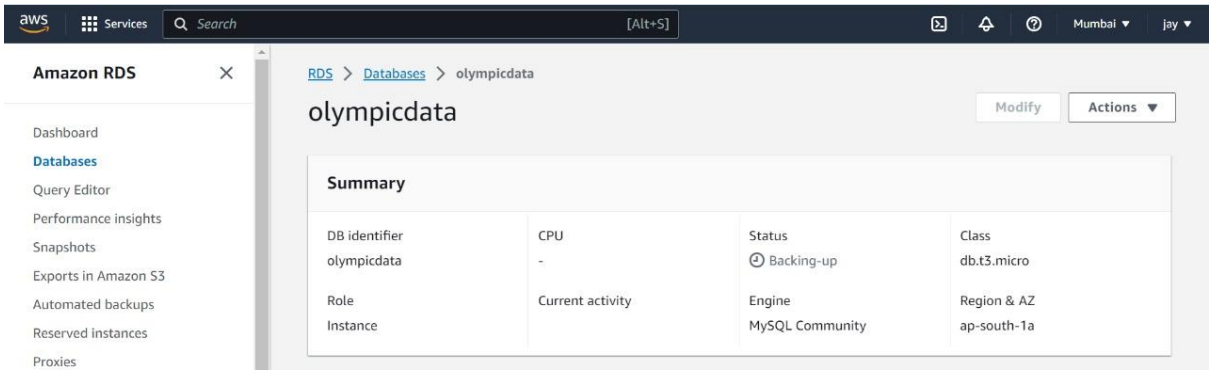
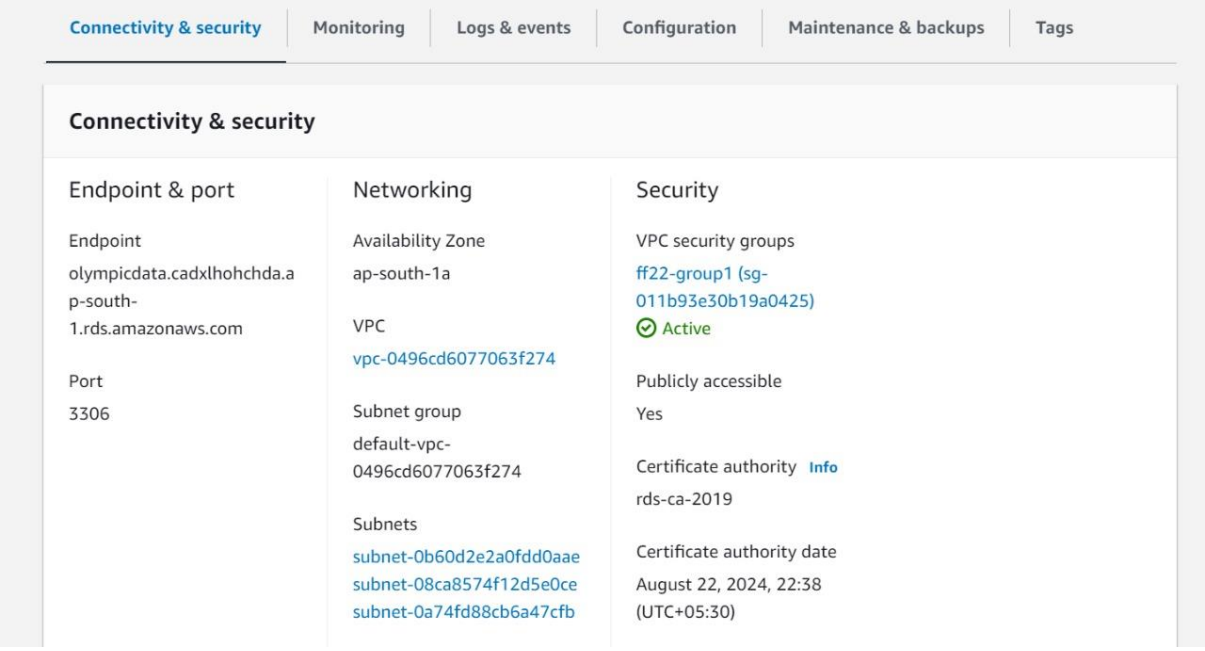
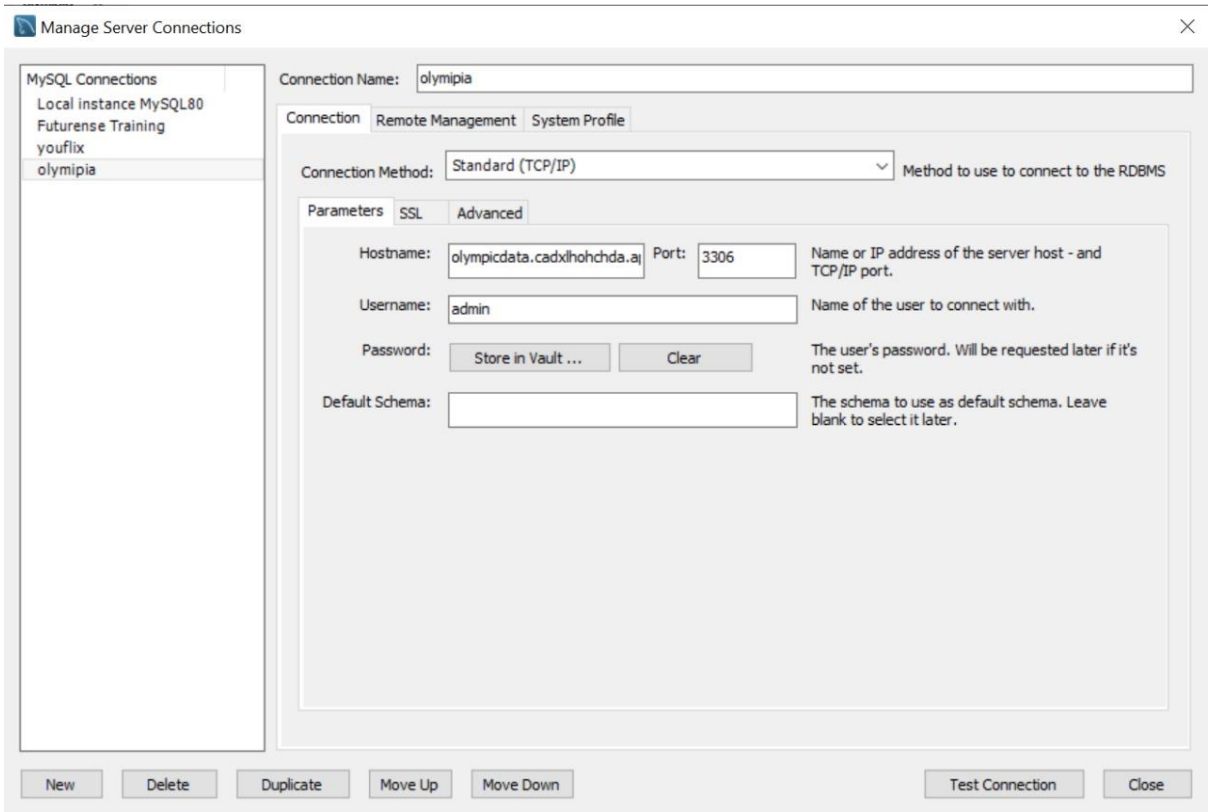
Step 1:- Creating Extraction pipelines from multiple client data sources to our data warehouse.

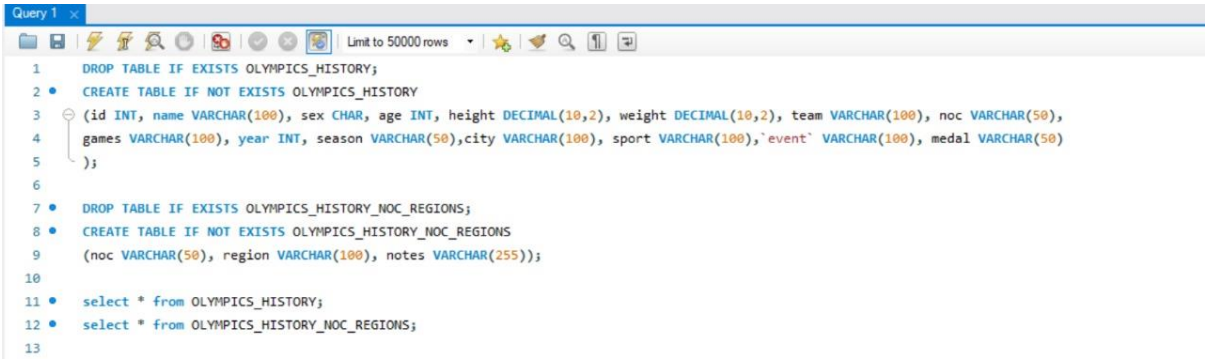
Pipeline 1- Amazon RDS or client relational database to Hive/HBase.

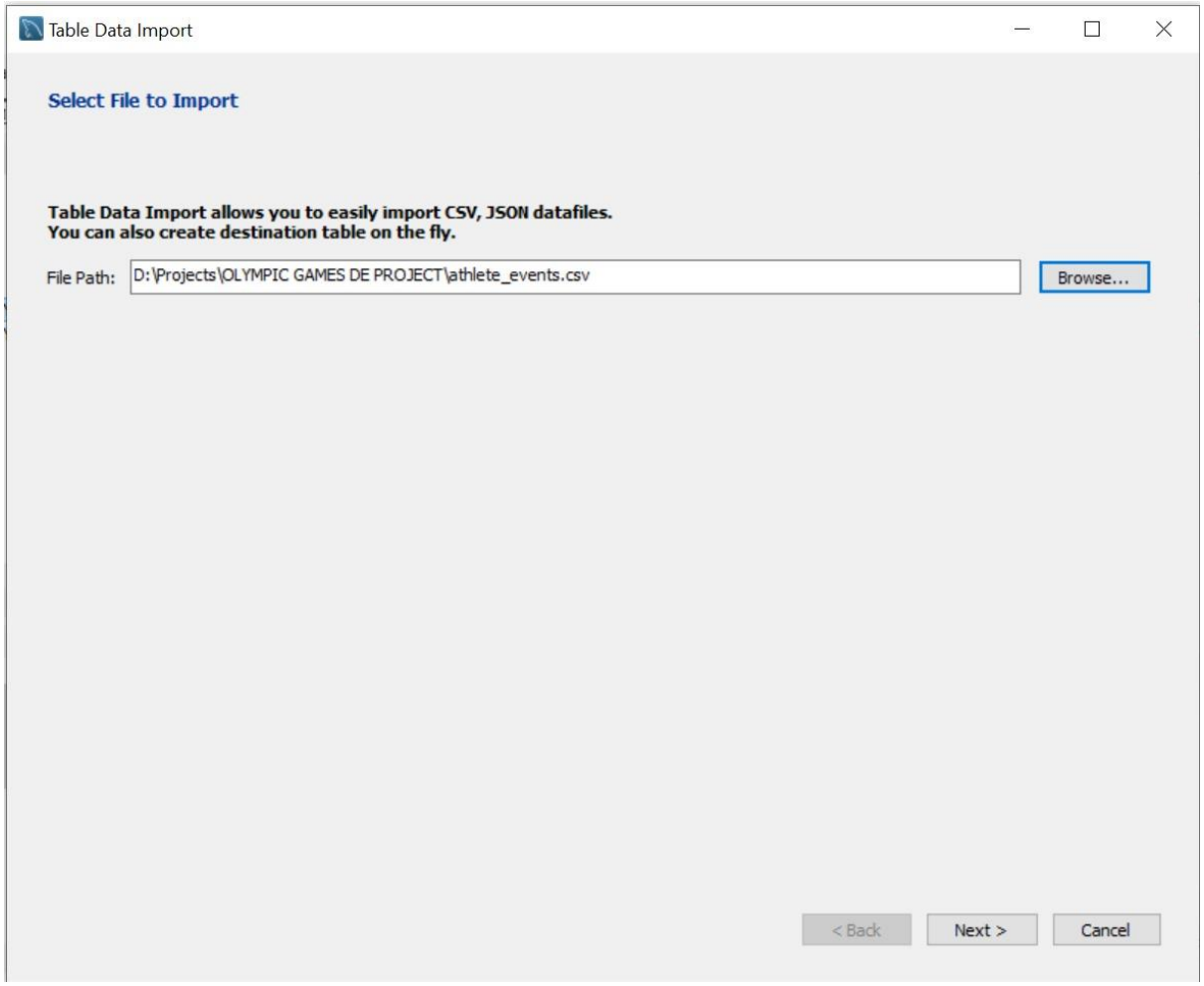
Ingestion Tool used: Sqoop






Manage Server Connections
MySQL Connections
Local instance MySQL80
Futurense Training
youflix
olymipia
Connection Name: olymipia
Connection
Remote Management
System Profile
Connection Method: Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters
SSL
Advanced
Hostname: olympicdata.cadxlhohchda.a
Port: 3306
Name or IP address of the server host - and TCP/IP port.
Username: admin
Name of the user to connect with.
Password:
Store in Vault ...
Clear
The user's password. Will be requested later if it's not set.
Default Schema:
The schema to use as default schema. Leave blank to select it later.
New
Delete
Duplicate
Move Up
Move Down
Test Connection
Close


Query 1
Limit to 50000 rows
DROP TABLE IF EXISTS OLYMPICS_HISTORY;
CREATE TABLE IF NOT EXISTS OLYMPICS_HISTORY
(id INT, name VARCHAR(100), sex CHAR, age INT, height DECIMAL(10,2), weight DECIMAL(10,2), team VARCHAR(100), noc VARCHAR(50),
games VARCHAR(100), year INT, season VARCHAR(50),city VARCHAR(100), sport VARCHAR(100),`event` VARCHAR(100), medal VARCHAR(50)
);
DROP TABLE IF EXISTS OLYMPICS_HISTORY_NOC_REGIONS;
CREATE TABLE IF NOT EXISTS OLYMPICS_HISTORY_NOC_REGIONS
(noc VARCHAR(50), region VARCHAR(100), notes VARCHAR(255));
select * from OLYMPICS_HISTORY;
select * from OLYMPICS_HISTORY_NOC_REGIONS;




```
[cloudera@quickstart ~]$ sqoop import-all-tables --connect jdbc:mysql://olympicdata.cadxlhohchda.ap-south-1.rds.amazonaws.com:3306/olympicdata --username admin --password olympia1 --hive-import --fields-terminat
ed-by ',' -m 1;
```

sqoop import-all-tables --connect jdbc:mysql://olympicdata.cadxlhohchda.ap-south-1.rds.amazonaws.com:3306/olympicdata --username admin --password olympia1 --hive-import --fields-terminated-by ',' -m 1;

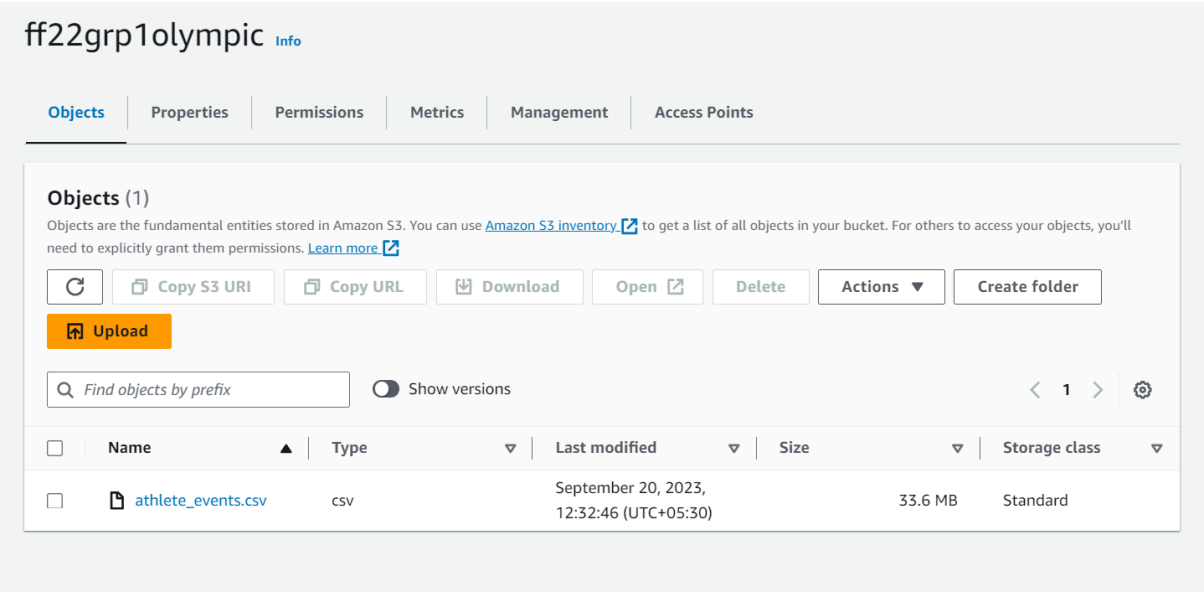
```
[cloudera@quickstart ~]$ sqoop import-all-tables --connect jdbc:mysql://localhost:3306/olympicdb --username root --password cloudera --hive-import --fields-terminated-by ',' -m 1;
```

sqoop import-all-tables --connect jdbc:mysql://localhost:3306/olympicdb --username root --password cloudera –hive-import –fields-terminated-by ‘,’ -m 1;

```
cloudera@quickstart:~
23/09/20 04:43:37 INFO mapreduce.Job:  map 0% reduce 0%
23/09/20 04:43:57 INFO mapreduce.Job:  map 100% reduce 0%
23/09/20 04:43:57 INFO mapreduce.Job: Job job_1695206321373_0001 completed successfully
23/09/20 04:43:57 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=141598
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=87
                HDFS: Number of bytes written=36998896
                HDFS: Number of read operations=4
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=16390
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=16390
                Total vcore-seconds taken by all map tasks=16390
                Total megabyte-seconds taken by all map tasks=16783360
        Map-Reduce Framework
                Map input records=271116
                Map output records=271116
                Input split bytes=87
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=274
                CPU time spent (ms)=5840
                Physical memory (bytes) snapshot=126324736
                Virtual memory (bytes) snapshot=1507131392
                Total committed heap usage (bytes)=60882944
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=36998896
23/09/20 04:43:57 INFO mapreduce.ImportJobBase: Transferred 35.2849 MB in 45.7625 seconds (789.5495 KB/sec)
23/09/20 04:43:57 INFO mapreduce.ImportJobBase: Retrieved 271116 records.
```

Pipeline 2- Amazon S3 Bucket to SnowFlake data warehouse.

Ingestion Tool used: Copy into




```
JAYJATHAR#OLYMPICDB@OLYMPICS.PUBLIC> create or replace file format olympics.public.
                                     my_csv_format
                                     type = csv
                                     field_delimiter = ','
                                     skip_header = 1
                                     null_if = ('NULL', 'null')
                                     empty_field_as_null = true;
+-----------------------------------------------+
| status                                        |
|-----------------------------------------------|
| File format MY_CSV_FORMAT successfully created. |
+-----------------------------------------------+
```

```
JAYJATHAR#OLYMPICDB@OLYMPICS.PUBLIC>create or replace storage INTEGRATION olympic_i
                                     nt
                                     type = EXTERNAL_STAGE storage_provider = s3
                                     enabled = true
                                     storage_aws_role_arn = 'arn:aws:iam::5593127351
                                     65:role/olympic_snow_import'
                                     storage_allowed_locations = ('s3://ff22grp1olym
                                     pic/');
+-----------------------------------------------+
| status                                        |
|-----------------------------------------------|
| Integration OLYMPIC_INT successfully created. |
+-----------------------------------------------+
```

```
JAYJATHAR#OLYMPICDB@OLYMPICS.PUBLIC>create stage olympic_stage
                                     storage_integration = olympic_int
                                     URL = 's3://ff22grp1olympic/athlete_events.csv'
                                     file_format = my_csv_format;
+-----------------------------------------------+
| status                                        |
|-----------------------------------------------|
| Stage area OLYMPIC_STAGE successfully created. |
+-----------------------------------------------+
```

Permissions | Trust relationships | Tags | Access Advisor | Revoke sessions

Trusted entities

Entities that can assume this role under specified conditions.

Edit trust policy

```
1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Effect": "Allow",
6              "Principal": {
7                  "AWS": "arn:aws:iam::559312735165:role/olympic_snow_import"
8              },
9              "Action": "sts:AssumeRole",
10             "Condition": {
11                 "StringEquals": {
12                     "sts:ExternalId": "ML16234_SFCRole=2_wYVfrp1kPMkoEa6Ny3SkGfN8f5I= "
13                 }
14             }
15         }
16     ]
17 }
```

```
DAYJATHAR@OLYMPICDB@OLYMPICS.PUBLIC>copy into OLYMPICS_HISTORY from @olympic_stage;
+-------------------------------------+--------+-------------+-------------+-----------+-----------+-------------+----------------+-----------------+-----
| file                                | status | rows_parsed | rows_loaded | error_limit | errors_seen | first_error | first_error_line | first_error_character | firs
|-------------------------------------+--------+-------------+-------------+-----------+-----------+-------------+----------------+-----------------+-----
| s3://ff22grplolympic/athlete_events.csv | LOADED |      271116 |      271116 |         1 |         0 | NULL        |           NULL |                NULL | NULL
+-------------------------------------+--------+-------------+-------------+-----------+-----------+-------------+----------------+-----------------+-----
1 Row(s) produced. Time Elapsed: 5.279s
DAYJATHAR@OLYMPICDB@OLYMPICS.PUBLIC>select count(age) from OLYMPICS_HISTORY;
+------------+
| COUNT(AGE) |
|------------|
|     261642 |
+------------+
```

Pipeline 3- Azure Blob Container to SnowFlake data warehouse.

Ingestion Tool used: Snow Pipe

Microsoft Azure | Search resources, services, and docs (G+/)

Home > Storage accounts >

Create a storage account

Basics | Advanced | Networking | Data protection | Encryption | Tags | Review

Project details

Select the subscription in which to create the new storage account. Choose a new or existing resource group to organize and manage your storage account together with other resources.

Subscription * Free Trial

Resource group * nithin_resource_group

Create new

A resource group is a container that holds related resources for an Azure solution.

Name * olympicGroup

OK | Cancel

Instance details

Storage account name *

Region *

Review | < Previous | Next : Advanced > | Give feedback

Instance details

Storage account name * ff22group1olympia

Region * (Asia Pacific) Central India

Deploy to an edge zone

Performance *

- (●) **Standard:** Recommended for most scenarios (general-purpose v2 account)
- ( ) **Premium:** Recommended for scenarios that require low latency.

Redundancy * Geo-redundant storage (GRS)

[x] Make read access to data available in the event of regional unavailability.

Home >

ff22group1olympia_1695190717337 | Overview

Deployment

Search

- Overview
- Inputs
- Outputs
- Template

Delete | Cancel | Redeploy | Download | Refresh

Your deployment is complete

Deployment name: ff22group1olympia_1695190717337
Subscription: Free Trial
Resource group: olympicGroup

Start time: 9/20/2023, 11:48:39 AM
Correlation ID: 103ac7f9-e330-4980-be3c-25af4b5433db

Deployment details

Next steps

Go to resource

Give feedback

Tell us about your experience with deployment

Cost Management
Get notified to stay within your budget and prevent unexpected charges on your bill.
Set up cost alerts >

Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >

Free Microsoft tutorials
Start learning today >

Work with an expert
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.
Find an Azure expert >

Dashboard | Futurense | Roundcube Webmail :: Inbox | Upload blob - Microsoft Azure

portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscriptions%2F0d19ff6c-5ba7-457e-868a-f65d34f9dc5f...

Microsoft Azure | Upgrade | Search resources, services, and docs (G+/) | nithinpogalla123@gmai... DEFAULT DIRECTORY (NITHINPO...

Home > Storage accounts > ff22group1olympia | Containers >

olympicdata

Container

Search

- Overview
- Diagnose and solve problems
- Access Control (IAM)

Settings

- Shared access tokens
- Access policy
- Properties
- Metadata

Upload | Change access level | Refresh | Delete | Change tier | Acquire lease

Authentication method: Access key (Switch to Azure AD User Account)
Location: olympicdata

Search blobs by prefix (case-sensitive)

Add filter

Name	Modified	Access tier	Ar
No results			

Upload blob

1 file(s) selected: noc_regions.csv

Drag and drop files here or Browse for files

Overwrite if files already exist

Advanced

Blob type
Block blob

Upload .vhd files as page blobs (recommended)

Block size
4 MiB

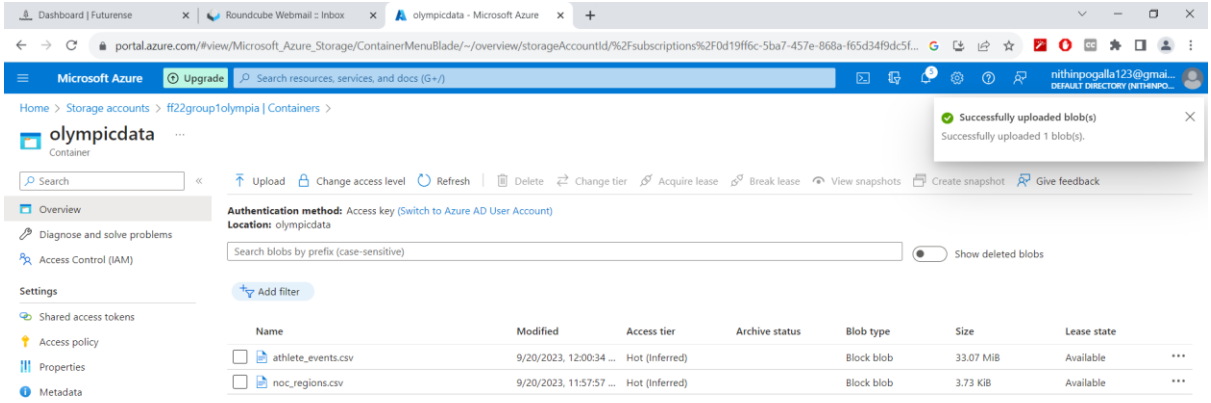
Access tier
Hot (Inferred)

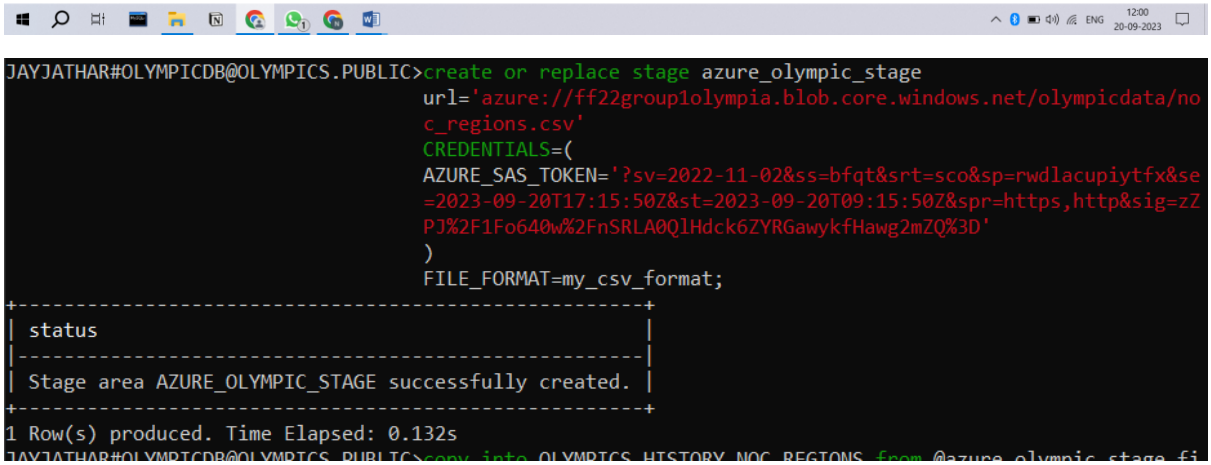
Upload to folder

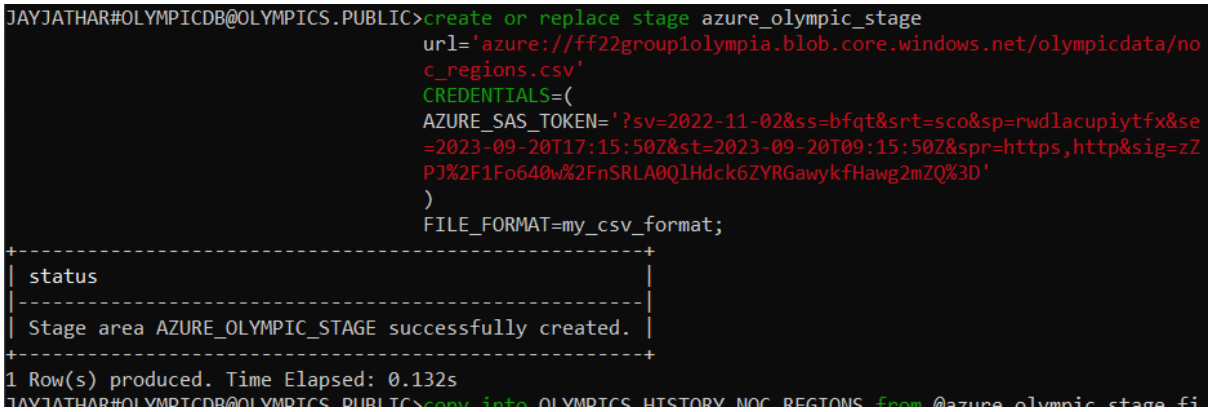
Blob index tags

Key | Value

ENG 11:57 20-09-2023







Step 2:- Transformation or Analysis of data in HIVE, HBASE, SNOWFLAKE according to the client requirements.

Analysis 1 – Snowfake

-- 1. How many Olympics games have been held?

```
SELECT COUNT(DISTINCT games) AS Total_Games FROM olympics_history;
```

-- 2. List down all Olympics games held so far.

```
SELECT DISTINCT games FROM olympics_history;
```

-- 3. Mention the total number of nations who participated in each Olympics game?

```
SELECT games, COUNT(DISTINCT n.region) AS Total_Nations
FROM olympics_history o
JOIN olympics_history_noc_regions n ON o.noc = n.noc
GROUP BY games;
```

Results Chart

	GAMES	...	TOTAL_NATIONS
1	1992 Summer		167
2	2012 Summer		203
3	1920 Summer		29
4	1900 Summer		31
5	1988 Winter		56
6	1992 Winter		64
7	1994 Winter		67
8	1932 Summer		47
9	2002 Winter		76
10	1952 Summer		66
11	2000 Summer		198

-- 4. Which year saw the highest and lowest number of countries participating in Olympics?

```
WITH cte AS (
    SELECT year, COUNT(DISTINCT n.region) AS Countries
    FROM olympics_history o
    JOIN olympics_history_noc_regions n ON o.noc = n.noc
    GROUP BY year
)
SELECT year, countries
FROM (
    SELECT year, countries, DENSE_RANK() OVER (ORDER BY countries) AS l, DENSE_RANK() OVER (ORDER BY countries DESC) AS h
    FROM cte
) WHERE l = 1 OR h = 1;
```

Results Chart

	...	YEAR
1		2,016
2		1,896

-- 5. Which nation has participated in all of the Olympic games?

```
WITH cte AS (
```

```
    SELECT n.region AS nations, COUNT(DISTINCT games) AS game_count
    FROM olympics_history o
    JOIN olympics_history_noc_regions n ON o.noc = n.noc
    GROUP BY nations
)
SELECT nations FROM cte WHERE game_count = (SELECT COUNT(DISTINCT games) FROM olympics_history);
```

-- 6. Identify the sport which was played in all summer Olympics.

```
WITH cte AS (
    SELECT sport, COUNT(DISTINCT games) AS game_count
    FROM olympics_history
    WHERE season = 'Summer'
    GROUP BY sport
)
SELECT sport FROM cte WHERE game_count = (SELECT COUNT(DISTINCT games) FROM olympics_history WHERE season = 'Summer');
```

-- 7. Which sports were just played only once in the Olympics?

```
SELECT sport, COUNT(DISTINCT games) AS game_count
FROM olympics_history
```

```
GROUP BY sport
HAVING game_count = 1;
```

-- 8. Fetch the total number of sports played in each Olympic game.

```
SELECT games, COUNT(DISTINCT sport) AS Total_Sports FROM olympics_history GROUP BY games;
```

-- 9. Fetch details of the oldest athletes to win a gold medal.

```
SELECT name, age, games, sport, medal
FROM olympics_history
WHERE medal = 'Gold' AND age = (SELECT MAX(age) FROM olympics_history WHERE medal = 'Gold');
```

-- 10. Find the ratio of male and female athletes participated in all Olympic games.

```
WITH cte AS (
    SELECT DISTINCT name, sex FROM olympics_history
)
SELECT
    SUM(CASE WHEN sex = 'M' THEN 1 ELSE 0 END) / (SELECT COUNT(DISTINCT name) FROM olympics_history) * 100 AS male_ratio,
```

```
    SUM(CASE WHEN sex = 'F' THEN 1 ELSE 0 END) / (SELECT
COUNT(DISTINCT name) FROM olympics_history) * 100 AS
female_ratio
FROM cte;
```

-- 11. Fetch the top 5 athletes who have won the most gold medals.

```
SELECT name, COUNT(medal) AS medalsWon
FROM olympics_history
WHERE medal = 'Gold'
GROUP BY name
ORDER BY medalsWon DESC
LIMIT 5;
```

-- 12. Fetch the top 5 athletes who have won the most medals (gold/silver/bronze).

```
SELECT name, COUNT(medal) AS medalsWon
FROM olympics_history
GROUP BY name
ORDER BY medalsWon DESC
LIMIT 5;
```

-- 13. Fetch the top 5 most successful countries in Olympics. Success is defined by the number of medals won.

```
SELECT nr.region, COUNT(oh.medal) AS medalsWon
FROM olympics_history oh
JOIN olympics_history_noc_regions nr ON oh.noc = nr.noc
GROUP BY nr.region
ORDER BY medalsWon DESC
LIMIT 5;
```

-- 14. <u>List down the total gold, silver, and bronze medals won by each country.</u>

```
WITH MedalCounts AS (
    SELECT
        noc,
        SUM(CASE WHEN medal = 'Gold' THEN 1 ELSE 0 END) AS GoldCount,
        SUM(CASE WHEN medal = 'Silver' THEN 1 ELSE 0 END) AS SilverCount,
        SUM(CASE WHEN medal = 'Bronze' THEN 1 ELSE 0 END) AS BronzeCount
    FROM olympics_history
    GROUP BY noc
)
SELECT
    nr.region,
```

```
    SUM(mc.GoldCount) AS GoldWon,
    SUM(mc.SilverCount) AS SilverWon,
    SUM(mc.BronzeCount) AS BronzeWon
FROM MedalCounts mc
JOIN olympics_history_noc_regions nr ON mc.noc = nr.noc
GROUP BY nr.region
ORDER BY GoldWon DESC, SilverWon DESC, BronzeWon DESC;
```

-- 15. List down the total gold, silver, and bronze medals won by each country corresponding to each Olympic game.

```
SELECT n.region, games, SUM(CASE WHEN medal='Gold' THEN 1 ELSE 0 END) AS gold,
       SUM(CASE WHEN medal='Silver' THEN 1 ELSE 0 END) AS silver,
       SUM(CASE WHEN medal='Bronze' THEN 1 ELSE 0 END) AS bronze
FROM olympics_history o
JOIN olympics_history_noc_regions n ON o.noc = n.noc
GROUP BY n.region, games;
```

-- 16. Identify which country won the most gold, most silver, and most bronze medals in each Olympic game.

```
WITH gold_cte AS (
    SELECT noc, games, region,
```

```
    SUM(CASE WHEN medal='Gold' THEN 1 ELSE 0 END) AS
gold_medals,
    RANK() OVER(PARTITION BY games ORDER BY gold_medals
DESC) AS gold_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
),
silver_cte AS (
    SELECT noc, games, region,
    SUM(CASE WHEN medal='Silver' THEN 1 ELSE 0 END) AS
silver_medals,
    RANK() OVER(PARTITION BY games ORDER BY silver_medals
DESC) AS silver_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
),
bronze_cte AS (
    SELECT noc, games, region,
    SUM(CASE WHEN medal='Bronze' THEN 1 ELSE 0 END) AS
bronze_medals,
    RANK() OVER(PARTITION BY games ORDER BY
bronze_medals DESC) AS bronze_rank
```

```
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
)
SELECT games, g.region AS g_region, s.region AS s_region,
b.region AS b_region
FROM gold_cte g
JOIN silver_cte s USING(games)
JOIN bronze_cte b USING(games)
WHERE gold_rank = 1 AND silver_rank = 1 AND bronze_rank =
1
ORDER BY games;
```

-- 17. Identify which country won the most gold, most silver, most bronze medals, and the most medals in each Olympic game.

```
WITH gold_cte AS (
    SELECT noc, games, region,
    SUM(CASE WHEN medal='Gold' THEN 1 ELSE 0 END) AS
gold_medals,
    RANK() OVER(PARTITION BY games ORDER BY gold_medals
DESC) AS gold_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
```

```
    GROUP BY noc, games, region
),
silver_cte AS (
    SELECT noc, games, region,
    SUM(CASE WHEN medal='Silver' THEN 1 ELSE 0 END) AS
silver_medals,
    RANK() OVER(PARTITION BY games ORDER BY silver_medals
DESC) AS silver_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
),
bronze_cte AS (
    SELECT noc, games, region,
    SUM(CASE WHEN medal='Bronze' THEN 1 ELSE 0 END) AS
bronze_medals,
    RANK() OVER(PARTITION BY games ORDER BY
bronze_medals DESC) AS bronze_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
),
total_cte AS (
```

```
    SELECT noc, games, region,
    COUNT(medal) AS total_medals,
    RANK() OVER(PARTITION BY games ORDER BY total_medals
DESC) AS total_rank
    FROM olympics_history
    JOIN olympics_history_noc_regions USING(noc)
    GROUP BY noc, games, region
)
SELECT games, g.region AS g_region, s.region AS s_region,
b.region AS b_region, t.region AS total_region
FROM gold_cte g
JOIN silver_cte s USING(games)
JOIN bronze_cte b USING(games)
JOIN total_cte t USING(games)
WHERE gold_rank = 1 AND silver_rank = 1 AND bronze_rank =
1 AND total_rank = 1
ORDER BY games;
```

-- 18. Which countries have never won a gold medal but have won silver/bronze medals?

```
WITH cte AS (
    SELECT noc,
    SUM(CASE WHEN medal='Gold' THEN 1 ELSE 0 END) AS
gold_medals,
```

```
    SUM(CASE WHEN medal='Bronze' OR medal='Silver' THEN 1
ELSE 0 END) AS other_medals
    FROM olympics_history
    GROUP BY noc
)
SELECT region FROM cte
JOIN olympics_history_noc_regions USING(noc)
WHERE gold_medals = 0 AND other_medals <> 0;
```

-- 19. <u>In which Sport/event, India has won the highest number of medals.</u>

```
SELECT sport, COUNT(medal) AS medals
FROM olympics_history
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India'
GROUP BY sport
ORDER BY medals DESC
LIMIT 1;
```

```
SELECT event, COUNT(medal) AS medals
FROM olympics_history
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India'
```

```
GROUP BY event
ORDER BY medals DESC
LIMIT 1;
```

-- 20. Break down all Olympic games where India won a medal for Hockey and how many medals in each Olympic game.

```
SELECT games, COUNT(medal) AS medals
FROM olympics_history
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India' AND sport LIKE '%Hockey%'
GROUP BY games
HAVING medals <> 0
ORDER BY medals DESC;
```

Analysis 2 – Hive

-- 1. How many Olympics games have been held?

```
SELECT COUNT(DISTINCT games) AS total_olympic_games
FROM olympics_history;
```

```
Total MapReduce CPU Time Spent: 9 seconds 430 msec
OK
51
Time taken: 70.271 seconds, Fetched: 1 row(s)
```

-- 2. List down all Olympics games held so far.

SELECT DISTINCT games AS total_olympic_games FROM olympics_history;

```
2000 Summer
2002 Winter
2004 Summer
2006 Winter
2008 Summer
2010 Winter
2012 Summer
2014 Winter
2016 Summer
Time taken: 59.713 seconds, Fetched: 51 row(s)
```

-- 3. <u>Mention the total number of nations who participated in each Olympics game?</u>

SELECT games, COUNT(DISTINCT noc) FROM OLYMPICS_HISTORY GROUP BY games;

-- 4. <u>Which year saw the highest and lowest number of countries participating in Olympics?</u>

```
SELECT year, nation_part
FROM (
    SELECT year, COUNT(DISTINCT noc) AS nation_part
    FROM olympics_history
    GROUP BY year
    ORDER BY nation_part DESC
    LIMIT 1
) highest_participation
UNION ALL
```

```
SELECT year, nation_part
FROM (
    SELECT year, COUNT(DISTINCT noc) AS nation_part
    FROM olympics_history
    GROUP BY year
    ORDER BY nation_part ASC
    LIMIT 1
) lowest_participation;
```

-- 5. Which nation has participated in all of the Olympic games?

```
SELECT noc
FROM (SELECT COUNT(DISTINCT games) AS cnt_games FROM olympics_history) AS cte, (SELECT noc, COUNT(DISTINCT games) AS cnt
FROM olympics_history
GROUP BY noc
ORDER BY cnt DESC) AS cte2
WHERE cnt_games = cnt;
```

-- 6. Identify the sport which was played in all summer Olympics.

```
SELECT sport
```

```
FROM (SELECT COUNT(DISTINCT games) AS cnt_games FROM olympics_history WHERE season='Summer') AS cte, (SELECT sport, COUNT(DISTINCT games) AS cnt
FROM olympics_history
WHERE season='Summer'
GROUP BY sport
ORDER BY cnt DESC) AS cte2
WHERE cnt_games = cnt;
```

-- 7. Which Sports were just played only once in the Olympics?

```
SELECT sport
FROM (
    SELECT sport, COUNT(DISTINCT games) AS total_games
    FROM OLYMPICS_HISTORY
    GROUP BY sport
    HAVING total_games = 1
) AS tt;
```

-- 8. Fetch the total number of sports played in each Olympic game.

```
SELECT games, COUNT(DISTINCT sport) AS total_sports_played
FROM olympics_history_optimized
GROUP BY games;
```

-- 9. Fetch details of the oldest athletes to win a gold medal.

```
SELECT name, age, games, sport, event
FROM olympics_history_optimized
WHERE medal = 'Gold'
ORDER BY age ASC
LIMIT 1;
```

-- 10. Find the Ratio of male and female athletes participated in all Olympic games.

```
WITH athlete_counts AS (
    SELECT games, sex, COUNT(DISTINCT name) AS athlete_count
    FROM olympics_history_optimized
    GROUP BY games, sex
),
total_athletes AS (
    SELECT games, SUM(athlete_count) AS total_count
    FROM athlete_counts
    GROUP BY games
)
SELECT t.games,
        m.athlete_count AS male_athletes,
```

```
    f.athlete_count AS female_athletes,
    ROUND(m.athlete_count / f.athlete_count, 2) AS gender_ratio
FROM total_athletes t
JOIN athlete_counts m ON t.games = m.games AND m.sex = 'M'
JOIN athlete_counts f ON t.games = f.games AND f.sex = 'F';
```

Partitioning table for optimization

```
CREATE TABLE olympics_history_optimized
(
    id          INT,
    name        STRING,
    sex         STRING,
    age         INT,
    height      STRING,
    weight      STRING,
    team        STRING,
    noc         STRING,
    games       STRING,
    year        INT,
    season      STRING,
```

```
    city        STRING,
    sport       STRING,
    event       STRING
)
PARTITIONED BY (medal STRING)
CLUSTERED BY (noc)
INTO 5 BUCKETS row format delimited fields terminated by ','
lines terminated by '\n' stored as orc;

INSERT OVERWRITE TABLE olympics_history_optimized
PARTITION(medal) SELECT id, name, sex, age, height, weight,
team, noc, games, year, season, city, sport, event, medal FROM
OLYMPICS_HISTORY;
```

-- 11. Fetch the top 5 athletes who have won the most gold medals.

```
SELECT name, COUNT(*) AS gold_medals_count
FROM olympics_history_optimized
WHERE medal = 'Gold'
GROUP BY name
ORDER BY gold_medals_count DESC
LIMIT 5;
```

```
Total MapReduce CPU Time Spent: 31 seconds 430 msec
OK
Michael Fred Phelps II  23
"Raymond Clarence ""Ray"" Ewry" 10
Larysa Semenivna Latynina (Diriy-)      9
Paavo Johannes Nurmi    9
Mark Andrew Spitz       9
Time taken: 188.765 seconds, Fetched: 5 row(s)
```

-- 12. Fetch the top 5 athletes who have won the most medals (gold/silver/bronze).

```
SELECT name, COUNT(medal) AS medalsWon

FROM olympics_history_optimized

GROUP BY name

ORDER BY medalsWon DESC

LIMIT 5;
```

-- 13. Fetch the top 5 most successful countries in Olympics. Success is defined by the number of medals won.

```
SELECT nr.region, COUNT(oh.medal) AS medalsWon

FROM olympics_history_optimized oh

JOIN olympics_history_noc_regions nr ON oh.noc = nr.noc

GROUP BY nr.region

ORDER BY medalsWon DESC

LIMIT 5;
```

-- 14. List down the total number of gold, silver, and bronze medals won by each country.

```
SELECT nr.region,
       SUM(mc.GoldCount) AS GoldWon,
       SUM(mc.SilverCount) AS SilverWon,
       SUM(mc.BronzeCount) AS BronzeWon
FROM (
   SELECT noc,
          SUM(CASE WHEN medal = 'Gold' THEN 1 ELSE 0 END)
AS GoldCount,
          SUM(CASE WHEN medal = 'Silver' THEN 1 ELSE 0 END)
AS SilverCount,
          SUM(CASE WHEN medal = 'Bronze' THEN 1 ELSE 0 END)
AS BronzeCount
   FROM olympics_history_optimized
   GROUP BY noc
) mc
JOIN olympics_history_noc_regions nr ON mc.noc = nr.noc
GROUP BY nr.region
ORDER BY GoldWon DESC, SilverWon DESC, BronzeWon DESC;
```

-- 15. List down the total number of gold, silver, and bronze medals won by each country corresponding to each Olympic game.

```
SELECT games, n.region,
      SUM(CASE WHEN medal = 'Gold' THEN 1 ELSE 0 END) AS gold,
      SUM(CASE WHEN medal = 'Silver' THEN 1 ELSE 0 END) AS silver,
      SUM(CASE WHEN medal = 'Bronze' THEN 1 ELSE 0 END) AS bronze
FROM olympics_history_optimized o
JOIN olympics_history_noc_regions n ON o.noc = n.noc
GROUP BY games, n.region;
```

-- 16. Identify which country won the most gold, most silver, and most bronze medals in each Olympic game.

```
SELECT games,
      MAX(CASE WHEN medal = 'Gold' THEN region END) AS most_gold_country,
      MAX(CASE WHEN medal = 'Silver' THEN region END) AS most_silver_country,
      MAX(CASE WHEN medal = 'Bronze' THEN region END) AS most_bronze_country
FROM (
   SELECT games, region, medal,
          RANK() OVER (PARTITION BY games, medal ORDER BY COUNT(*) DESC) AS r
```

```
    FROM olympics_history_optimized oh
    JOIN olympics_history_noc_regions nr ON oh.noc = nr.noc
    GROUP BY games, region, medal
) ranked
WHERE r = 1
GROUP BY games;
```

-- 17. Identify which country won the most gold, most silver, most bronze medals, and the most medals in each Olympic game.

```
WITH medal_counts AS (
    SELECT games, region, medal, COUNT(*) AS medal_count
    FROM olympics_history_optimized oh
    JOIN olympics_history_noc_regions nr ON oh.noc = nr.noc
    GROUP BY games, region, medal
),
ranked_medals AS (
    SELECT games, region, medal,
           RANK() OVER (PARTITION BY games, medal ORDER BY medal_count DESC) AS r
    FROM medal_counts
)
SELECT games,
```

```
    MAX(CASE WHEN medal = 'Gold' THEN region END) AS most_gold_country,
    MAX(CASE WHEN medal = 'Silver' THEN region END) AS most_silver_country,
    MAX(CASE WHEN medal = 'Bronze' THEN region END) AS most_bronze_country,
    MAX(CASE WHEN r = 1 THEN region END) AS most_medal_country
FROM ranked_medals
GROUP BY games;
```

Analysis 3 – Hbase

First, in HBase shell, create 'olympics_history', 'person_details', 'games_details'.

Then, in Hive shell, create an external table olympics_history_integrated and import data from HBase.

In HBase shell:

Create 'olympics_history', 'person_details', 'games_details' tables.

In Hive shell:

```
CREATE EXTERNAL TABLE olympics_history_integrated
```

```
(
    num INT,
    id INT,
    name STRING,
    sex STRING,
    age INT,
    height INT,
    weight INT,
    team STRING,
    noc STRING,
    games STRING,
    year INT,
    season STRING,
    city STRING,
    sport STRING,
    event STRING,
    medal STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED BY
'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES (
```

```
  "hbase.columns.mapping" =
":key,person_details:id,person_details:name,person_details:sex,p
erson_details:age,person_details:height,person_details:weight,pe
rson_details:team,person_details:noc,games_details:games,gam
es_details:year,games_details:season,games_details:city,games_
details:sport,games_details:event,games_details:medal"
)
TBLPROPERTIES (
  "hbase.table.name" = "olympics_history"
);
```

Import data from HBase into the external table.

Make sure the HBase table has the necessary data populated. Now you can run queries on the olympics_history_integrated table in Hive.

-- 18. <u>Which countries have never won a gold medal but have won silver/bronze medals?</u>

```
WITH cte AS (
    SELECT noc,
    SUM(CASE WHEN medal='Gold' THEN 1 ELSE 0 END) AS
gold_medals,
    SUM(CASE WHEN medal='Bronze' OR medal='Silver' THEN 1
ELSE 0 END) AS other_medals
    FROM olympics_history_integrated
```

```
    GROUP BY noc
)
SELECT region FROM cte
JOIN olympics_history_noc_regions USING(noc)
WHERE gold_medals = 0 AND other_medals <> 0;
```

-- 19. <u>In which Sport/event, India has won the highest number of medals.</u>

```
SELECT sport, COUNT(medal) AS medals
FROM olympics_history_integrated
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India'
GROUP BY sport
ORDER BY medals DESC
LIMIT 1;

SELECT event, COUNT(medal) AS medals
FROM olympics_history_integrated
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India'
GROUP BY event
ORDER BY medals DESC
LIMIT 1;
```

-- 20. Break down all Olympic games where India won a medal for Hockey and how many medals in each Olympic game.

```
SELECT games, COUNT(medal) AS medals
FROM olympics_history_integrated
JOIN olympics_history_noc_regions USING(noc)
WHERE region='India' AND sport LIKE '%Hockey%'
GROUP BY games
HAVING medals <> 0
ORDER BY medals DESC;
```

Step 3:- Loading Results to client database or snowflake data warehouse.

In Snowflake, we create result table in data warehouse

```
--How many olympics games have been held?
create table result1 as select count(distinct games) as Total_Games from olympics_history;

--List down all Olympics games held so far.
create table result2 as select distinct games from olympics_history;
```

In Hive/Hbase, We create external table and then export them to client database

--List down all Olympics games held so far.

create external table op_ol_2(distinct_olympics_game string) row format delimited fields terminated by ',' location '/user/cloudera/olympic/op2';

INSERT OVERWRITE TABLE op_ol_2 SELECT DISTINCT games AS total_olympic_games FROM olympics_history;

--Mention the total no of nations who participated in each olympics game?

Create external table op_ol_3(games string, total_participating_nations int) row format delimited fields terminated by ',' location '/user/cloudera/olympic/op3';

INSERT OVERWRITE TABLE op_ol_3 SELECT games, COUNT(DISTINCT noc) FROM OLYMPICS_HISTORY GROUP BY games;