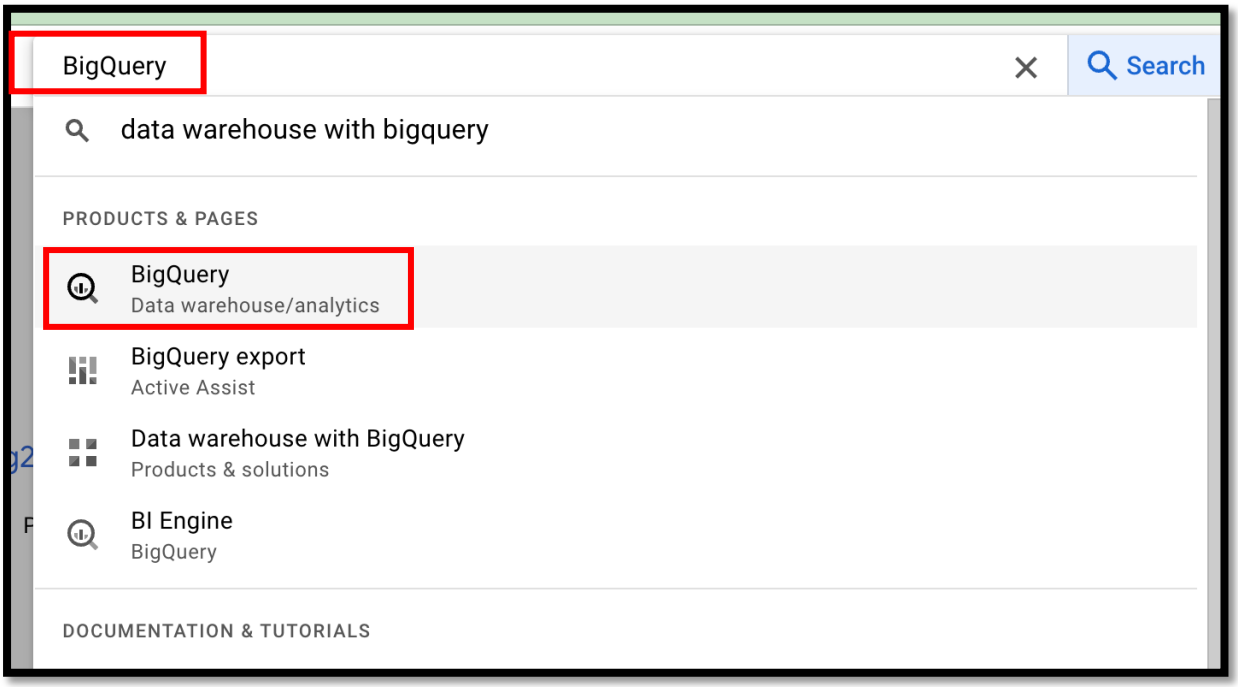
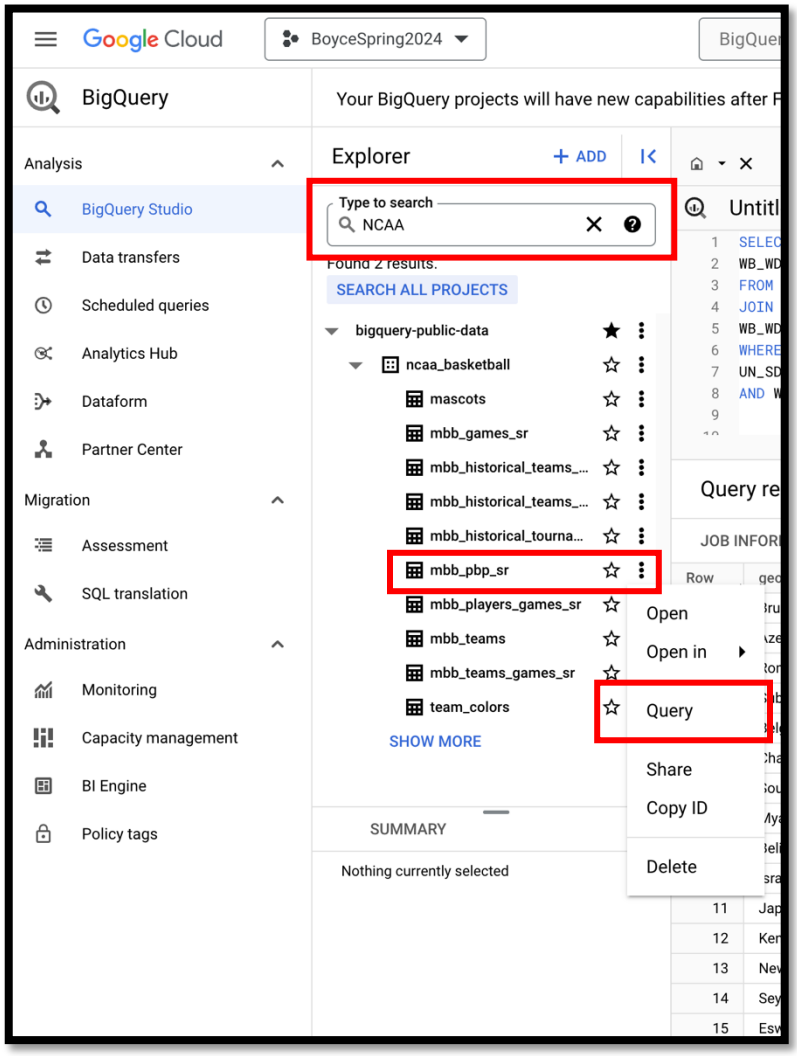
ADTA 5240: Data Harvesting
Querying Data with BigQuery

For this homework you will be using another public BigQuery dataset: NCAA Basketball Dataset. "This dataset contains data about NCAA Basketball games, teams, and players. Game data covers play-by-play and box scores back to 2009, as well as final scores back to 1996. Additional data about wins and losses goes back to the 1894-1895 season for some teams. All data runs through the end of the 2017-2018 season." (Much of the content is copied from Google)

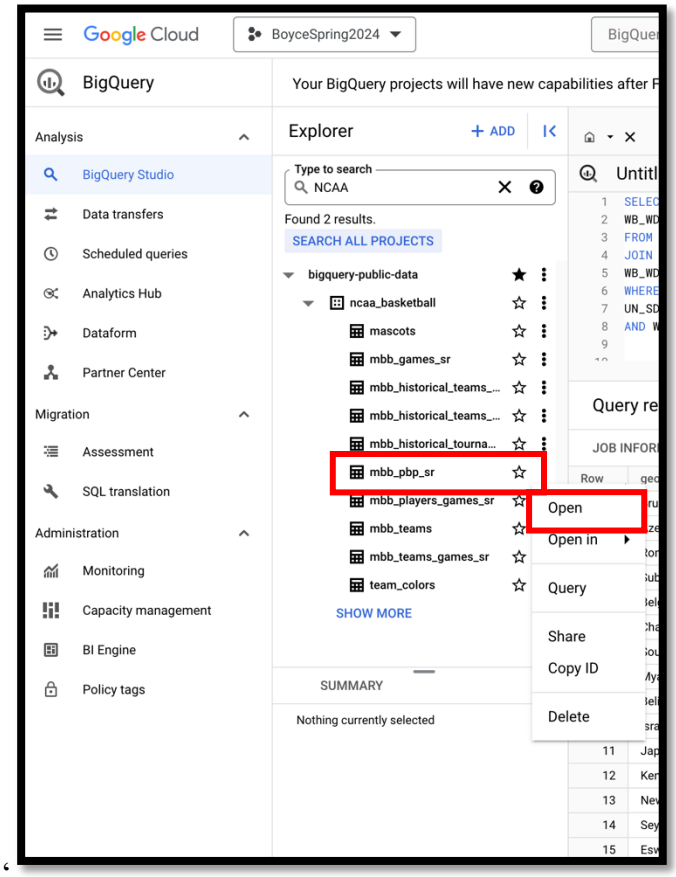
Step 1: Go to BigQuery. There are many ways to get to BigQuery. I just use the search bar. Type in BigQuery and click on BigQuery.

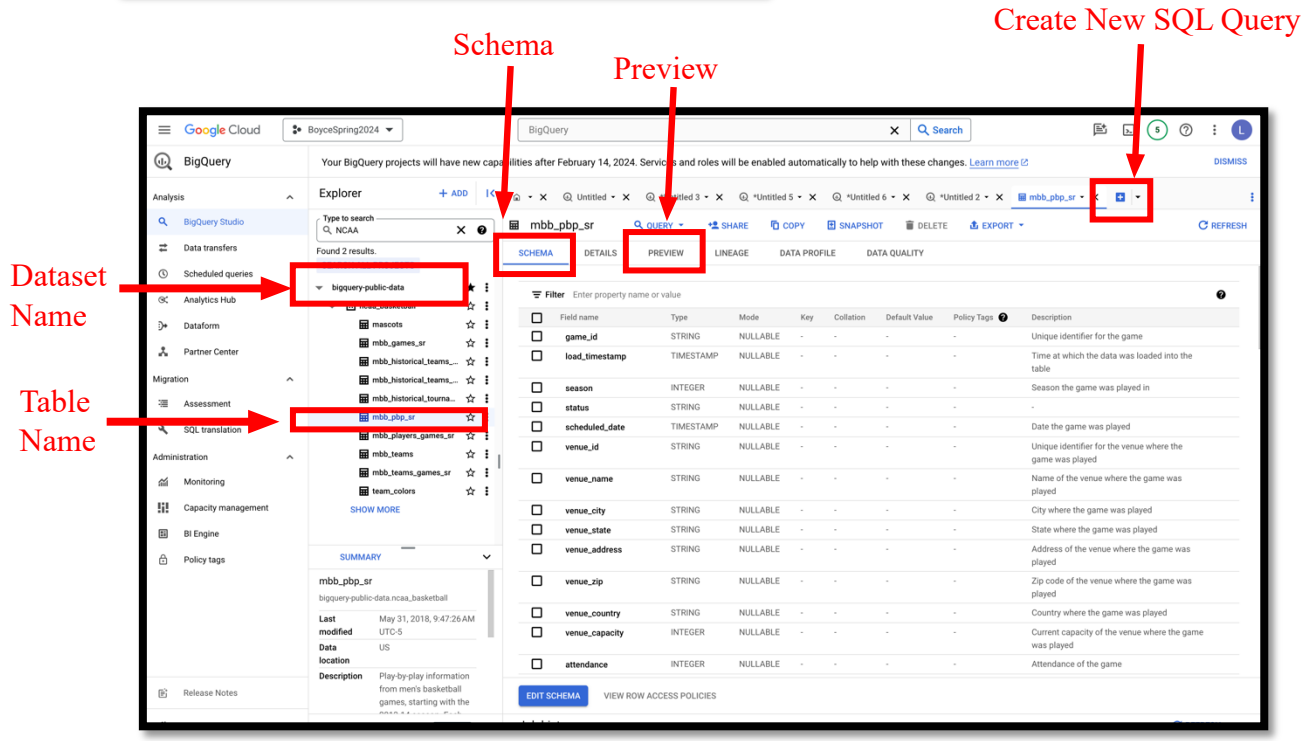


Step 2: You will use the NCAA Basketball Dataset. You have already worked with the public datasets in a previous assignment. If you forget how to get access, please revisit the previous homework. Go to the mbb_pbp_sr dataset to see the schema of this predefined table.



Step 3: I would like you to open the mbb_pbp_sr dataset to see the schema of this predefined table.
"This table has play-by-play information of all men's basketball games in the 2013–2014 season, and each row in the table represents a single event in a game." Click on preview to see more information about the dataset. Click on Details: How many rows are in the dataset?


Google Cloud
BoyceSpring2024
BigQuery
Your BigQuery projects will have new capabilities after F
Analysis
BigQuery Studio
Data transfers
Scheduled queries
Analytics Hub
Dataform
Partner Center
Migration
Assessment
SQL translation
Administration
Monitoring
Capacity management
BI Engine
Policy tags
Explorer
+ ADD
Type to search
NCAA
Found 2 results.
SEARCH ALL PROJECTS
bigquery-public-data
ncaa_basketball
mascots
mbb_games_sr
mbb_historical_teams_...
mbb_historical_teams_...
mbb_historical_tourna...
mbb_pbp_sr
mbb_players_games_sr
mbb_teams
mbb_teams_games_sr
team_colors
SHOW MORE
SUMMARY
Nothing currently selected
Open
Open in
Query
Share
Copy ID
Delete
Query re
JOB INFOR
Row


Schema
Preview
Create New SQL Query
Dataset Name
Table Name
Google Cloud
BoyceSpring2024
BigQuery
Search
Your BigQuery projects will have new capabilities after February 14, 2024. Services and roles will be enabled automatically to help with these changes. Learn more
DISMISS
Explorer
+ ADD
Type to search
NCAA
Found 2 results.
bigquery-public-data
mbb_pbp_sr
QUERY
SHARE
COPY
SNAPSHOT
DELETE
EXPORT
REFRESH
SCHEMA
DETAILS
PREVIEW
LINEAGE
DATA PROFILE
DATA QUALITY
Filter
Enter property name or value
Field name
Type
Mode
Key
Collation
Default Value
Policy Tags
Description
game_id
load_timestamp
season
status
scheduled_date
venue_id
venue_name
venue_city
venue_state
venue_address
venue_zip
venue_country
venue_capacity
attendance
SUMMARY
EDIT SCHEMA
VIEW ROW ACCESS POLICIES

Step 4: You will now use SQL to query the dataset (Click on the + sign). You will select the following columns from the table:

- `game_clock`: Time left in the game before the finish
- `points_scored` : Points were scored in an event
- `team_name`: Name of the team who scored the points
- `event_description`: Description about the event
- `timestamp`: Time when the event occurred

Use the following code to select these columns (see below). Click on "COMPOSE NEW QUERY" in the top right-hand side of the table schema (see above).

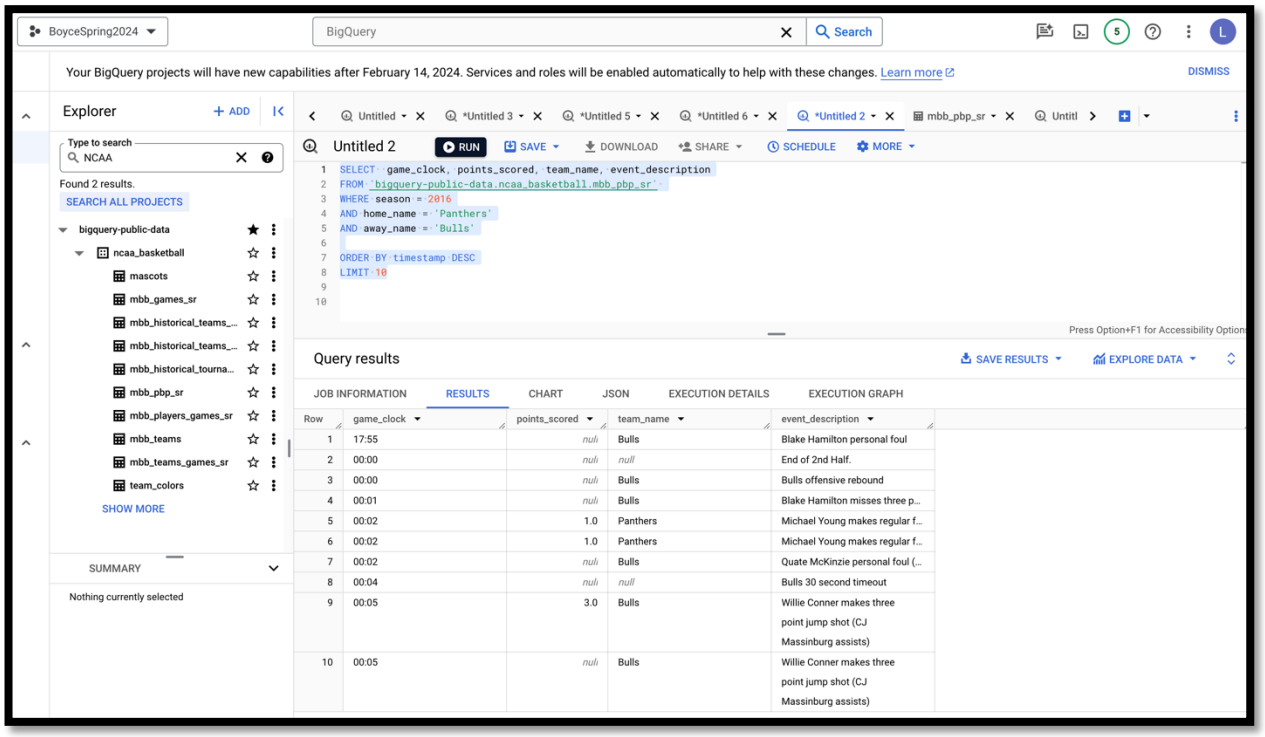
```
SELECT  game_clock, points_scored, team_name, event_description
FROM `bigquery-public-data.ncaa_basketball.mbb_pbp_sr`
WHERE season = 2016
AND home_name = 'Panthers'
AND away_name = 'Bulls'

ORDER BY timestamp DESC
LIMIT 10
```

In this query you are:

- Using the SELECT statement to retrieve the rows and the specified columns
- FROM the specified table
- Using the WHERE clause to filter the rows returned by SELECT, thus only returning the rows for the specific game we have listed.
- Using the ORDER BY statement controls the order of rows by the timestamp in descending order.
- Using the LIMIT statement to return only 10 events from the results after the rows are sorted. It is important to note that adding the statement "LIMIT" does not reduce the amount of data processed by the query engine. This is important when thinking about billing.

Don't forget to Click → "Run."

You should see the following results:



Using the above code, we see that Blake Hamilton made a personal foul at 17:55 on the game clock and Willie Conner made a three point jump shot at 00:05 on the game clock.;

Tip: In order to have better queries, avoid using SELECT * in the query. Instead query only the columns needed. To exclude only certain columns use SELECT * EXCEPT.

Step 5: Let's change the SQL code so the query includes a cumulative sum of scores for each team throughout the game. This can be done using analytic (window) functions. Analytic functions computes the aggregates for each row over a group of rows defined by a window whereas aggregate functions compute a single aggregate value over a group of rows.

Click on "COMPOSE NEW QUERY" in the top right-hand side of the table schema and use the following code to run the below query with two new columns added: wildcats_score and fighting_irish_score.

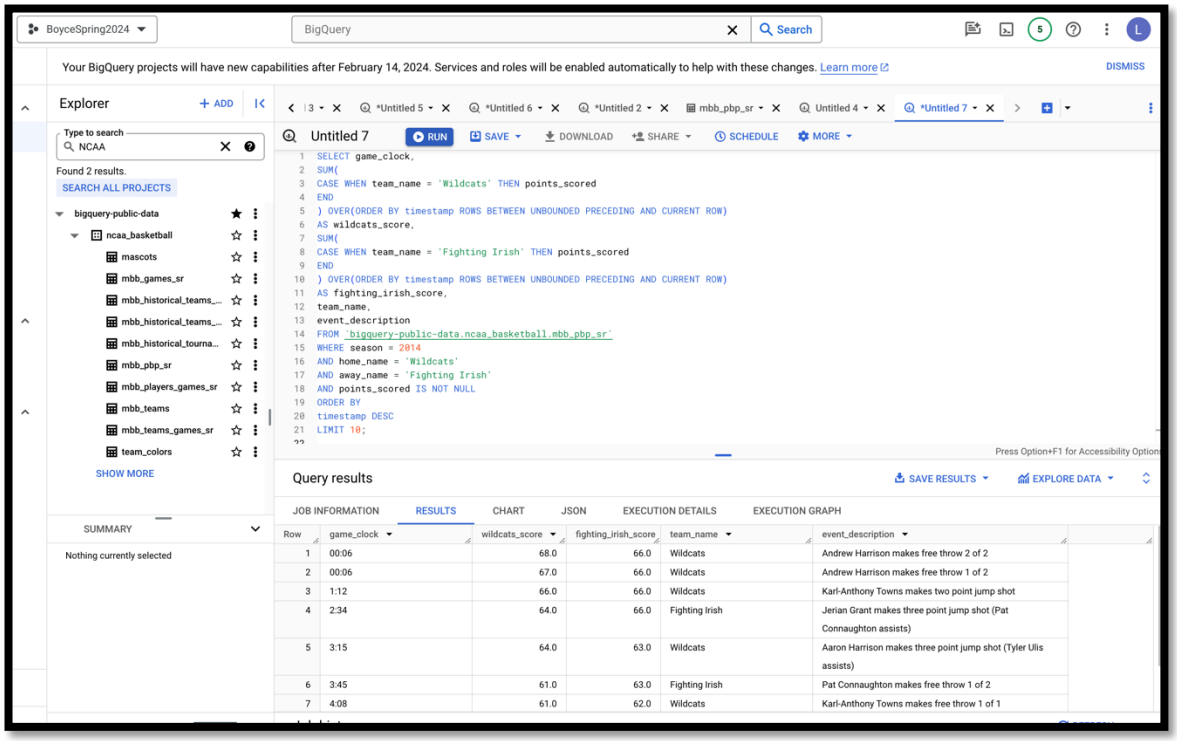
```
SELECT game_clock,
SUM(
CASE WHEN team_name = 'Wildcats' THEN points_scored
END
) OVER(ORDER BY timestamp ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
AS wildcats_score,
SUM(
CASE WHEN team_name = 'Fighting Irish' THEN points_scored
END
) OVER(ORDER BY timestamp ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
AS fighting_irish_score,
team_name,
event_description
FROM `bigquery-public-data.ncaa_basketball.mbb_pbp_sr`
WHERE season = 2014
AND home_name = 'Wildcats'
AND away_name = 'Fighting Irish'
AND points_scored IS NOT NULL
ORDER BY
timestamp DESC
LIMIT 10;
```

In this query you are:

- Calculating the cumulative SUM of scores by each team in the game. This is indicated by CASE statement
- The SUM is calculated on scores in the window indicated by the OVER clause
- The OVER clause references a "window" or a group of rows to use the SUM statement
- ORDER BY is part of the description of the group of rows (window) that defines sort order within that window. Here you see the query orders rows by timestamp
- Define the window frame from the start of the game specified by UNBOUNDED PRECEDING to the CURRENT ROW over which the analytic function SUM() is evaluated.

Don't forget to Click → "Run."

You should see the following results:



With this modified code, we can see the score throughout the game. We see the Fighting Irish were in the lead by four points until the last 04:28 of the game. Karl-Anthony Towns (Wildcats) tied the game on a layup with just over a minute remaining in the game. Then we see that Andrew Harrison made two free throws with just over 5 seconds left in the game, which put the Wildcats as the winning team.

Much of the content is copied and or paraphrased from Google and is retrieved from: https://cloud.google.com/blog/topics/developers-practitioners/bigquery-explained-queryingyour-data

More resources:

BigQuery code samples. Retrieved from: https://cloud.google.com/bigquery/docs/samples

BigQuery Documentation. Retrieved from: https://cloud.google.com/bigquery/docs

How to query Google Analytics Data in BigQuery Retrieved from: https://www.optimizesmart.com/query-google-analytics-data-in-bigquery/#a2