=Punter's Belt Line
        - 2=Punter's Belly Button-Nipples/Crotch-Middle Thigh
        - 3=Punter's Nipples-Helmet/Middle Thigh-shins
        - The goal is to have the lowest location score.
        - Multiply by Snap Time



Red = 3
Yellow = 2
Blue = 1

- Each value (SnapLocID and Snaptime) will become normalized by using min/max normalization:

$$X_{\text{normalized}} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- WeightedSum_Score = 0.65 * SnapLocID + 0.35 * Snaptime
    - Multiply each normalized value by the features' weight and sum.