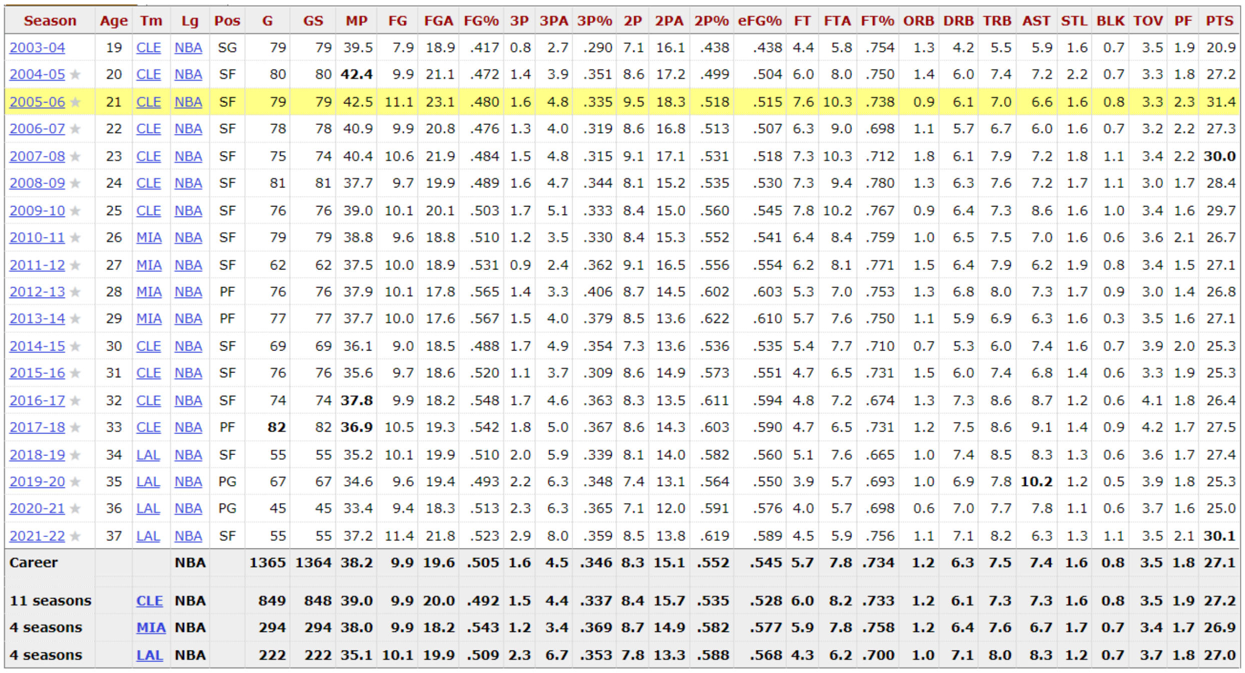
Group 2

- **Google Cloud has another processing service called DataProc. Name another processing service that is usually used in the cloud environment (not necessarily GCP). Compare between it and both Dataflow and DataProc. Your comparison may include but is not limited to the major differences, advantages, disadvantages, and limitations.**
    - **Service name:** Another processing service by Trifacta but is also under the company name of google cloud.
    - **Difference: Dataflow** is an easy to use streaming analytics service that prioritizes lowering cost, processing, time and latency to a minimum. This service can be utilized in both batches and streams. **DataProc** is an automatically scaling purpose or goal oriented build cluster that can assist in multiple analytical or data processing work. **Dataprep** is an intelligently designed service that specializes in cleaning, reporting, exploring, machine learning, and preparing both unstructured and structured learning of data.
    - **Advantages: Dataflow's main** advantage is that since it specializes in minimizing many aspects of data processing it is very efficient and cheap when it comes to using real-time sources and it’s simple to implement. **Dataproc’s** main advantage is due to its flexibility in scaling as multiple large data processing jobs can be done with it. **Dataprep’s** main advantage lies within its many features to organize and sequence data as well as its intelligent ability to adapt to new data using machine learning.
    - **Disadvantages: Dataflow’s** big disadvantage is that it still lacks certain features that are present in other services such as adaptability and flexibility along with that it is still in the early stages and does not have a very large community. **Dataproc’s** main disadvantage is that, although it’s cost may be cheaper depending on what you do, since its scale is large it has higher latency and processing is not as good as dataflow or dataprep. **Dataprep’s** main disadvantages lie on the fact that it is quite slow when handling larger amounts of data and that user-defined functions are not supported which makes it a bit more frustrating when you are using alternate ways to solve data problems.
    - **Limitations:**
        - **Dataflow’s:** Input shards 20000, Element value size 80Mb, Messages per time period 15000 every 30 sec, Metric/dataflow job 100.
        - **Dataproc:** Scaling operation request/minute 400, Cluster operation request/minute 200, Get job request/minute 7500, Workflow operation request/minute 400.

- **Dataprep:** # of workspaces/dataset in workspace 1000, # rules/dataset 500, File size exported to cloud 100Mb, Columns/dataset 333, Local file size importing 100Mb.

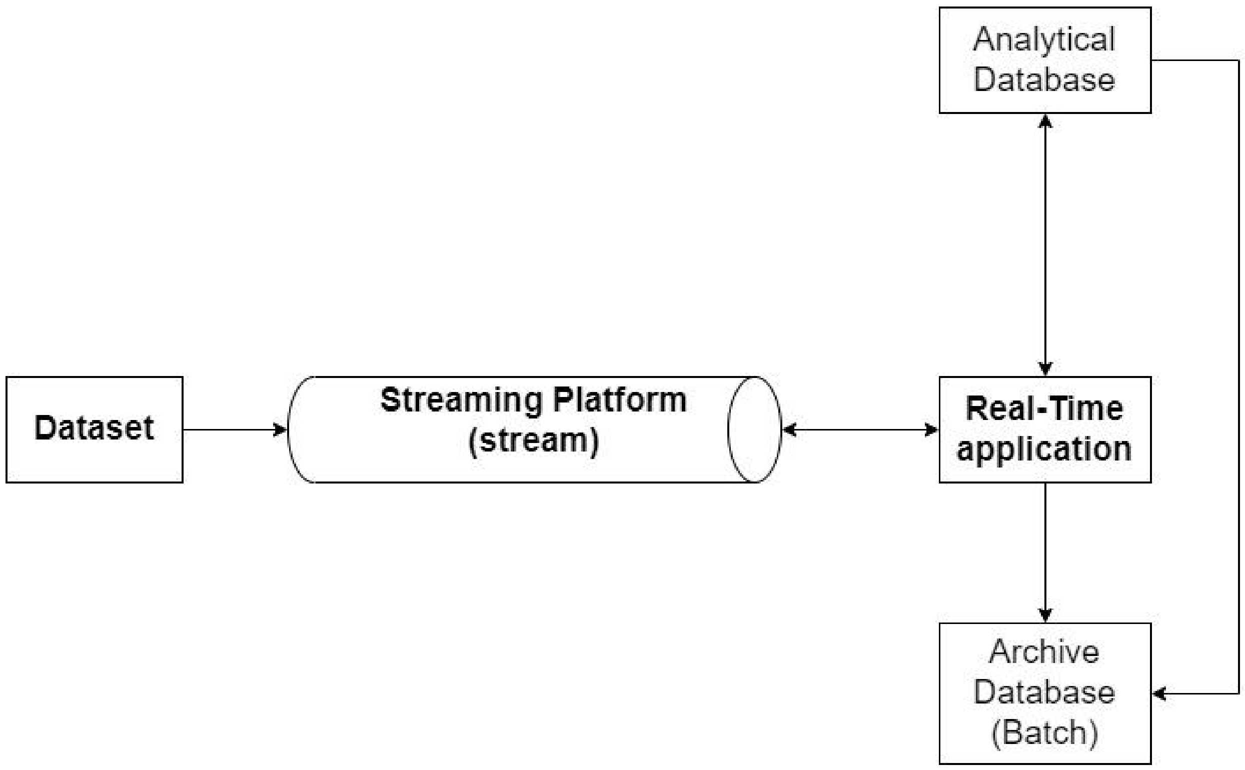
- **Suggest a practical application using both stream and batch processing that can be applied to a given dataset. It's expected to use the dataset uploaded in the third milestone but you can use any other dataset. If you decide to use another dataset, It should maintain both variety and huge volume. Your report should include but not limited to**
- **The dataset to be used:** *Athlete Statistics*
  - **The application:** Stream processing can be used during real-time events such as live display, updating, insertion, and deletion of these statistics during live games like soccer or basketball. Batch processing can be used for non real-time events such as simply inputting the statistics into the organizations (like NBA, or FIFA) server to keep records of each game in history.
  - **Its impact:** The use of the dataset will impact both real-time and non real-time events as the game will be streamed and updated while live then once completed will be sent to the organizations server.
  - **The used dataset (size, schema/structure):** This dataset in particular will be of the basketball player lebron james overall career from the year 2003-2022. The size will contain a total of 475 elements and the schema will be a table considering scores from "G" (games played) to "PTS" (points per game) so a 25x19 table = 475 values.



Season	Age	Tm	Lg	Pos	G	GS	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	eFG%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
2003-04	19	CLE	NBA	SG	79	79	39.5	7.9	18.9	.417	0.8	2.7	.290	7.1	16.1	.438	.438	4.4	5.8	.754	1.3	4.2	5.5	5.9	1.6	0.7	3.5	1.9	20.9
2004-05 ★	20	CLE	NBA	SF	80	80	**42.4**	9.9	21.1	.472	1.4	3.9	.351	8.6	17.2	.499	.504	6.0	8.0	.750	1.4	6.0	7.4	7.2	2.2	0.7	3.3	1.8	27.2
2005-06 ★	21	CLE	NBA	SF	79	79	42.5	11.1	23.1	.480	1.6	4.8	.335	9.5	18.3	.518	.515	7.6	10.3	.738	0.9	6.1	7.0	6.6	1.6	0.8	3.3	2.3	31.4
2006-07 ★	22	CLE	NBA	SF	78	78	40.9	9.9	20.8	.476	1.3	4.0	.319	8.6	16.8	.513	.507	6.3	9.0	.698	1.1	5.7	6.7	6.0	1.6	0.7	3.2	2.2	27.3
2007-08 ★	23	CLE	NBA	SF	75	74	40.4	10.6	21.9	.484	1.5	4.8	.315	9.1	17.1	.531	.518	7.3	10.3	.712	1.8	6.1	7.9	7.2	1.8	1.1	3.4	2.2	**30.0**
2008-09 ★	24	CLE	NBA	SF	81	81	37.7	9.7	19.9	.489	1.6	4.7	.344	8.1	15.2	.535	.530	7.3	9.4	.780	1.3	6.3	7.6	7.2	1.7	1.1	3.0	1.7	28.4
2009-10 ★	25	CLE	NBA	SF	76	76	39.0	10.1	20.1	.503	1.7	5.1	.333	8.4	15.0	.560	.545	7.8	10.2	.767	0.9	6.4	7.3	8.6	1.6	1.0	3.4	1.6	29.7
2010-11 ★	26	MIA	NBA	SF	79	79	38.8	9.6	18.8	.510	1.2	3.5	.330	8.4	15.3	.552	.541	6.4	8.4	.759	1.0	6.5	7.5	7.0	1.6	0.6	3.6	2.1	26.7
2011-12 ★	27	MIA	NBA	SF	62	62	37.5	10.0	18.9	.531	0.9	2.4	.362	9.1	16.5	.556	.554	6.2	8.1	.771	1.5	6.4	7.9	6.2	1.9	0.8	3.4	1.5	27.1
2012-13 ★	28	MIA	NBA	PF	76	76	37.9	10.1	17.8	.565	1.4	3.3	.406	8.7	14.5	.602	.603	5.3	7.0	.753	1.3	6.8	8.0	7.3	1.7	0.9	3.0	1.4	26.8
2013-14 ★	29	MIA	NBA	PF	77	77	37.7	10.0	17.6	.567	1.5	4.0	.379	8.5	13.6	.622	.610	5.7	7.6	.750	1.1	5.9	6.9	6.3	1.6	0.3	3.5	1.6	27.1
2014-15 ★	30	CLE	NBA	SF	69	69	36.1	9.0	18.5	.488	1.7	4.9	.354	7.3	13.6	.536	.535	5.4	7.7	.710	0.7	5.3	6.0	7.4	1.6	0.7	3.9	2.0	25.3
2015-16 ★	31	CLE	NBA	SF	76	76	35.6	9.7	18.6	.520	1.1	3.7	.309	8.6	14.9	.573	.551	4.7	6.5	.731	1.5	6.0	7.4	6.8	1.4	0.6	3.3	1.9	25.3
2016-17 ★	32	CLE	NBA	SF	74	74	**37.8**	9.9	18.2	.548	1.7	4.6	.363	8.3	13.5	.611	.594	4.8	7.2	.674	1.3	7.3	8.6	8.7	1.2	0.6	4.1	1.8	26.4
2017-18 ★	33	CLE	NBA	PF	**82**	82	**36.9**	10.5	19.3	.542	1.8	5.0	.367	8.6	14.3	.603	.590	4.7	6.5	.731	1.2	7.5	8.6	9.1	1.4	0.9	4.2	1.7	27.5
2018-19 ★	34	LAL	NBA	SF	55	55	35.2	10.1	19.9	.510	2.0	5.9	.339	8.1	14.0	.582	.560	5.1	7.6	.665	1.0	7.4	8.5	8.3	1.3	0.6	3.6	1.7	27.4
2019-20 ★	35	LAL	NBA	PG	67	67	34.6	9.6	19.4	.493	2.2	6.3	.348	7.4	13.1	.564	.550	3.9	5.7	.693	1.0	6.9	7.8	**10.2**	1.2	0.5	3.9	1.8	25.3
2020-21 ★	36	LAL	NBA	PG	45	45	33.4	9.4	18.3	.513	2.3	6.3	.365	7.1	12.0	.591	.576	4.0	5.7	.698	0.6	7.0	7.7	7.8	1.1	0.6	3.7	1.6	25.0
2021-22 ★	37	LAL	NBA	SF	55	55	37.2	11.4	21.8	.523	2.9	8.0	.359	8.5	13.8	.619	.589	4.5	5.9	.756	1.1	7.1	8.2	6.3	1.3	1.1	3.5	2.1	**30.1**
Career			**NBA**		**1365**	**1364**	**38.2**	**9.9**	**19.6**	**.505**	**1.6**	**4.5**	**.346**	**8.3**	**15.1**	**.552**	**.545**	**5.7**	**7.8**	**.734**	**1.2**	**6.3**	**7.5**	**7.4**	**1.6**	**0.8**	**3.5**	**1.8**	**27.1**
11 seasons		**CLE**	**NBA**		**849**	**848**	**39.0**	**9.9**	**20.0**	**.492**	**1.5**	**4.4**	**.337**	**8.4**	**15.7**	**.535**	**.528**	**6.0**	**8.2**	**.733**	**1.2**	**6.1**	**7.3**	**7.3**	**1.6**	**0.8**	**3.5**	**1.9**	**27.2**
4 seasons		**MIA**	**NBA**		**294**	**294**	**38.0**	**9.9**	**18.2**	**.543**	**1.2**	**3.4**	**.369**	**8.7**	**14.9**	**.582**	**.577**	**5.9**	**7.8**	**.758**	**1.2**	**6.4**	**7.6**	**6.7**	**1.7**	**0.7**	**3.4**	**1.7**	**26.9**
4 seasons		**LAL**	**NBA**		**222**	**222**	**35.1**	**10.1**	**19.9**	**.509**	**2.3**	**6.7**	**.353**	**7.8**	**13.3**	**.588**	**.568**	**4.3**	**6.2**	**.700**	**1.0**	**7.1**	**8.0**	**8.3**	**1.2**	**0.7**	**3.7**	**1.8**	**27.0**

Within the practical application database the structure (just like in the picture above) will have each column represent a different statistic score and each row being a generation or a (year) that those scores were taken.

- **A graph showing the proposed pipeline(s):**



- **List of other tools (AI, clustering,…) needed to implement that application:** The only other tool required for this application to be implemented is to have a data bufferer so that both real-time and archived data are presented to the viewer quickly, with minimal latency, and so that data does not become corrupted due to its size. Another possible tool you could use is AI (machine learning) so that predictions of future analytics can be presupposed or predicted and exist prior to the game so that when scores are entered for the next archive there will only need to be a change in some values for some columns and not others.