SINGAPORE

SCTP Junior Data Engineer Program 2024

Final Project

NHL Game Data Analysis

Submitted by:

TEAM NAME	Nexus Nucleus
MEMBERS	Asher Pang Teck Kiat
	Chee Ser Chen
	Tan Kaeh Choy
	Wei Zhi Min

Submission Date: 7 Oct 2024

Table Of Contents

1. Background

This document outlines the process and findings from the ASMR research team, aimed at analyzing historical NHL data obtained from Kaggle. The goal is to provide actionable insights that will enhance both real-world NHL performance and gameplay dynamics for the next NHL computer game.

The project includes building an ETL pipeline on Microsoft Azure for data ingestion, transformation, as well as performing data analysis. The insights gathered will directly inform the development of a Non-Playable Character (NPC) Dream Team and strategic recommendations to improve gameplay realism and player engagement.

This document details the data architecture, the methodologies used, how the analysis was performed, the challenges encountered, and the solutions implemented.

2. Problem Statement

In late 2020, a non-disclosed game studio began developing the next NHL computer game, with the objective of creating an immersive and realistic experience for players. The studio had access to historical NHL data, which includes intricate details about individual plays such as shots, goals, stoppages, and associated x, y coordinates. However, the data remains largely unexplored due to a lack of expertise.

Recognizing its potential to provide strategic insights and improve gameplay, the studio partnered with ASMR Research to analyse the data. This collaboration aims to leverage data-driven insights to significantly enhance gameplay mechanics and player dynamics, particularly focusing on the formation of an NPC Dream Team that will serve as a formidable opponent in the game.

3. Objectives

The main objectives of the project are to:

a. **Create a comprehensive ETL pipeline** to ingest, transform and store the NHL historical data in a PostgreSQL database.
b. **Perform data analysis** to extract insights that would optimize the game play experience and to provide strategic recommendations for the business
    i. **Optimal Scoring Strategies**: To propose the best attack angles, distances, and shot types with the highest probability of scoring goals in the rink. This analysis will inform game mechanics, ensuring that NPCs can utilize strategies reflective of real-world hockey dynamics.
    ii. **NPC Dream Team Formation**: To identify and propose a dream team of 23 non-playable characters (NPCs), comprising 12 forwards, 8 defensemen, and 3 goalies. This team will be selected based on historical performance data with the aim of featuring players who have the highest potential to rank within the top 100 NHL players from 2021 to at least 2023. This objective seeks to enhance the game's competitive realism and player engagement by showcasing the best of NHL talent.

4. Scope

The scope of the project includes:

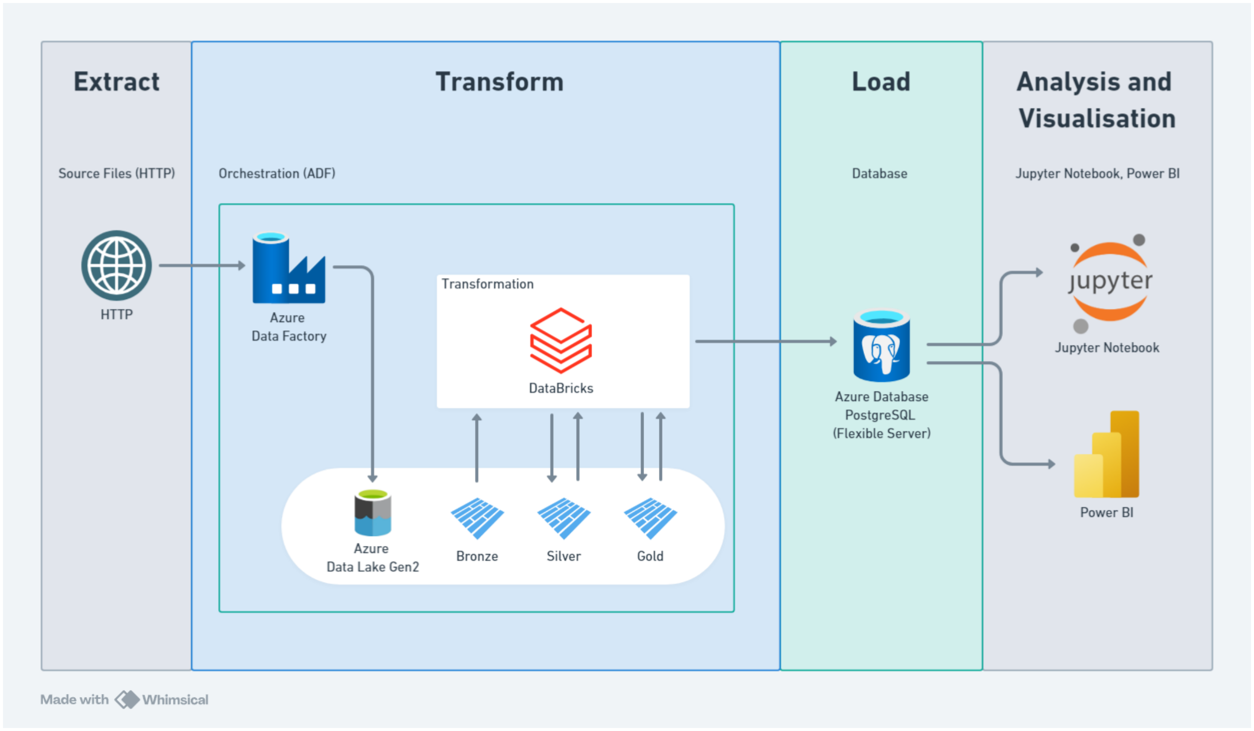
- **Timeframe of Data:** The dataset includes games from 2000 to 2020
- **Present Time:** The project's "present" is set in the year 2020.
- **Cloud Processing:** Data ingestion and processing will be handled via Microsoft Azure cloud services.
- Shots with x coordinates beyond the goal post are not considered for analysis.
- Teams identified as 87-90 are excluded, as they are not part of the standard season.
- The team's analysis will address the following use cases
    - **Offensive Strategy Identification:** Analysing shot locations, distances, and angles to determine the most effective offensive strategies for maximizing scoring opportunities.
    - **Shot Type Effectiveness:** Evaluating the most effective types of shots for various in-game situations to enhance gameplay realism and strategic depth.
    - **Performance-Based Recommendations**: Proposing NHL player-specific NPCs derived from historical performance data to optimize gameplay and player engagement.

5. Data Source

The data is sourced from Kaggle's NHL dataset: NHL Game Data. It consists of **13 tables,** of which the first 5 are used for the analysis in this project.

1. **game_plays.csv**: Records individual plays that occur during NHL games, capturing details like play ID, event type (like goal, shots, etc..), coordinates (x, y), and secondary information related to the action, for example shot types.
2. **team_info.csv**: Contains information about each NHL team, such as team name, abbreviation, and other identifying attributes.
3. **player_info.csv**: Stores data about NHL players, including their player IDs, names, positions, nationality.
4. **game_skater_stats.csv**: Holds detailed statistics for skaters (non-goalies) in each game, such as shots, assists, hits, and blocked shots.
5. **game_goalie_stats.csv**: Contains detailed statistics for goalies, including saves, goals allowed, and shots faced during each game.
6. **game_officials.csv**: Provides information about the officials (referees and linesmen) who participated in the games.
7. **game_scratches.csv**: Lists the players who were benched or "scratched" from playing in specific games.
8. **game_teams_stats.csv**: Summarizes key stats for each team per game, such as win/loss, home/away, shots, goals, penalties, and time on attack.
9. **game_shifts.csv**: Captures the ice time for individual players during each game, including start and end times of shifts.
10. **game_goals.csv**: Provides details about goals scored in each game, in which strength (even play, power play), if that goal is the winning goal and if the net was empty (no goalie)
11. **game_penalties.csv**: Documents penalties taken in every game, how severe is the penalty and the penalty duration
12. **game_plays_players.csv**: Documents the action (assist, blocker, hitter, shooter...) of each player during each play of each game
13. **game.csv**: Contains metadata about each game, such as game ID, date, home and away teams, and final scores.

6. Architecture Overview



Azure Services used:

- Azure Data Lake Gen 2 (ADLS Gen2)
- Azure Data Factory (ADF)
- Azure Databricks
- Azure Database PostgreSQL

ADLS Gen2 - Scalable Data Storage Solution

ADLS Gen2 is a highly scalable and cost-effective storage solution designed for big data analytics. It provides both file system semantics and the scale of object storage, enabling efficient management of vast amounts of data.

In this project, the team implemented a Medallion Architecture on our ADLS Gen2 storage account, consisting of three distinct layers: Bronze, Silver and Gold. This architecture facilitates incremental transformations of the NHL data and helps to organize the data logically. It also provides data lineage as our data flows through each layer:

- **Bronze Layer:** The Bronze layer serves as the repository for raw, unprocessed data. The data within this layer mirrors the original source system without any modification, and no schema is enforced. The primary objectives of the Bronze layer include ensuring historical archiving, maintaining data lineage, enabling auditability, and allowing data to be reprocessed without necessitating a reread from the source system. In our project, the bronze layer is designated to store raw ZIP file ingested from the Kaggle website, as well as the outputs of these unzipped files.

- **Silver Layer:** The Silver layer represents the cleansed and conformed stage of our data architecture. Data from the Bronze layer undergoes cleansing and transformation processes, which include deduplication, handling of null values, and enforcement of data types. While our project currently involves a single data source, this layer is also designed so that data from multiple sources can be integrated into a unified view, which will benefit future use cases that require data ingestion from other sources.
- **Gold Layer:** The Gold layer consists of highly curated, consumption-ready data organized into project-specific use cases. This layer is designed for reporting and business intelligence purposes, facilitating the final presentation of analytical projects. Data from the silver undergoes transformation at varying levels of complexity, tailored to meet specific project or specific requirements.

Delta format file type: In both Silver and Gold layers, data is stored in Delta format, which is based on Apache Parquet. The Delta format is specifically designed for high-volume data processing and analytics, and it has been selected over parquet and csv formats due to several advantages:

- **Caching:** Delta format includes a Delta Cache that enhances performance for repeated queries while ensuring data consistency amid changes.
- **Versioned Data:** With transaction logs, Delta allows users to query data as it existed at a specific point in time, facilitating data restoration and audit trails as necessary.
- **ACID Transactions:** Delta format supports ACID (Atomicity, Consistency, Isolation, Durability) transactions, ensuring reliable data operations. This capability allows multiple concurrent writes without data corruption, representing a significant improvement over Parquet.
- **Unified Batch and Streaming:** Delta can efficiently manage both batch and streaming data, providing a cohesive approach to data ingestion and processing, thereby simplifying workflows and minimizing the need for separate processing pipelines.
- **Schema Enforcement:** Delta enforces a defined data structure, addressing the limitations of CSV files in this area.

Azure Data Factory (ADF) – Automated Data Integration

Azure Data Factory is a cloud-based ETL service that automates data movement and transformation. It allows for seamless orchestration of complex workflows with trigger-based scheduling. In our project, ADF is used to ingest raw files into ADLS Gen2. It automates the file extraction, transformation and loading tasks that move data between different stages (from raw files in the bronze layer to the cleaned data in the silver and gold layers). Further regarding the implementation of ADF can be found in the Methodology section.

Azure Databricks – Data Transformation

Azure Databricks is a jointly developed data and AI service from Databricks and Microsoft. It provides a unified interface to execute a variety of data tasks. In the context of our project, Databricks notebooks were utilized to execute data transformations across the Bronze, Silver layers in ADLS Gen2, as well as loading data from gold layers to a database. Further details regarding the implementation of Azure Databricks can be found in the Methodology section.

Azure Database for PostgreSQL – Processed Data Storage

Azure Database PostgreSQL is a managed relational database service that provides a scalable, reliable, and secure environment for running PostgreSQL-based applications. For our project, the data from the Gold layer is loaded into this database, making it ready for data analyst and business users to query using SQL, or to access through other tools including Jupyter notebooks and Power BI.

7. Methodology

- **Data Ingestion**: Data is ingested from Kaggle website using Azure Data Factory, followed by unzipping and loading into ADLS Gen2.
- **Data Transformation and Loading**: Databricks notebooks were used to clean and transform data, as well as to load data into a database for further analysis.
- **Analysis & Visualization**: Power BI and Jupyter Notebooks are used to explore player performance and generate actionable insights.

Data Ingestion using Azure Data Factory

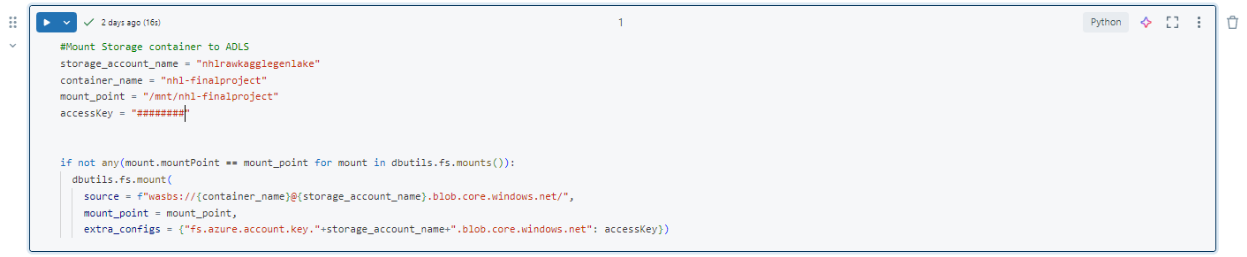
Data ingestion is achieved through two "Copy Data" activities within Azure Data Factory (ADF). The first activity establishes a connection to the Kaggle website to download the source files as ZIP archives, which are then stored in the Bronze layer. Upon the successful completion of this task, the second activity accesses the Bronze layer to unzip the downloaded file, subsequently loading all CSV files into the same layer.

Data Transformation and Loading using Databricks Notebooks

Data transformation is conducted through Databricks notebooks utilizing Python. The team has explored Pandas and PySpark for executing these notebooks. PySpark is specifically designed for distributed computing and big data processing, making it ideal for handling large-scale datasets. In contrast, Pandas is tailored for moderately sized datasets that can be accommodated in memory.

The project data contains a dataset “game_plays.csv” which comprises several million rows. As this was a substantial volume of data, PySpark was employed to handle the size by leveraging on its distributed computing and parallelized data processing capabilities. Conversely, for the other smaller to moderately sized datasets, using PySpark may lead to inefficiencies in resource allocation and consumption; thus, Pandas was utilized for them instead.

Before initiating the transformation, access to ADLS Gen2 was configured by mounting it to the Databricks File System. This setup allows Databricks to read the latest files and write output files to the relevant ADLS container.



```python
#Mount Storage container to ADLS
storage_account_name = "nhlrawkagglegenlake"
container_name = "nhl-finalproject"
mount_point = "/mnt/nhl-finalproject"
accessKey = "########"


if not any(mount.mountPoint == mount_point for mount in dbutils.fs.mounts()):
  dbutils.fs.mount(
    source = f"wasbs://{container_name}@{storage_account_name}.blob.core.windows.net/",
    mount_point = mount_point,
    extra_configs = {"fs.azure.account.key."+storage_account_name+".blob.core.windows.net": accessKey})
```

Transformation stage follows a two-step approach: transitioning from Bronze to Silver, and subsequently from Silver to Gold. During the Bronze to Silver transformation, the following steps are taken to ensure data is cleansed and conformed:

- Dropping duplicate entries
- Renaming columns for consistency
- Changing column data types to appropriate formats
- Handling null values effectively
- Writing the cleansed data into the Silver layer

Once the data is cleansed, further project-specific transformations from the Silver to Gold stage can be performed, where the data is tailored to the requirements of two analyses (details here) performed by the team: X-Y Coordinates Analysis and NPC Dream Team Analysis

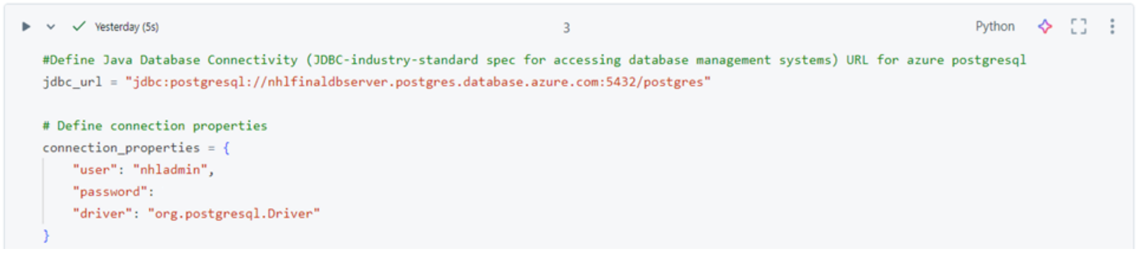
For the **X-Y Coordinate Analysis**, all x and y coordinates are transposed to the right side of the rink for more accessible computation generating new columns for these coordinates. The required columns are extracted, and additional columns are introduced.

- Transposed x (“x1”): Absolute value of x
- Transposed y (“y1”): If x is positive, transposed y remains unchanged. If x is negative, transposed y is “flipped” by multiplying by -1
- Goal post position (“gp_x”): Set to 89
- Removal of shots behind goal posts: Filtering out x coordinates that are less than -89 and more than 89.

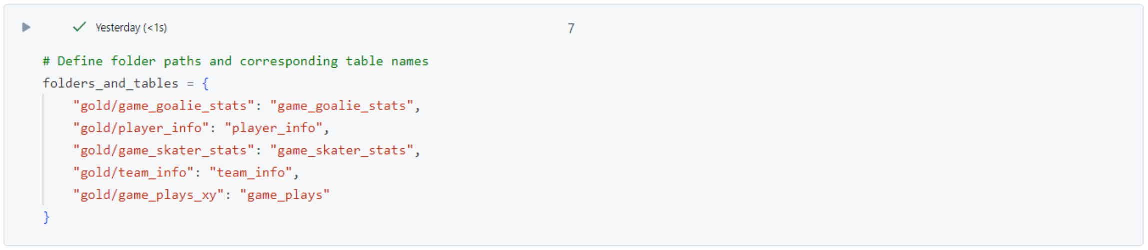
For the **NPC Dream Team Analysis**, data undergoes the following transformations:

- An “age” column is created, based on birthdate.
- A “weight_kg” column is created, based on the weight (in pounds) column.
- Rows containing values of 87, 88, 89, and 90 in the “team_id” column are removed.
- All unnecessary columns are removed.

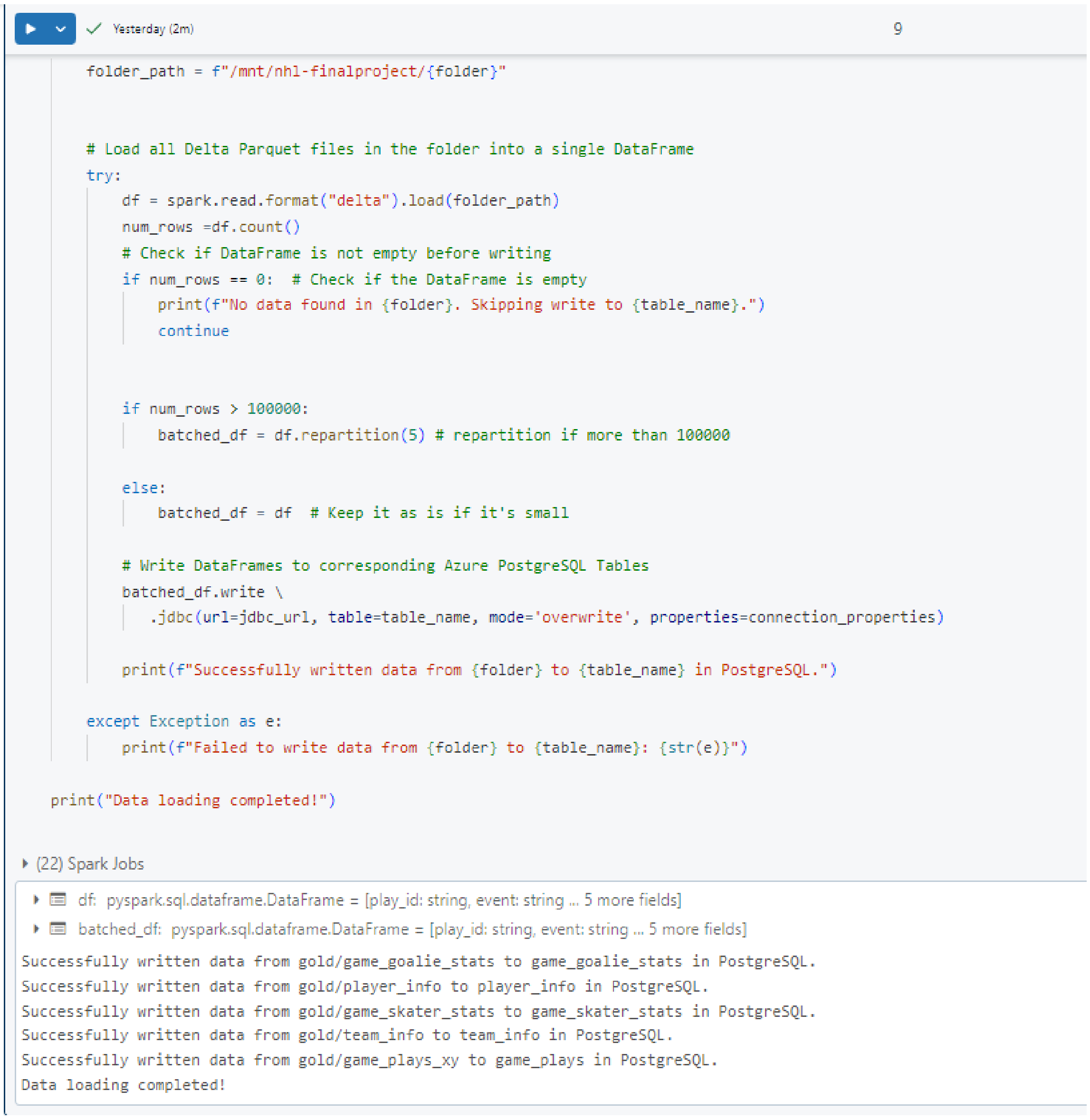
Data Loading is performed on the gold layer by uploaded the data in the Gold layer to an Azure PostgreSQL database using Databricks notebook. The notebook connects to the database by specifying a Java Database Connectivity (JDBC) URL as well as the relevant connection properties such as the user id and password. Using a dictionary of folders and tables, the notebook converts each of the files into a spark dataframe and loads it to the database.



```python
#Define Java Database Connectivity (JDBC-industry-standard spec for accessing database management systems) URL for azure postgresql
jdbc_url = "jdbc:postgresql://nhlfinaldbserver.postgres.database.azure.com:5432/postgres"

# Define connection properties
connection_properties = {
    "user": "nhladmin",
    "password":
    "driver": "org.postgresql.Driver"
}
```



```python
# Define folder paths and corresponding table names
folders_and_tables = {
    "gold/game_goalie_stats": "game_goalie_stats",
    "gold/player_info": "player_info",
    "gold/game_skater_stats": "game_skater_stats",
    "gold/team_info": "team_info",
    "gold/game_plays_xy": "game_plays"
}
```



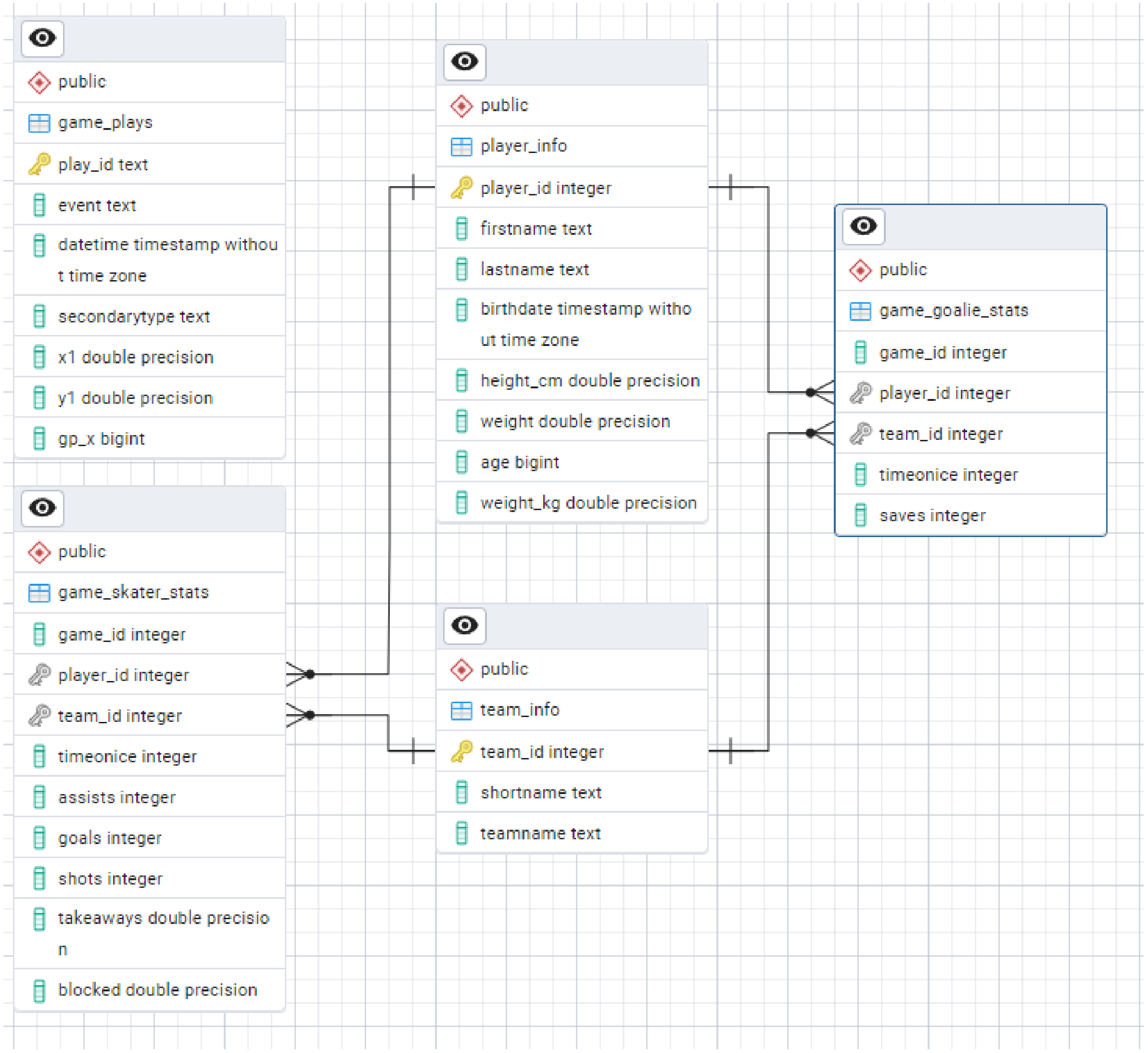
```python
    folder_path = f"/mnt/nhl-finalproject/{folder}"


    # Load all Delta Parquet files in the folder into a single DataFrame
    try:
        df = spark.read.format("delta").load(folder_path)
        num_rows =df.count()
        # Check if DataFrame is not empty before writing
        if num_rows == 0:  # Check if the DataFrame is empty
            print(f"No data found in {folder}. Skipping write to {table_name}.")
            continue


        if num_rows > 100000:
            batched_df = df.repartition(5) # repartition if more than 100000

        else:
            batched_df = df  # Keep it as is if it's small

        # Write DataFrames to corresponding Azure PostgreSQL Tables
        batched_df.write \
           .jdbc(url=jdbc_url, table=table_name, mode='overwrite', properties=connection_properties)

        print(f"Successfully written data from {folder} to {table_name} in PostgreSQL.")

    except Exception as e:
        print(f"Failed to write data from {folder} to {table_name}: {str(e)}")

print("Data loading completed!")
```

(22) Spark Jobs

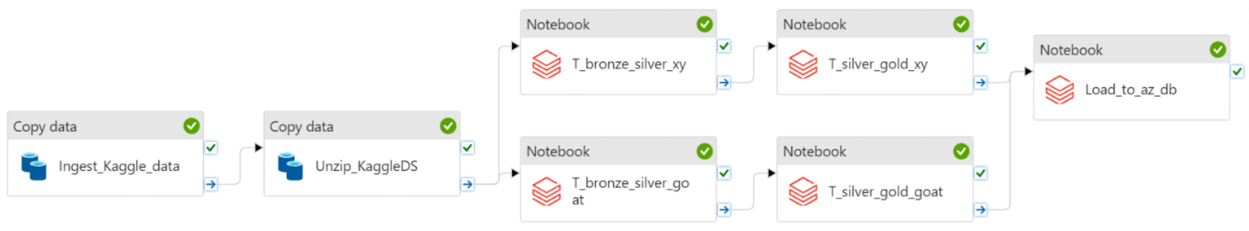
df: pyspark.sql.dataframe.DataFrame = [play_id: string, event: string ... 5 more fields]
batched_df: pyspark.sql.dataframe.DataFrame = [play_id: string, event: string ... 5 more fields]

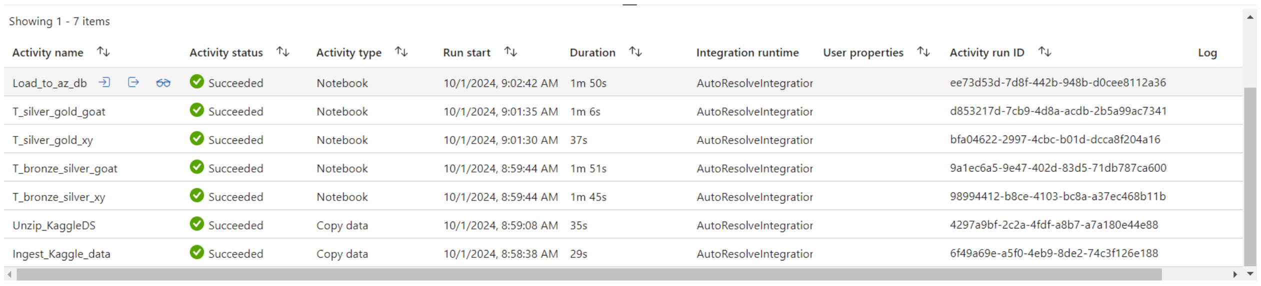
```
Successfully written data from gold/game_goalie_stats to game_goalie_stats in PostgreSQL.
Successfully written data from gold/player_info to player_info in PostgreSQL.
Successfully written data from gold/game_skater_stats to game_skater_stats in PostgreSQL.
Successfully written data from gold/team_info to team_info in PostgreSQL.
Successfully written data from gold/game_plays_xy to game_plays in PostgreSQL.
Data loading completed!
```

The final ERD for the database is as follows:



Data Pipeline Orchestration using Azure Data Factory





Showing 1 - 7 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID	Log
Load_to_az_db	✓ Succeeded	Notebook	10/1/2024, 9:02:42 AM	1m 50s	AutoResolveIntegratior		ee73d53d-7d8f-442b-948b-d0cee8112a36	
T_silver_gold_goat	✓ Succeeded	Notebook	10/1/2024, 9:01:35 AM	1m 6s	AutoResolveIntegratior		d853217d-7cb9-4d8a-acdb-2b5a99ac7341	
T_silver_gold_xy	✓ Succeeded	Notebook	10/1/2024, 9:01:30 AM	37s	AutoResolveIntegratior		bfa04622-2997-4cbc-b01d-dcca8f204a16	
T_bronze_silver_goat	✓ Succeeded	Notebook	10/1/2024, 8:59:44 AM	1m 51s	AutoResolveIntegratior		9a1ec6a5-9e47-402d-83d5-71db787ca600	
T_bronze_silver_xy	✓ Succeeded	Notebook	10/1/2024, 8:59:44 AM	1m 45s	AutoResolveIntegratior		98994412-b8ce-4103-bc8a-a37ec468b11b	
Unzip_KaggleDS	✓ Succeeded	Copy data	10/1/2024, 8:59:08 AM	35s	AutoResolveIntegratior		4297a9bf-2c2a-4fdf-a8b7-a7a180e44e88	
Ingest_Kaggle_data	✓ Succeeded	Copy data	10/1/2024, 8:58:38 AM	29s	AutoResolveIntegratior		6f49a69e-a5f0-4eb9-8de2-74c3f126e188	

Automation of the data pipeline is achieved using Azure Data Factory (ADF) to orchestrate tasks. Key steps in the ETL process are broken down into individual activities, with each activity executing automatically once the preceding task is completed, following the flow sequence depicted in the diagram above.

- **Extract**: Two "Copy Data" activities were configured to ingest the data. The source and sink locations were defined to point to the appropriate data sources and destination locations.
- **Transform and Load**: Notebook activities were created, specifying the relevant paths to Databricks notebooks for each transformation and load task.

Azure Data Factory enables the entire process to be triggered with a single action, providing the capability to monitor execution in real time. It tracks whether each activity has succeeded or failed and offers detailed insights into each run. This allows for easy identification of failures and faster resolution, ensuring an efficient pipeline workflow.

Analysis & Visualization using Power BI and Jupyter Notebook:

Once the data was loaded into PostgreSQL, Power BI and Jupyter Notebooks are used to explore and analyse the data.

8. Introduction to NHL

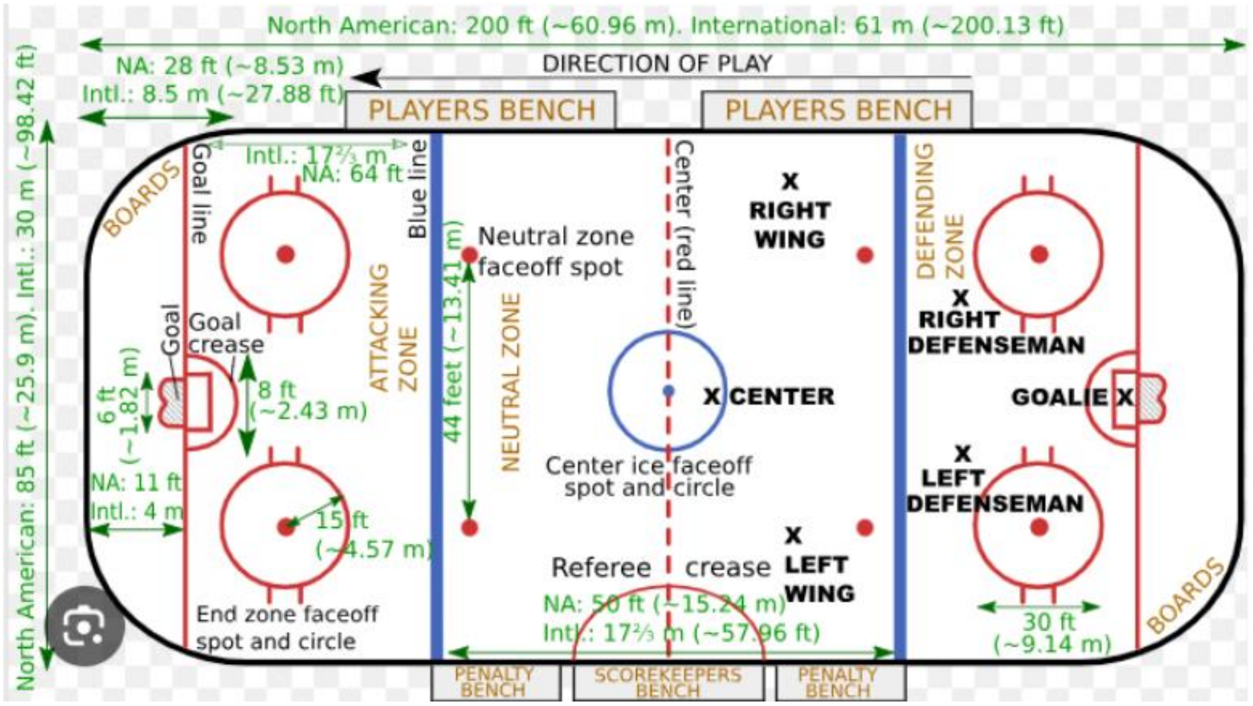
This section provides a brief introduction to the National Hockey League (NHL) for readers who are unfamiliar with the game. It also serves to outline the groundwork undertaken by the team to familiarize themselves with NHL gameplay and offers context for the analysis conducted later in the project. The foundational knowledge covered here is important in aiding understanding of the later sections on analysis.

The National Hockey League (NHL) is a professional ice hockey league in North America, comprising 32 teams from the United States and Canada. A season is from October to April. There are a total of 1312 regular games per season.

A standard game is 60 minutes long, divided into 3 periods of 20 minutes each. Each team fields 6 players consisting of 3 offensive players, 2 defensive players and 1 goalkeeper. A full team comprises of 23 players (4 sets of offenders, 4 sets of defenders and 3 goalies). Players can be substituted on the fly.

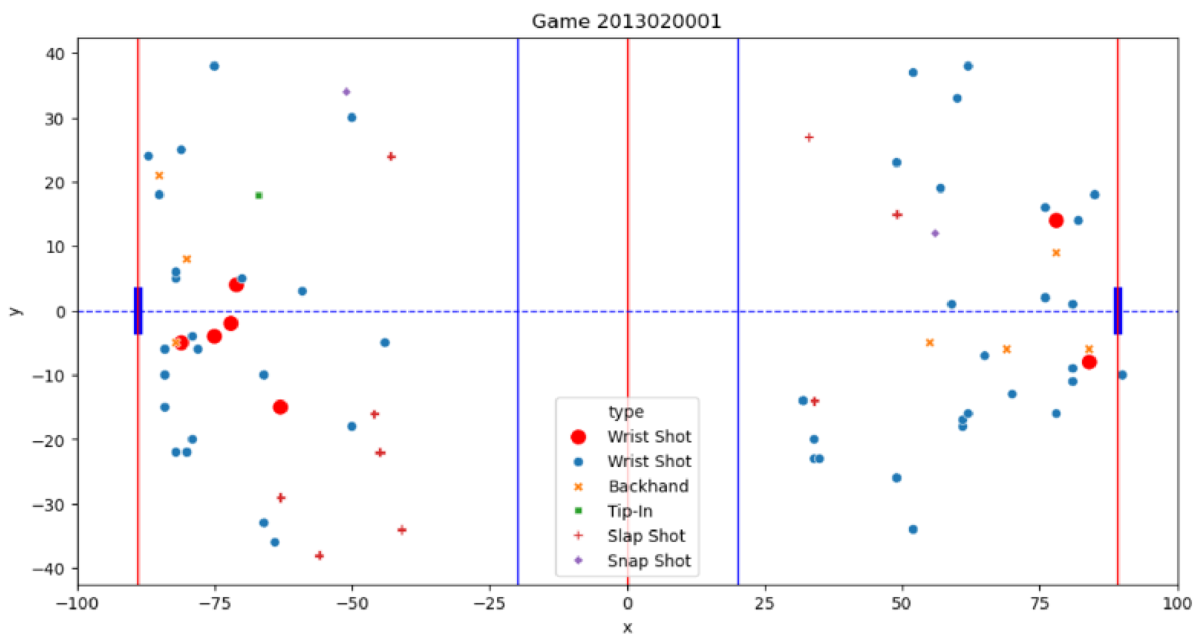
Ice Hockey Rink Dimensions and Gameplay Zones

An ice hockey rink measures 200 feet in length (x-axis) and 85 feet in width (y-axis). It is divided into two halves, with the centre of play located at the origin (0, 0) on the X-Y coordinate plane. The rink's layout is critical for understanding gameplay dynamics and shot analysis. The goal posts are positioned at x = 89 feet and x = -89 feet, representing the offensive and defensive ends of the rink. The opponent’s offensive zone spans from the blue line to the goal line in the defender's half, which also serves as the defender’s defensive zone.



Goals and Shots Overview

The accompanying diagram illustrates a plot of goals and shots from game 2013020001 of the 2013 NHL season. In this game, seven goals were scored, with five occurring on the left side of the rink and two on the right. The shot data includes various coordinates representing different types of shots, highlighting the variety in shot strategies used by players throughout the game.



Types of Shots in Ice Hockey

The analysis of shot data in ice hockey requires an understanding of the various shot types commonly used during gameplay. Each shot type has distinct characteristics that influence its effectiveness based on the game context, player positioning, and shot execution. Below are more details on the six common shot types observed:

1. Wrist Shot
   - Characteristics: Quick release, accurate, low-to-medium speed.
   - Description: A shot executed by flicking the wrist to release the puck. Wrist shots are known for their precision and are often used when quick and controlled shots are needed.
2. Snap Shot
   - Characteristics: Quick release, accurate, medium-to-high speed
   - Description: A shot characterized by a snapping motion of the stick, allowing for a rapid release with moderate accuracy. Snap shots are typically used to surprise goalies with their quick execution.
3. Slap Shot
   - Characteristics: Powerful, less accurate, high speed

- Description: A powerful shot released with a sweeping motion of the stick. Slap shots are designed to generate high speeds but often trade off accuracy for power.

4. Backhand Shot
   - Characteristics: Quick release, accurate, medium speed.
   - Description: A shot executed with the back of the stick, usually when players need to quickly release the puck under pressure. Backhand shots can be more difficult for goalies to predict due to the angle and stick position.
5. Tip Shot
   - Characteristics: Redirects a teammate's shot or pass.
   - Description: A specialized shot where a player deflects or redirects a shot or pass from a teammate, often executed near the goal to catch the goalie off guard.
6. Wrap Around
   - Characteristics: Quick release, close range, medium speed.
   - Description: A close-range shot where the player moves around the back of the net to shoot from the opposite side, typically used when the goalie is out of position.

9. Analysis

9.1: X-Y Coordinates Analysis

Problem Statement

In performing this analysis, the team seeks to address the following problem statements:

- What are the most likely positions to score a goal?
- What type of shots to be used at what distance and angle from the goal post?

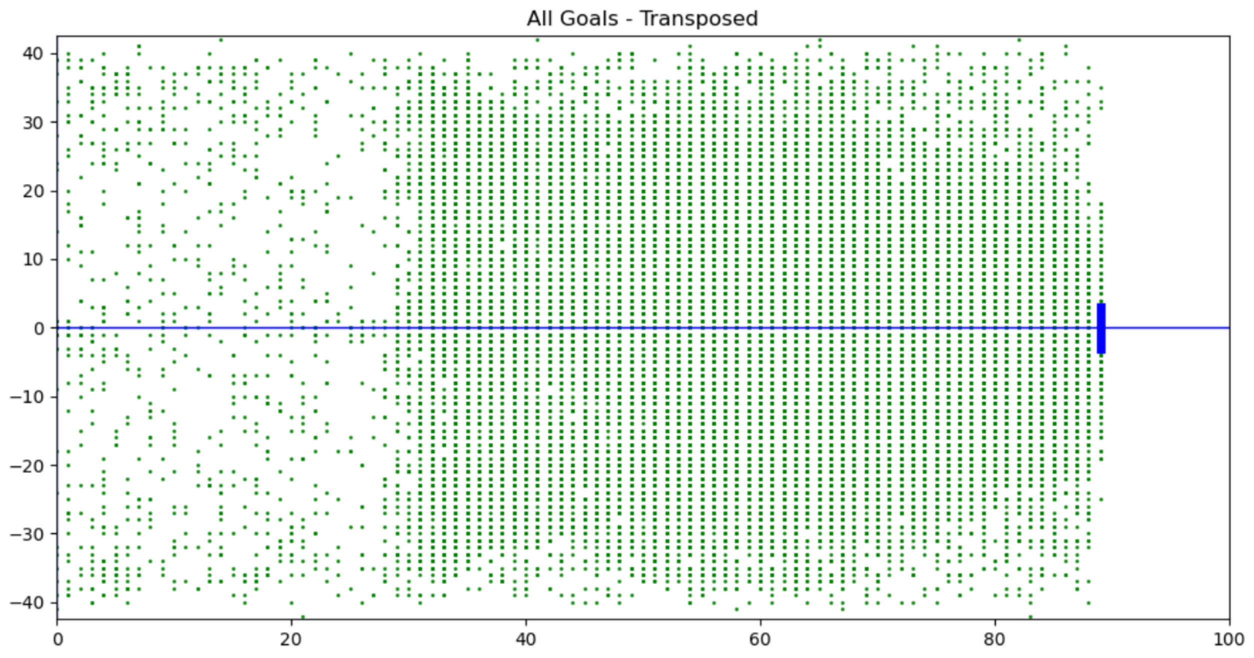
To answer this, we will analyse NHL data from 2011 to 2019 seasons in game_plays table to investigate.

Transform and Transpose table

The first (positive x and positive y) and third quadrants are the Left Wings and the second and fourth quadrants are the Right Wings respectively to the opponent's goalpost. The left side of the rink can be transposed to the right side with the appropriate flipping of the Wings.

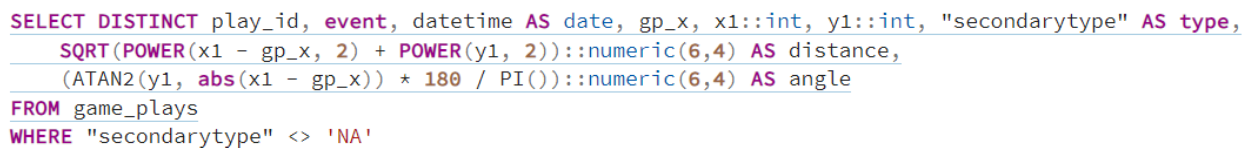
Due to the icing rule which penalizes a player from shooting the puck from its own half into the opponent's goal line without scoring a goal, the offside rule that penalizes a player from waiting at the opponent's defensive zone and the distance and accuracy of a long shot, we assume all shots and goals happen on the offensive side of the rink in this project.

After transforming our data (transform details under methodolgy), we obtain the following shot at goal plot:

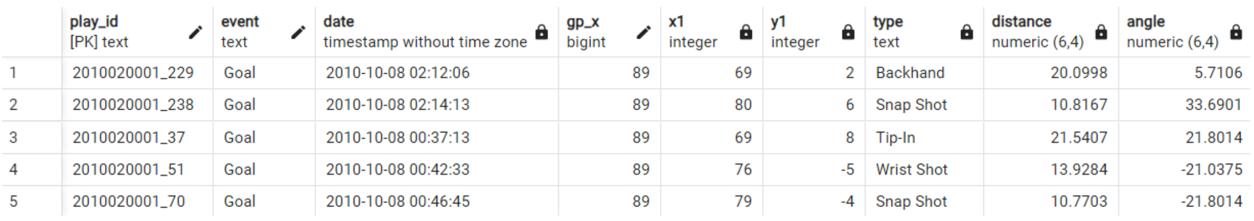


Create Distance and Angle

Queries from the transposed table create distance (using Pythagoras theorem) and angle (using trigonometry) columns. Types of shots are taken from non-NULL "secondaryType" columns.



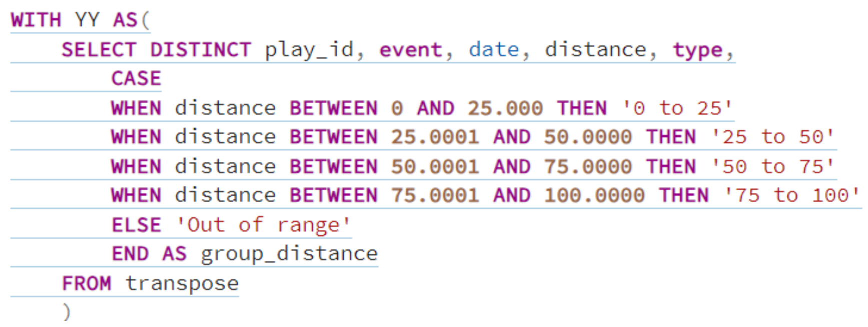
```
SELECT DISTINCT play_id, event, datetime AS date, gp_x, x1::int, y1::int, "secondarytype" AS type,
    SQRT(POWER(x1 - gp_x, 2) + POWER(y1, 2))::numeric(6,4) AS distance,
    (ATAN2(y1, abs(x1 - gp_x)) * 180 / PI())::numeric(6,4) AS angle
FROM game_plays
WHERE "secondarytype" <> 'NA'
```



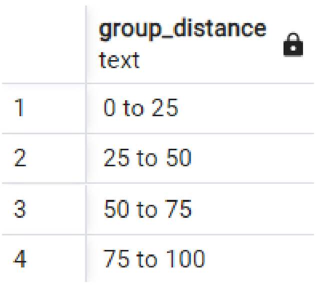
	play_id [PK] text	event text	date timestamp without time zone	gp_x bigint	x1 integer	y1 integer	type text	distance numeric (6,4)	angle numeric (6,4)
1	2010020001_229	Goal	2010-10-08 02:12:06	89	69	2	Backhand	20.0998	5.7106
2	2010020001_238	Goal	2010-10-08 02:14:13	89	80	6	Snap Shot	10.8167	33.6901
3	2010020001_37	Goal	2010-10-08 00:37:13	89	69	8	Tip-In	21.5407	21.8014
4	2010020001_51	Goal	2010-10-08 00:42:33	89	76	-5	Wrist Shot	13.9284	-21.0375
5	2010020001_70	Goal	2010-10-08 00:46:45	89	79	-4	Snap Shot	10.7703	-21.8014

Create Group_Distance and Group_Angle

To help visualization, we group distance and angle into discrete groups called group_distance and group_angle and into 2 different tables.



```
WITH YY AS(
    SELECT DISTINCT play_id, event, date, distance, type,
        CASE
        WHEN distance BETWEEN 0 AND 25.000 THEN '0 to 25'
        WHEN distance BETWEEN 25.0001 AND 50.0000 THEN '25 to 50'
        WHEN distance BETWEEN 50.0001 AND 75.0000 THEN '50 to 75'
        WHEN distance BETWEEN 75.0001 AND 100.0000 THEN '75 to 100'
        ELSE 'Out of range'
        END AS group_distance
    FROM transpose
    )
```



	group_distance text
1	0 to 25
2	25 to 50
3	50 to 75
4	75 to 100

```
WITH YY AS(
    SELECT DISTINCT play_id, event, date, angle, type,
        CASE
        WHEN angle BETWEEN -90 AND -70.0001 THEN '-90 to -70'
        WHEN angle BETWEEN -70.0000 AND -50.0001 THEN '-70 to -50'
        WHEN angle BETWEEN -50.0000 AND -30.0001 THEN '-50 to -30'
        WHEN angle BETWEEN -30.0000 AND -10.0001 THEN '-30 to -10'
        WHEN angle BETWEEN -10.0000 AND 10.0000 THEN '-10 to 10'
        WHEN angle BETWEEN 10.0001 AND 30.0000 THEN '10 to 30'
        WHEN angle BETWEEN 30.0001 AND 50.0000 THEN '30 to 50'
        WHEN angle BETWEEN 50.0001 AND 70.0000 THEN '50 to 70'
        WHEN angle BETWEEN 70.0001 AND 90 THEN '70 to 90'
        ELSE 'Out of range'
        END AS group_angle
    FROM transpose
    )
```

	group_angle text
1	-90 to -70
2	-70 to -50
3	-50 to -30
4	-30 to -10
5	-10 to 10
6	10 to 30
7	30 to 50
8	50 to 70
9	70 to 90

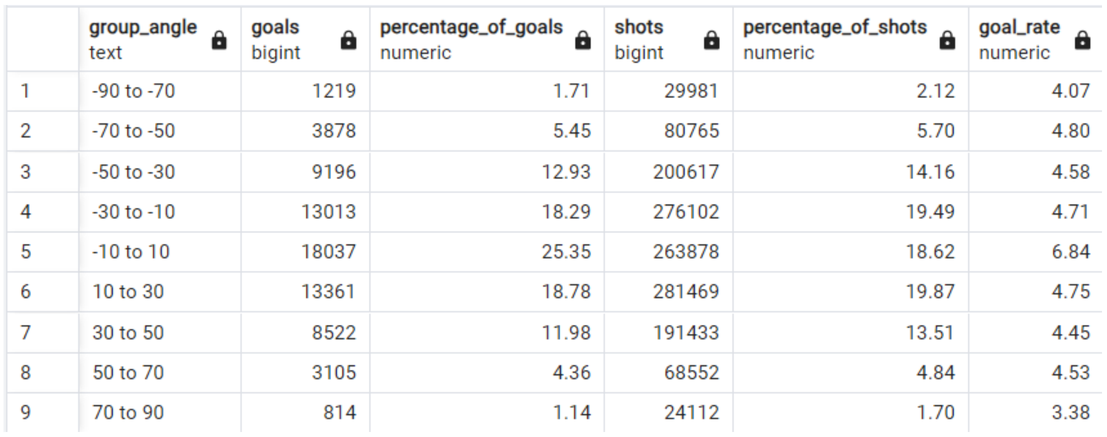
Analysis and Visualization

Goals and Saves distribution based on distance from goal post and angle of shot

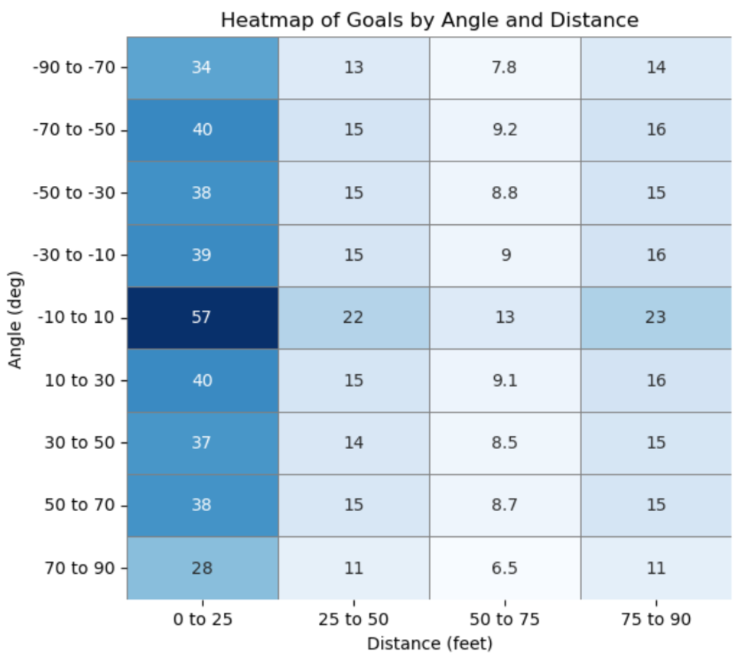
The index (goal_rate) is the product of percentage of goal (distribution of goals) and the goal percentage over shots for each distance group and angle group. Final index for heatmap is the product of angle_index and distance_index.

- **Goals:** The distribution of goals shows higher chance of goals from close ranges and narrow angles over all types of shots

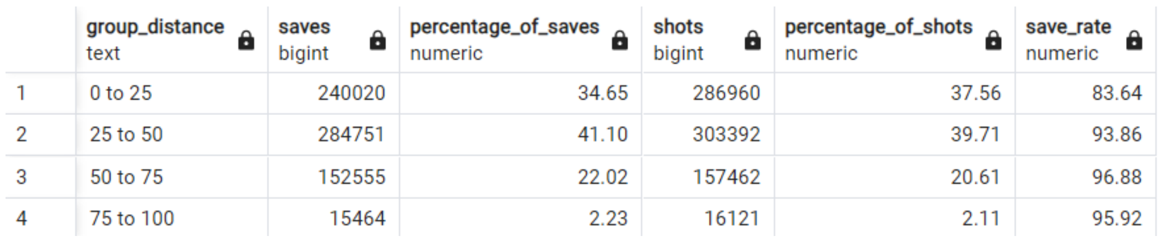
	group_distance text	goals bigint	percentage_of_goals numeric	shots bigint	percentage_of_shots numeric	goal_rate numeric
1	0 to 25	46940	65.98	562482	39.70	8.35
2	25 to 50	18641	26.20	579580	40.90	3.22
3	50 to 75	4907	6.90	255321	18.02	1.92
4	75 to 100	657	0.92	19526	1.38	3.36



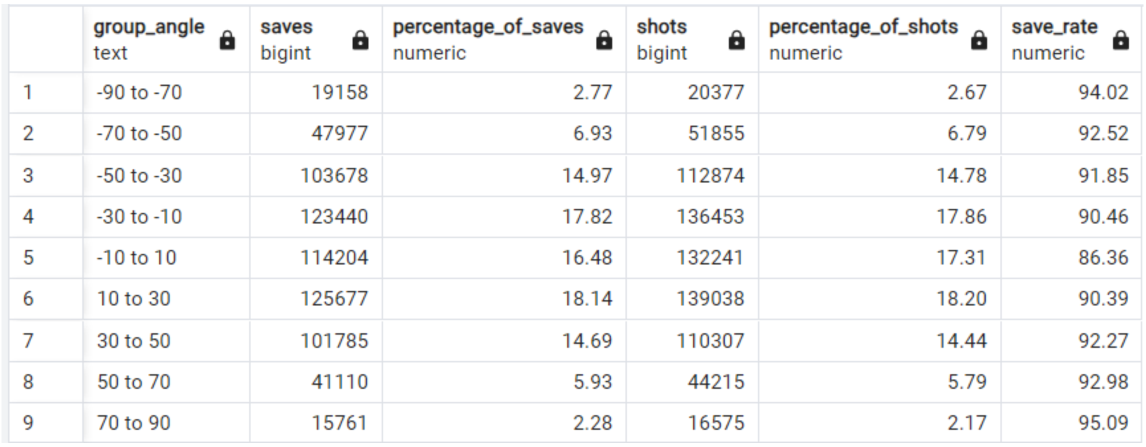
	group_angle text	goals bigint	percentage_of_goals numeric	shots bigint	percentage_of_shots numeric	goal_rate numeric
1	-90 to -70	1219	1.71	29981	2.12	4.07
2	-70 to -50	3878	5.45	80765	5.70	4.80
3	-50 to -30	9196	12.93	200617	14.16	4.58
4	-30 to -10	13013	18.29	276102	19.49	4.71
5	-10 to 10	18037	25.35	263878	18.62	6.84
6	10 to 30	13361	18.78	281469	19.87	4.75
7	30 to 50	8522	11.98	191433	13.51	4.45
8	50 to 70	3105	4.36	68552	4.84	4.53
9	70 to 90	814	1.14	24112	1.70	3.38



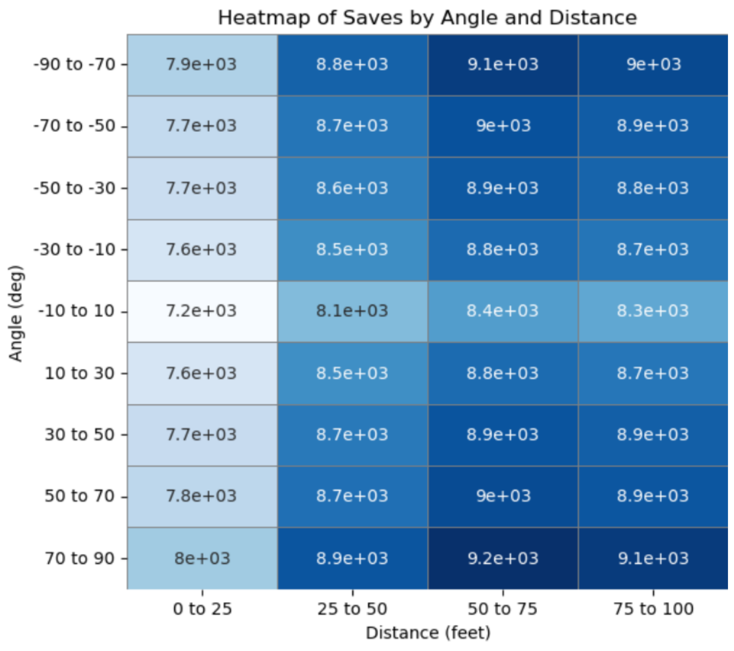
- **Saves:** Using the same model, we visualize saves instead of goals. The distribution shows consistency in goal savings. Shots coming from far and wide angles have higher chance of getting saved compared to shots coming from shorter distances and narrower angles.



	group_distance text	saves bigint	percentage_of_saves numeric	shots bigint	percentage_of_shots numeric	save_rate numeric
1	0 to 25	240020	34.65	286960	37.56	83.64
2	25 to 50	284751	41.10	303392	39.71	93.86
3	50 to 75	152555	22.02	157462	20.61	96.88
4	75 to 100	15464	2.23	16121	2.11	95.92

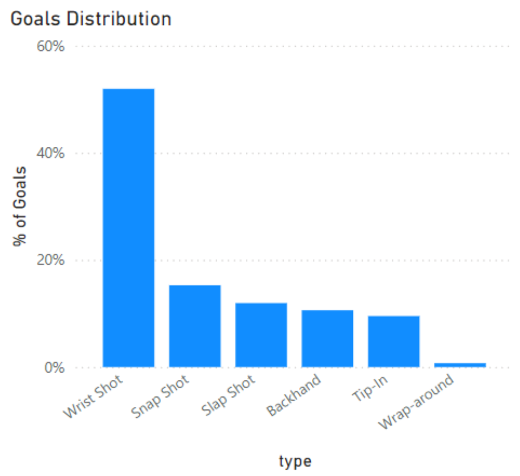


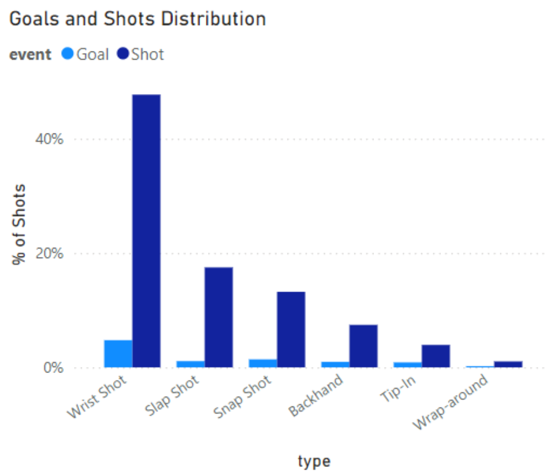
	group_angle text	saves bigint	percentage_of_saves numeric	shots bigint	percentage_of_shots numeric	save_rate numeric
1	-90 to -70	19158	2.77	20377	2.67	94.02
2	-70 to -50	47977	6.93	51855	6.79	92.52
3	-50 to -30	103678	14.97	112874	14.78	91.85
4	-30 to -10	123440	17.82	136453	17.86	90.46
5	-10 to 10	114204	16.48	132241	17.31	86.36
6	10 to 30	125677	18.14	139038	18.20	90.39
7	30 to 50	101785	14.69	110307	14.44	92.27
8	50 to 70	41110	5.93	44215	5.79	92.98
9	70 to 90	15761	2.28	16575	2.17	95.09



Distribution of Goals and Shots by Shot Type

Wrist Shot is the most common shot, followed by Snap and Slap Shots. Backhand and Tip-in are less common, while Wrap Arounds are used least.

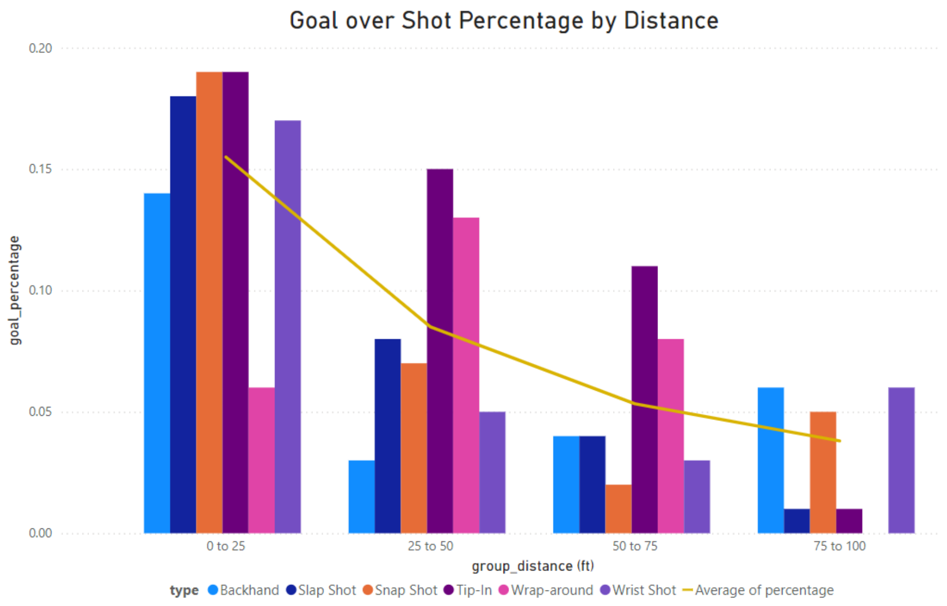




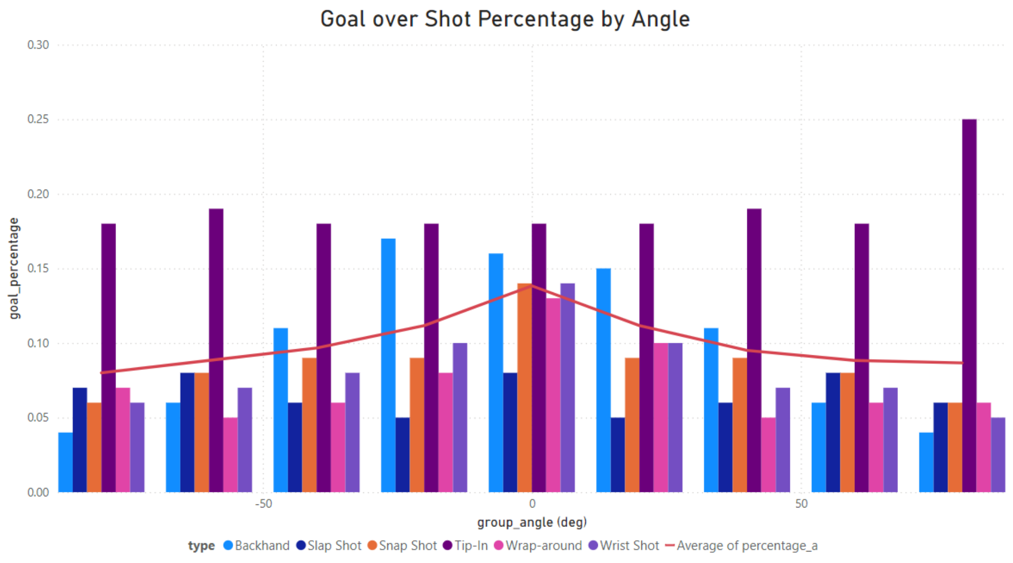
Goal Percentage over Shots

We analyse the percentage of goal for each type of shot, first by distance and then by angle.

- **By Distance:** We find that there is a higher probability of Goals at close range and the probability reduces by distance. Tip-In is highly effective converting shots to goal at different distances.



- **By Angle:** The probability of goals at different angles varies lesser than distance, with goals probability higher around the centre. Tip-In is again an effective goal converting shot at various angles.

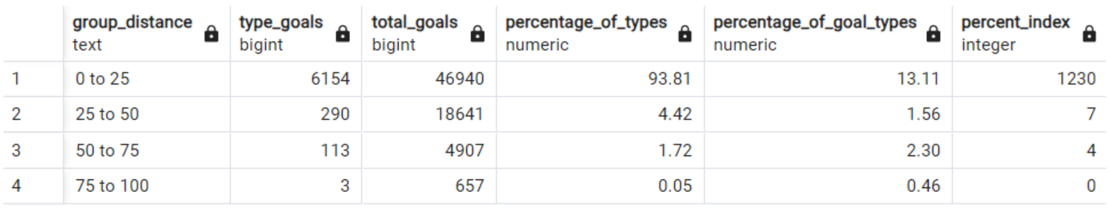


Goals distribution based on the type of shot over distance and angle

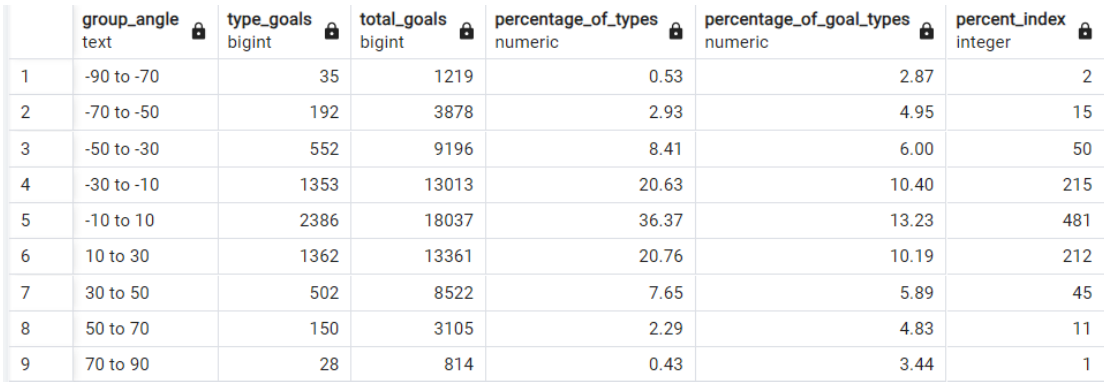
Here, we analyse the different shots that lead to goals. Index (percent_index) is the product of the % of goals (distribution of gaols) for that type (tip-in, slap shot, etc..) and % of goal over total goals at each distance group and angle group. Final index for heatmap is the product of distance_index and angle_index.

- **Tip-In:** A Tip-In goal normally happens at close range at very narrow angles.

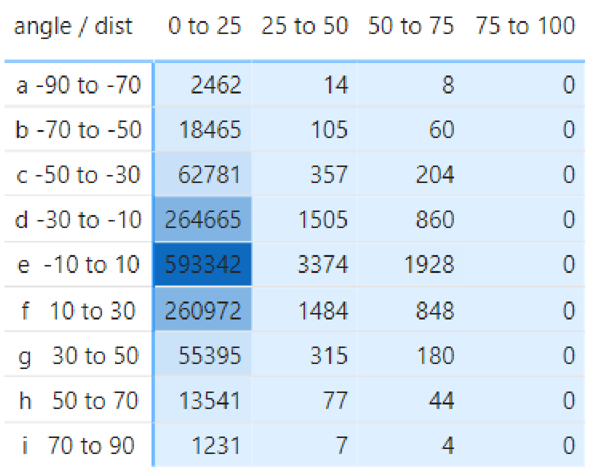
  (Tip-In Tables example)



	group_distance text	type_goals bigint	total_goals bigint	percentage_of_types numeric	percentage_of_goal_types numeric	percent_index integer
1	0 to 25	6154	46940	93.81	13.11	1230
2	25 to 50	290	18641	4.42	1.56	7
3	50 to 75	113	4907	1.72	2.30	4
4	75 to 100	3	657	0.05	0.46	0

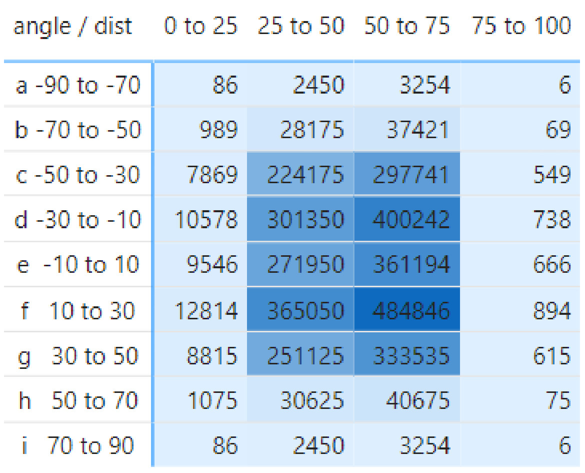


	group_angle text	type_goals bigint	total_goals bigint	percentage_of_types numeric	percentage_of_goal_types numeric	percent_index integer
1	-90 to -70	35	1219	0.53	2.87	2
2	-70 to -50	192	3878	2.93	4.95	15
3	-50 to -30	552	9196	8.41	6.00	50
4	-30 to -10	1353	13013	20.63	10.40	215
5	-10 to 10	2386	18037	36.37	13.23	481
6	10 to 30	1362	13361	20.76	10.19	212
7	30 to 50	502	8522	7.65	5.89	45
8	50 to 70	150	3105	2.29	4.83	11
9	70 to 90	28	814	0.43	3.44	1



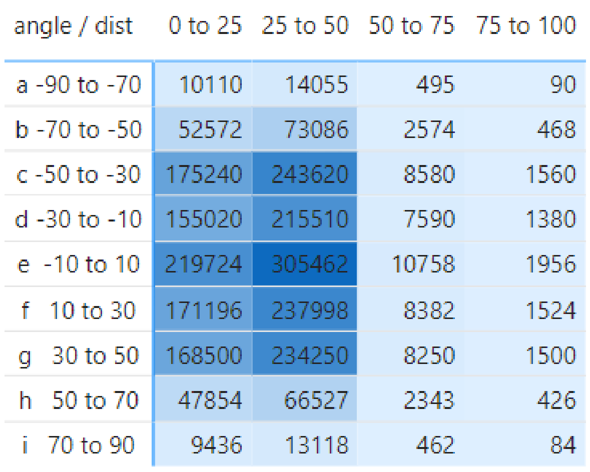
angle / dist	0 to 25	25 to 50	50 to 75	75 to 100
a -90 to -70	2462	14	8	0
b -70 to -50	18465	105	60	0
c -50 to -30	62781	357	204	0
d -30 to -10	264665	1505	860	0
e -10 to 10	593342	3374	1928	0
f 10 to 30	260972	1484	848	0
g 30 to 50	55395	315	180	0
h 50 to 70	13541	77	44	0
i 70 to 90	1231	7	4	0

- **Slap Shot:** Compared to a Tip-In, Slap Shot goals happen from a distance and wider angles.



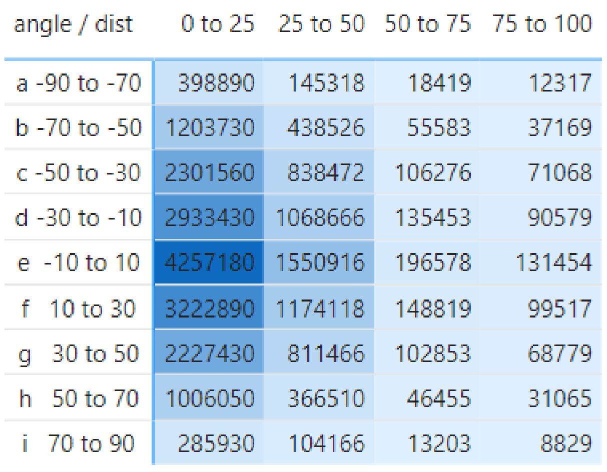
angle / dist	0 to 25	25 to 50	50 to 75	75 to 100
a -90 to -70	86	2450	3254	6
b -70 to -50	989	28175	37421	69
c -50 to -30	7869	224175	297741	549
d -30 to -10	10578	301350	400242	738
e -10 to 10	9546	271950	361194	666
f 10 to 30	12814	365050	484846	894
g 30 to 50	8815	251125	333535	615
h 50 to 70	1075	30625	40675	75
i 70 to 90	86	2450	3254	6

- **Snap Shot:** A Snap Shot comes in between a Tip-In and a Slap-Shot and goals happen in mid ranges and a range of wide angles.



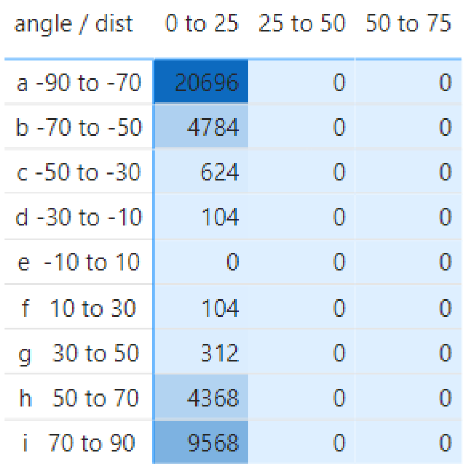
angle / dist	0 to 25	25 to 50	50 to 75	75 to 100
a -90 to -70	10110	14055	495	90
b -70 to -50	52572	73086	2574	468
c -50 to -30	175240	243620	8580	1560
d -30 to -10	155020	215510	7590	1380
e -10 to 10	219724	305462	10758	1956
f 10 to 30	171196	237998	8382	1524
g 30 to 50	168500	234250	8250	1500
h 50 to 70	47854	66527	2343	426
i 70 to 90	9436	13118	462	84

- **Wrist Shot:** A more versatile and flexible shot and goal is effective over wider distances and angle. 50% of all shots and goals are from Wrist Shots



angle / dist	0 to 25	25 to 50	50 to 75	75 to 100
a -90 to -70	398890	145318	18419	12317
b -70 to -50	1203730	438526	55583	37169
c -50 to -30	2301560	838472	106276	71068
d -30 to -10	2933430	1068666	135453	90579
e -10 to 10	4257180	1550916	196578	131454
f 10 to 30	3222890	1174118	148819	99517
g 30 to 50	2227430	811466	102853	68779
h 50 to 70	1006050	366510	46455	31065
i 70 to 90	285930	104166	13203	8829

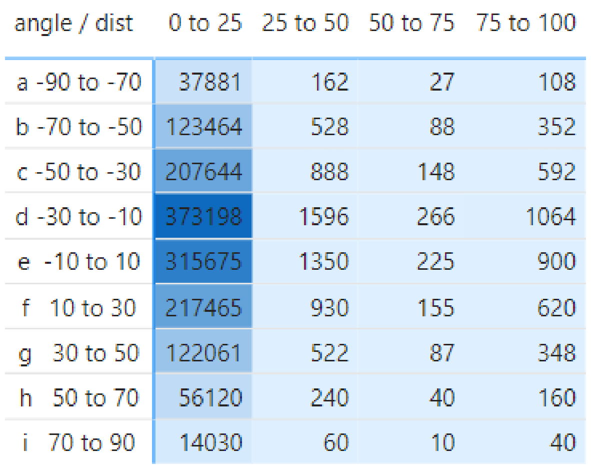
- **Wrap Around:** A Wrap Around is a shooting technique that score at a large angle near or almost behind the goal line. The occurrence is rare (about 1% of shots), but it is effective at these difficult positions. Wrap Around is almost never used at distance greater than 25 ft from goal.



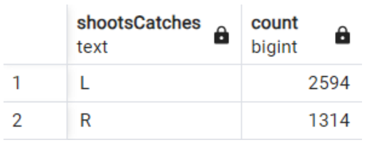
angle / dist	0 to 25	25 to 50	50 to 75
a -90 to -70	20696	0	0
b -70 to -50	4784	0	0
c -50 to -30	624	0	0
d -30 to -10	104	0	0
e -10 to 10	0	0	0
f 10 to 30	104	0	0
g 30 to 50	312	0	0
h 50 to 70	4368	0	0
i 70 to 90	9568	0	0

- **Backhand:** Backhand goal is executed from the reverse side of the hitting hand. Goals happen at short distances but it's relatively less powerful.

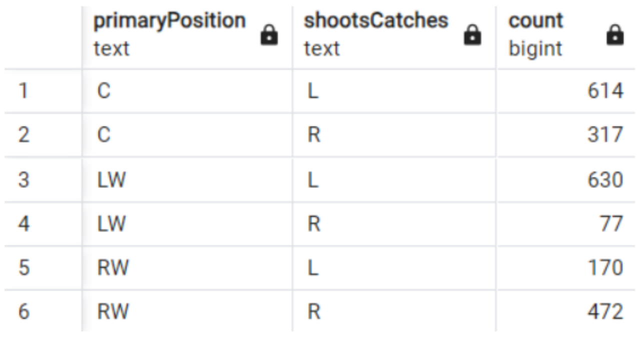
  Backhand goals in NHL are skewed towards the Right Wing ($y < 0$) because there are more left than right hitting players. Number of players using Left hand to shoot, or catch are twice that of the Right. Left/Right hand shooters are positioned at Left/Right Wing respectively. For a reverse shot like the backhand, we will see more on the Right Wing executed by Left hand shooters



angle / dist	0 to 25	25 to 50	50 to 75	75 to 100
a -90 to -70	37881	162	27	108
b -70 to -50	123464	528	88	352
c -50 to -30	207644	888	148	592
d -30 to -10	373198	1596	266	1064
e -10 to 10	315675	1350	225	900
f 10 to 30	217465	930	155	620
g 30 to 50	122061	522	87	348
h 50 to 70	56120	240	40	160
i 70 to 90	14030	60	10	40



	shootsCatches text	count bigint
1	L	2594
2	R	1314



	primaryPosition text	shootsCatches text	count bigint
1	C	L	614
2	C	R	317
3	LW	L	630
4	LW	R	77
5	RW	L	170
6	RW	R	472

X-Y Coordinates Analysis Conclusion

Our analysis shows the use of different shot depends on both distance from goal and angle to goal, with distance having a bigger contribution than angle. Moving from the goal post to the centre of the rink, Tip-in and Backhand are used at close range of 0-25 ft from goal. Snap Shot comes at 25-50 ft from goal and Slap Shot is used for long distance normally 50 to 75 ft from goal.

Tip-In goals normally happen at narrow angles to goal (-30 to 30) while a Wrap Around converts goal at very wide angles (-50 to –90 or 50 to 90). Other shot types are relatively independent of angle to goal.

Wrist Shot is the most common and versatile goal conversion shot. This shot type can be used in a wide range of distances and angles.

9.2: NPC Dream Team Analysis

Problem Statement

In performing this analysis, the team seeks to address the following problem statements:

- How do we determine which players, based on historical NHL data, would form the optimal Dream Team?
- What are the key performance metrics that should be considered when selecting top-performing players for their relative positions? (Defensemen, Forwards, Goalie)

To answer this, we will analyze NHL data from the 2011 to 2019 seasons to find the best candidates for the Dream NHL Team, using the following tables

- player_info
- game_skater_stats
- team_info
- game_goalie_stats

<u>Approach and Proposal</u>

Research suggests that the peak performance age in the NHL is around 28 years old, after which performance may stagnate or decline. Therefore, the focus will be on young players under 26 years old who have the potential to rank among the top 100 in their field of expertise from 2021 to 2023.

This analysis identifies the top 23 players out of 3,925 in the NHL dataset based on their performance metrics. A composite scoring system, with a maximum of 10 points, is used to categorize the players in each of the three key player types: forwards, defensemen, and goalies.

<u>1. Determining Top 12 Forwards</u>

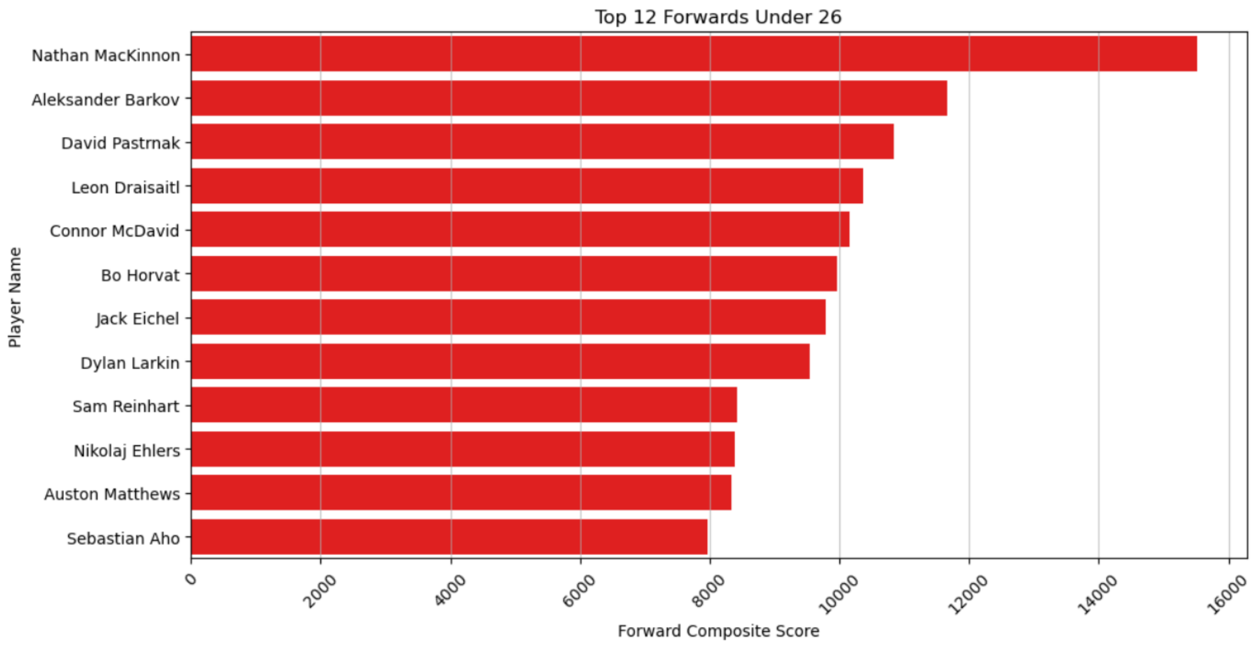
To assess the forwards, we calculated a Forward Composite Score that provides a comprehensive view of their offensive contributions. This score was derived using the following weighted metrics:

- Goals: 3 points (The number of goals the player scored during the game. It measures the player's offensive productivity.)
- Assists: 1 point (The number of assists the player recorded during the game. awarded when player helps to set up a goal through shot, pass or deflect towards a scoring teammate., highlighting their playmaking abilities.)
- Shots: 3 points (The number of shots attempted by the player. It indicates offensive activity and the player’s ability to generate scoring chances.)

- Time on Ice (Normalized): 3 points (Total time the player spent on the ice during the game, measured in seconds. It reflects the player's overall involvement and usage during the game.

```
# Composite Score for Forwards with TOI normalization
forward_players.loc[:, 'ForwardCompositeScore'] = (
    (forward_players['goals'] * 3) +
    (forward_players['shots'] * 3) +
    (forward_players['assists'] * 1) +
    (forward_players['timeonice'] / 60 * 3) / 4
)
```

Based on this, we arrive at a ranked list for each player based on their Offensive Composite Score.

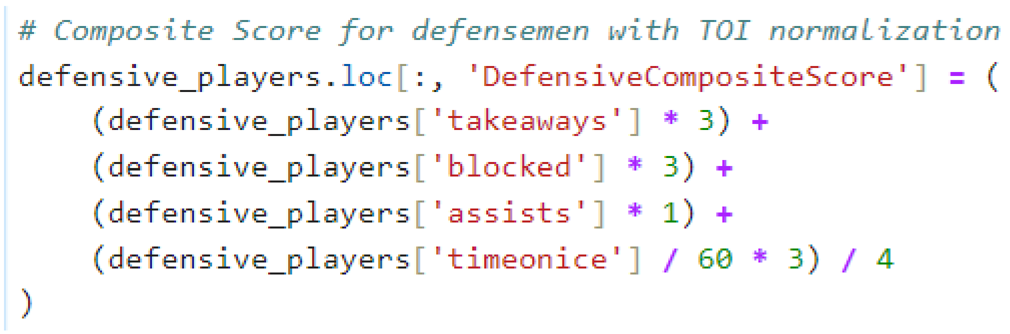


<u>2. Determining the top 8 Defensive Players</u>

The analysis calculated a Defensive Composite Score for defensemen, providing insight into their effectiveness in defensive roles. This score was derived using the following weighted metrics:

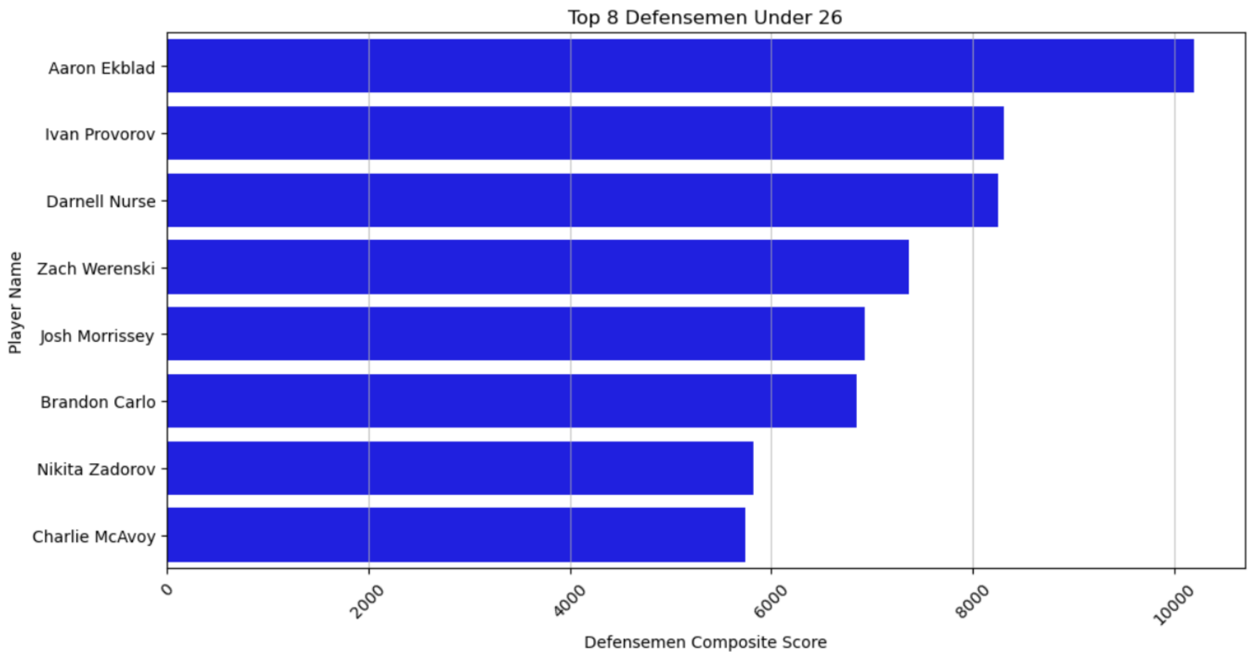
- Takeaways: 3 points (The number of times the player successfully steals the puck from an opponent. Indicates defensive skill and puck management.)
- Blocked Shots: 3 points (The number of shots the player blocked. Indicates defensive prowess and willingness to sacrifice body to prevent goals.)

- Assists: 1 point (The number of assists the player recorded during the game. awarded when player helps to set up a goal through shot, pass or deflect towards a scoring teammate., highlighting their playmaking abilities.)
- Time on Ice (Normalized): 3 points (Total time the player spent on the ice during the game, measured in seconds. It reflects the player's overall involvement and usage during the game.



```
# Composite Score for defensemen with TOI normalization
defensive_players.loc[:, 'DefensiveCompositeScore'] = (
    (defensive_players['takeaways'] * 3) +
    (defensive_players['blocked'] * 3) +
    (defensive_players['assists'] * 1) +
    (defensive_players['timeonice'] / 60 * 3) / 4
)
```

Based on this, we arrive at a ranked list for each player based on their Defensive Composite Score:



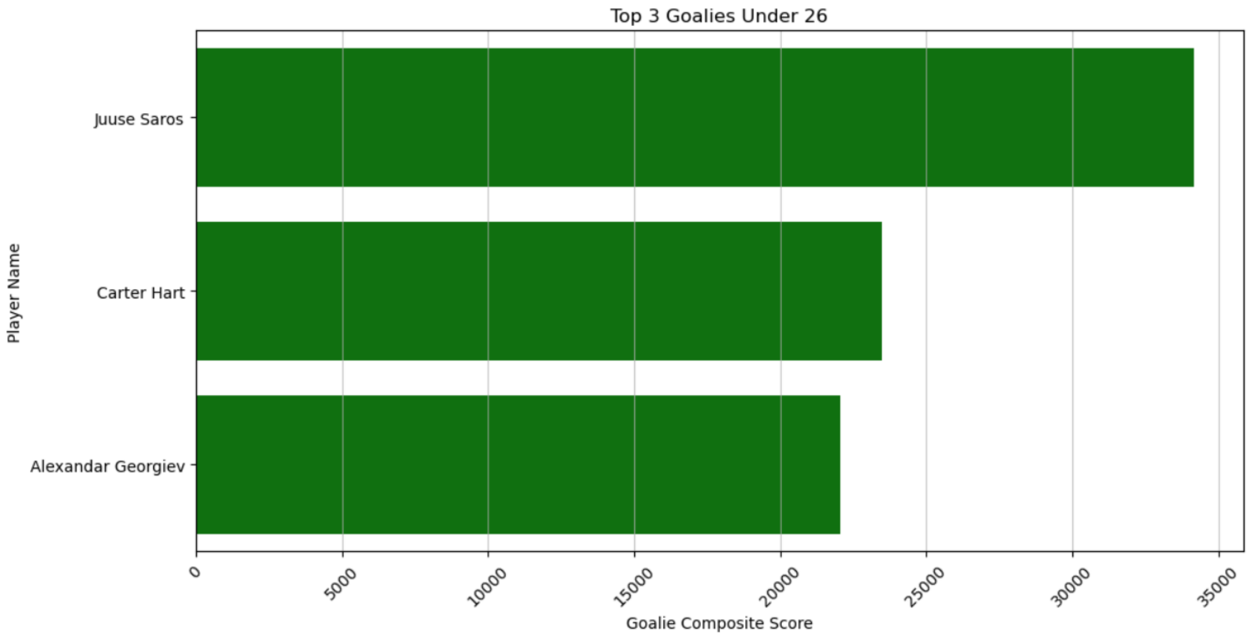
3. Top 3 Goalies

The analysis evaluated goalies using a Goalie Composite Score, which reflects their effectiveness in preventing goals. This score was calculated based on the following weighted metrics:

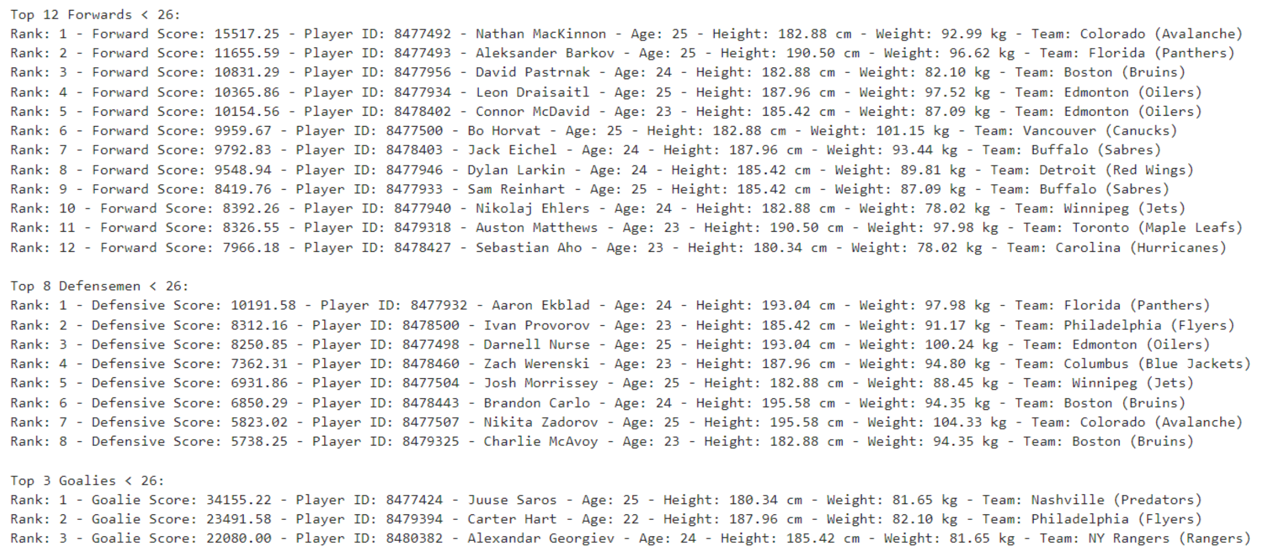
- Saves: 7 points (Total number of saves made by the goalie.)
- Time on Ice (Normalized): 3 points (Total time the player spent on the ice during the game, measured in seconds. It reflects the player's overall involvement and usage during the game.

```
# Composite Score for goalies with TOI normalization
goalie_players.loc[:, 'GoalieCompositeScore'] = (
    (goalie_players['saves'] * 7) +
    (goalie_players['timeonice'] / 60 * 3) / 2
)
```

Based on this, we arrive at a ranked list for each player based on their Goalie Composite Score:



Consolidate results:



```
Top 12 Forwards < 26:
Rank: 1 - Forward Score: 15517.25 - Player ID: 8477492 - Nathan MacKinnon - Age: 25 - Height: 182.88 cm - Weight: 92.99 kg - Team: Colorado (Avalanche)
Rank: 2 - Forward Score: 11655.59 - Player ID: 8477493 - Aleksander Barkov - Age: 25 - Height: 190.50 cm - Weight: 96.62 kg - Team: Florida (Panthers)
Rank: 3 - Forward Score: 10831.29 - Player ID: 8477956 - David Pastrnak - Age: 24 - Height: 182.88 cm - Weight: 82.10 kg - Team: Boston (Bruins)
Rank: 4 - Forward Score: 10365.86 - Player ID: 8477934 - Leon Draisaitl - Age: 25 - Height: 187.96 cm - Weight: 97.52 kg - Team: Edmonton (Oilers)
Rank: 5 - Forward Score: 10154.56 - Player ID: 8478402 - Connor McDavid - Age: 23 - Height: 185.42 cm - Weight: 87.09 kg - Team: Edmonton (Oilers)
Rank: 6 - Forward Score: 9959.67 - Player ID: 8477500 - Bo Horvat - Age: 25 - Height: 182.88 cm - Weight: 101.15 kg - Team: Vancouver (Canucks)
Rank: 7 - Forward Score: 9792.83 - Player ID: 8478403 - Jack Eichel - Age: 24 - Height: 187.96 cm - Weight: 93.44 kg - Team: Buffalo (Sabres)
Rank: 8 - Forward Score: 9548.94 - Player ID: 8477946 - Dylan Larkin - Age: 24 - Height: 185.42 cm - Weight: 89.81 kg - Team: Detroit (Red Wings)
Rank: 9 - Forward Score: 8419.76 - Player ID: 8477933 - Sam Reinhart - Age: 25 - Height: 185.42 cm - Weight: 87.09 kg - Team: Buffalo (Sabres)
Rank: 10 - Forward Score: 8392.26 - Player ID: 8477940 - Nikolaj Ehlers - Age: 24 - Height: 182.88 cm - Weight: 78.02 kg - Team: Winnipeg (Jets)
Rank: 11 - Forward Score: 8326.55 - Player ID: 8479318 - Auston Matthews - Age: 23 - Height: 190.50 cm - Weight: 97.98 kg - Team: Toronto (Maple Leafs)
Rank: 12 - Forward Score: 7966.18 - Player ID: 8478427 - Sebastian Aho - Age: 23 - Height: 180.34 cm - Weight: 78.02 kg - Team: Carolina (Hurricanes)

Top 8 Defensemen < 26:
Rank: 1 - Defensive Score: 10191.58 - Player ID: 8477932 - Aaron Ekblad - Age: 24 - Height: 193.04 cm - Weight: 97.98 kg - Team: Florida (Panthers)
Rank: 2 - Defensive Score: 8312.16 - Player ID: 8478500 - Ivan Provorov - Age: 23 - Height: 185.42 cm - Weight: 91.17 kg - Team: Philadelphia (Flyers)
Rank: 3 - Defensive Score: 8250.85 - Player ID: 8477498 - Darnell Nurse - Age: 25 - Height: 193.04 cm - Weight: 100.24 kg - Team: Edmonton (Oilers)
Rank: 4 - Defensive Score: 7362.31 - Player ID: 8478460 - Zach Werenski - Age: 23 - Height: 187.96 cm - Weight: 94.80 kg - Team: Columbus (Blue Jackets)
Rank: 5 - Defensive Score: 6931.86 - Player ID: 8477504 - Josh Morrissey - Age: 25 - Height: 182.88 cm - Weight: 88.45 kg - Team: Winnipeg (Jets)
Rank: 6 - Defensive Score: 6850.29 - Player ID: 8478443 - Brandon Carlo - Age: 24 - Height: 195.58 cm - Weight: 94.35 kg - Team: Boston (Bruins)
Rank: 7 - Defensive Score: 5823.02 - Player ID: 8477507 - Nikita Zadorov - Age: 25 - Height: 195.58 cm - Weight: 104.33 kg - Team: Colorado (Avalanche)
Rank: 8 - Defensive Score: 5738.25 - Player ID: 8479325 - Charlie McAvoy - Age: 23 - Height: 182.88 cm - Weight: 94.35 kg - Team: Boston (Bruins)

Top 3 Goalies < 26:
Rank: 1 - Goalie Score: 34155.22 - Player ID: 8477424 - Juuse Saros - Age: 25 - Height: 180.34 cm - Weight: 81.65 kg - Team: Nashville (Predators)
Rank: 2 - Goalie Score: 23491.58 - Player ID: 8479394 - Carter Hart - Age: 22 - Height: 187.96 cm - Weight: 82.10 kg - Team: Philadelphia (Flyers)
Rank: 3 - Goalie Score: 22080.00 - Player ID: 8480382 - Alexandar Georgiev - Age: 24 - Height: 185.42 cm - Weight: 81.65 kg - Team: NY Rangers (Rangers)
```

NPC Dream Team Analysis Conclusion

This analysis reveals a wealth of young talent across all key positions, with each player bringing unique strengths to the NPC Dream Team. This diverse roster not only showcases individual excellence but also serves as a foundation for crafting compelling characters in game development. As these players continue to grow and evolve in the competitive world of hockey, they embody the potential for dynamic storytelling that enhances gameplay experiences, ultimately deepening engagement with gamers interested in the NHL.

Analysis Conclusion

ASMR has successfully analysed and narrated the gameplay mechanics and player dynamics for the gaming studio to build the next generation NHL computer game.

The X-Y coordinate analysis provides insights into gameplay techniques. Information about scoring positions (angle and distance to goal) and the selection of shot types used at various positions will be incorporated into the game strategy profile to strengthen the competitive realism of the computer game.

With these young talents analysis, the gaming studio can assemble a dream team of 23 NPC characters for the game. The composite score of these contemporary players, based on key weighted metrics, will enhance play engagement in the upcoming season through 2023.

10. Challenges

Throughout the project, the ASMR Research team encountered several challenges, which included navigating unfamiliar cloud technologies, managing resource constraints, and handling large volumes of data while adhering to tight deadlines.

Key challenges faced include:

- **Unfamiliarity with Azure Services:** While the team had experience building data pipelines, this was the first time they were implementing one in Azure. As such, they were unfamiliar with key Azure services and had to build up their Azure knowledge to be able to select the appropriate services for building the ETL pipeline. This learning process required substantial research and understanding of the resources to use, as well as how to integrate these services to set up the pipeline infrastructure.
- **Limited Access to Cloud Resources and Cost Management:** Due to license restrictions on the Microsoft Azure account provided to the team, they were faced with constraints on costs and resource availability. Much time was spent configuring access of these resources and figuring out workarounds, as well as keeping a close watch on the costs to ensure that the budget was sufficient.
- **Understanding NHL Gameplay Rules and Regulations:** Another core challenge for the team was their unfamiliarity with NHL rules and the nuances of hockey gameplay. This knowledge gap made it difficult to contextualize the data, especially when selecting relevant metrics for player performance and scoring strategies. Gaining domain knowledge was crucial to making meaningful data-driven decisions.
- **Navigating Complex Data:** The project involved large volumes of historical NHL data. The team had to identify and filter relevant metrics from the dataset to extract insights, such as scoring probabilities and player performance. Managing this data and drawing accurate insights took a good amount of time and involved multiple discussions in order to land on the right fields.
- **Tight Turnaround:** The project was to be completed in a few weeks, which added time pressure on the team. Balancing their work responsibilities outside the project, along with steep learning curves in technology and game rules, made this a challenging project.

11. Solutions

To overcome these challenges, the team devised several solutions aimed at addressing both the technical and organizational barriers.

Key strategies included:

- **Researching Azure Services:** The team undertook comprehensive research and study how to build ETL pipelines using Azure services. They explored key services, compared alternatives, and selected the appropriate tools (e.g., Azure Data Factory for orchestrating workflows, Azure Databricks for data transformation, and Azure PostgreSQL for storage) based on the project's requirements and cost constraints.
- **Cost Management:** Conscious efforts were made to minimize cloud costs. The team opted for cost-effective solutions across Azure services, actively monitored and shut down resources when they were not in use. For example, Databricks clusters were deactivated immediately after use to avoid unnecessary expenses.
- **Understanding NHL Gameplay:** To bridge the gap in domain knowledge, the team spent time understanding NHL gameplay rules and regulations. This helped in identifying the key metrics in the dataset and aligning the data analysis with authentic game mechanics, making the insights more meaningful for performance analytics.
- **Collaborative Learning:** The team focused on collaborative learning sessions to upskill in Azure services and NHL data. Team members shared their learnings and findings from their independent research, which accelerated understanding and enhanced the project's knowledge base.
- **Parallel Task Execution and Project Management**: The project was split into defined roles and responsibilities, allowing team members to work in parallel on different tasks. Milestones were established to track progress and ensure all components of the project were completed on time. Close monitoring of project timelines helped mitigate the risk of delays and ensured that deadlines were met.

12. Conclusion

The project provided several important learnings for the team in the realm of data engineering, cloud services and data analysis. Specifically, the learnings are:

- **Developing ETL Pipelines Using Microsoft Azure:** The team gained hands-on experience in designing and implementing ETL pipelines using Azure Data Factory for orchestration, Databricks for data transformation (using Pandas and PySpark), and Azure PostgreSQL for data storage. This involved defining the data flow from raw ingestion to transformation and finally to analysis-ready datasets.
- **Data-Driven Decision Making in Sports:** Understanding NHL gameplay and correlating it with player performance data allowed the team to derive meaningful insights. This exercise helped the team appreciate the importance of domain knowledge when interpreting data, particularly in specialized fields like sports analytics.
- **Collaborative Learning and Adaptability:** The project highlighted the importance of adaptability and teamwork, especially in fast-paced and challenging environments. Despite facing a steep learning curve in cloud technology and domain-specific challenges, the team maintained a positive and growth-oriented mindset. They demonstrated the ability to quickly upskill and collaborate, allowing them to overcome their challenges and deliver the project successfully.

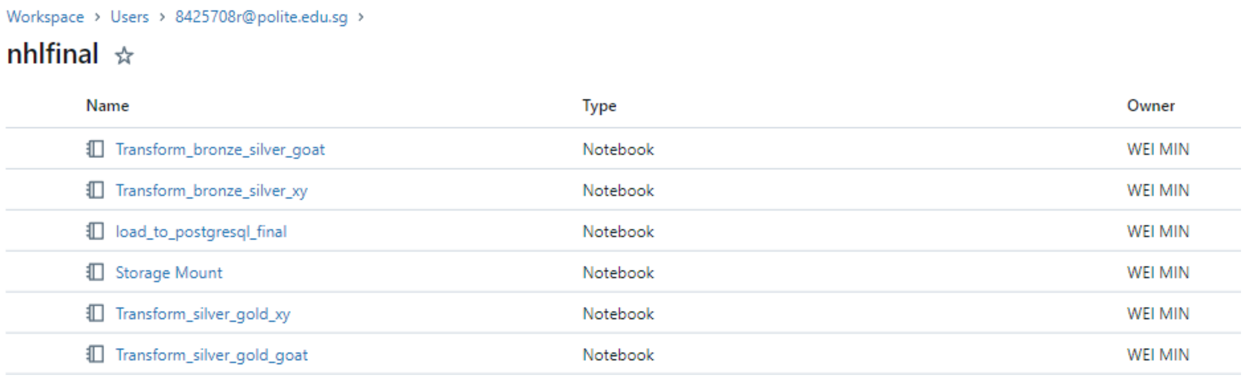
13. References

1. **Kaggle Dataset**
   Kaggle. (n.d.). *NHL Game Data*.
   Retrieved from https://www.kaggle.com
2. **NHL Rink and Rules**
   National Hockey League. (n.d.). *NHL Official Rules*.
   Retrieved from https://www.nhl.com/rules
3. **Medallion Architecture**
   Databricks. (n.d.). *Medallion Architecture*.
   Retrieved from https://www.databricks.com/glossary/medallion-architecture
4. **YouTube: An End-to-End Azure Data Engineering Real-Time Project Demo**
   Anadi, V. (2023). *An End-to-End Azure Data Engineering Real-Time Project Demo* [Video]. YouTube. Retrieved from https://youtu.be/iQ41WqhHglk?si=M62y7DIZSRb_N7g3
5. **Microsoft Learn Resources**
   a. Microsoft. (n.d.). *Mounts on Databricks File System (DBFS)*.
   Retrieved from https://learn.microsoft.com/en-us/azure/databricks/dbfs/mounts
   b. Microsoft. (n.d.). *Transform Data using Databricks Notebook in Azure Data Factory*.
   Retrieved from https://learn.microsoft.com/en-us/azure/data-factory/transform-data-using-databricks-notebook
6. **Mounting ADLS to Databricks**
   Microsoft Tech Community. (2023). *Mount ADLS Gen2 or Blob Storage in Azure Databricks*. Retrieved from https://techcommunity.microsoft.com/t5/azure-paas-blog/mount-adls-gen2-or-blob-storage-in-azure-databricks/ba-p/3802926
7. **Delta Files**
   Delta Lake. (2023). *Delta Lake vs Parquet Comparison*.
   Retrieved from https://delta.io/blog/delta-lake-vs-parquet-comparison/

14. Appendix

Appendix 1: Databricks Notebooks

The following notebooks have been created in Databricks workspace. Copies of the notebooks can also be found in the project folder.


Workspace › Users › 8425708r@polite.edu.sg ›

nhlfinal ☆

Name	Type	Owner
Transform_bronze_silver_goat	Notebook	WEI MIN
Transform_bronze_silver_xy	Notebook	WEI MIN
load_to_postgresql_final	Notebook	WEI MIN
Storage Mount	Notebook	WEI MIN
Transform_silver_gold_xy	Notebook	WEI MIN
Transform_silver_gold_goat	Notebook	WEI MIN

Appendix 2: Azure ADF Activity set up

Activity 1: ingest_kaggle_data

- Source
    - Dataset Name: Ingest_Binary1_Source
    - Linked Service: Ingest_Kaggle
        - Purpose: Linked service to connect and download NHL data from Kaggle
        - Type: HTTP
        - Base URL: https://www.kaggle.com/api/v1/datasets/download/martinellis/nhl-game-data
- Sink
    - Dataset Name: Ingest_Binary1_SaveNHLkaggleZIP
    - Linked Service: LS_AzureDataLakeStorage1
        - Purpose: Linked service to connect to ADLS Gen2
        - Type: Azure Data Lake Storage Gen2
        - URL: Points to the ADLS Gen2 storage account that was set up.

Activity 2: unzip_kaggle_data

- Source
    - Dataset Name: Ingest_unzip_source_2step
    - Linked Service: LS_AzureDataLakeStorage1
    - File Path: Points to the bronze folder in ADLS

    - Compression Type: ZipDeflate (.zip)
- Sink
    - Dataset Name: ingest_unzip_Dest_2step
    - Linked Service: LS_AzureDataLakeStorage1
    - File path: Points to the bronze folder in ADLS

The following notebook activities draw on Databricks notebooks to perform the transformation. Their notebook paths each link to the corresponding Databricks notebooks that have been setup in Azure Databricks workspace.

- Activity 3a: T_bronze_silver_xy
- Activity 3b: T_silver_gold_xy
- Activity 3c: T_bronze_silver_goat
- Activity 3d: T_silver_gold_goat
- Activity 4: Load_to_az_db