DROP TABLE IF EXISTS OLYMPICS\_HISTORY;

CREATE TABLE IF NOT EXISTS OLYMPICS\_HISTORY

(

id INT,

name VARCHAR,

sex VARCHAR,

age VARCHAR,

height VARCHAR,

weight VARCHAR,

team VARCHAR,

noc VARCHAR,

games VARCHAR,

year INT,

season VARCHAR,

city VARCHAR,

sport VARCHAR,

event VARCHAR,

medal VARCHAR

);

DROP TABLE IF EXISTS OLYMPICS\_HISTORY\_NOC\_REGIONS;

CREATE TABLE IF NOT EXISTS OLYMPICS\_HISTORY\_NOC\_REGIONS

(

noc VARCHAR,

region VARCHAR,

notes VARCHAR

);

select \* from OLYMPICS\_HISTORY;

select \* from OLYMPICS\_HISTORY\_NOC\_REGIONS;

-- 1. HOW MANY OLYMPICS GAMES HAVE BEEN HELD?

select count(distinct games) as total\_olympic\_games

from olympics\_history;

-- 2. LIST DOWN ALL OLYMPICS GAMES HELD SO FAR. (DATA ISSUE AT 1956-"SUMMER"-"STOCKHOLM")

select distinct oh.year,oh.season,oh.city

from olympics\_history oh

order by year;

-- 3. MENTION THE TOTAL NO OF NATIONS WHO PARTICIPATED IN EACH OLYMPICS GAME?

with all\_countries as

(select games, nr.region

from olympics\_history oh

join olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

group by games, nr.region)

select games, count(1) as total\_countries

from all\_countries

group by games

order by games;

-- 4. WHICH YEAR SAW THE HIGHEST AND LOWEST NO OF COUNTRIES PARTICIPATING IN OLYMPICS

with all\_countries as

(select games, nr.region

from olympics\_history oh

join olympics\_history\_noc\_regions nr ON nr.noc=oh.noc

group by games, nr.region),

tot\_countries as

(select games, count(1) as total\_countries

from all\_countries

group by games)

select distinct

concat(first\_value(games) over(order by total\_countries)

, ' - '

, first\_value(total\_countries) over(order by total\_countries)) as Lowest\_Countries,

concat(first\_value(games) over(order by total\_countries desc)

, ' - '

, first\_value(total\_countries) over(order by total\_countries desc)) as Highest\_Countries

from tot\_countries

order by 1;

-- 5. WHICH NATION HAS PARTICIPATED IN ALL OF THE OLYMPIC GAMES

with tot\_games as

(select count(distinct games) as total\_games

from olympics\_history),

countries as

(select games, nr.region as country

from olympics\_history oh

join olympics\_history\_noc\_regions nr ON nr.noc=oh.noc

group by games, nr.region),

countries\_participated as

(select country, count(1) as total\_participated\_games

from countries

group by country)

select cp.\*

from countries\_participated cp

join tot\_games tg on tg.total\_games = cp.total\_participated\_games

order by 1;

-- 6. IDENTIFY THE SPORT WHICH WAS PLAYED IN ALL SUMMER OLYMPICS.

with t1 as

(select count(distinct games) as total\_games

from olympics\_history where season = 'Summer'),

t2 as

(select distinct games, sport

from olympics\_history where season = 'Summer'),

t3 as

(select sport, count(1) as no\_of\_games

from t2

group by sport)

select \*

from t3

join t1 on t1.total\_games = t3.no\_of\_games;

-- 7. WHICH SPORTS WERE JUST PLAYED ONLY ONCE IN THE OLYMPICS.

with t1 as

(select distinct games, sport

from olympics\_history),

t2 as

(select sport, count(1) as no\_of\_games

from t1

group by sport)

select t2.\*, t1.games

from t2

join t1 on t1.sport = t2.sport

where t2.no\_of\_games = 1

order by t1.sport;

-- 8. FETCH THE TOTAL NO OF SPORTS PLAYED IN EACH OLYMPIC GAMES.

with t1 as

(select distinct games, sport

from olympics\_history),

t2 as

(select games, count(1) as no\_of\_sports

from t1

group by games)

select \* from t2

order by no\_of\_sports desc;

-- 9. FETCH OLDEST ATHLETES TO WIN A GOLD MEDAL

with temp as

(select name,sex,cast(case when age = 'NA' then '0' else age end as int) as age

,team,games,city,sport, event, medal

from olympics\_history),

ranking as

(select \*, rank() over(order by age desc) as rnk

from temp

where medal='Gold')

select \*

from ranking

where rnk = 1;

-- 10. FIND THE RATIO OF MALE AND FEMALE ATHLETES PARTICIPATED IN ALL OLYMPIC GAMES.

with t1 as

(select sex, count(1) as cnt

from olympics\_history

group by sex),

t2 as

(select \*, row\_number() over(order by cnt) as rn

from t1),

min\_cnt as

(select cnt from t2 where rn = 1),

max\_cnt as

(select cnt from t2 where rn = 2)

select concat('1 : ', round(max\_cnt.cnt::decimal/min\_cnt.cnt, 2)) as ratio

from min\_cnt, max\_cnt;

-- 11. TOP 5 ATHLETES WHO HAVE WON THE MOST GOLD MEDALS.

with t1 as

(select name, team, count(1) as total\_gold\_medals

from olympics\_history

where medal = 'Gold'

group by name, team

order by total\_gold\_medals desc),

t2 as

(select \*, dense\_rank() over (order by total\_gold\_medals desc) as rnk

from t1)

select name, team, total\_gold\_medals

from t2

where rnk <= 5;

-- 12. TOP 5 ATHLETES WHO HAVE WON THE MOST MEDALS (GOLD/SILVER/BRONZE).

with t1 as

(select name, team, count(1) as total\_medals

from olympics\_history

where medal in ('Gold', 'Silver', 'Bronze')

group by name, team

order by total\_medals desc),

t2 as

(select \*, dense\_rank() over (order by total\_medals desc) as rnk

from t1)

select name, team, total\_medals

from t2

where rnk <= 5;

-- 13. TOP 5 MOST SUCCESSFUL COUNTRIES IN OLYMPICS. SUCCESS IS DEFINED BY NO OF MEDALS WON.

with t1 as

(select nr.region, count(1) as total\_medals

from olympics\_history oh

join olympics\_history\_noc\_regions nr on nr.noc = oh.noc

where medal <> 'NA'

group by nr.region

order by total\_medals desc),

t2 as

(select \*, dense\_rank() over(order by total\_medals desc) as rnk

from t1)

select \*

from t2

where rnk <= 5;

-- PIVOT

In Postgresql, we can use crosstab function to create pivot table.

crosstab function is part of a PostgreSQL extension called tablefunc.

To call the crosstab function, you must first enable the tablefunc extension by executing the following SQL command:

CREATE EXTENSION TABLEFUNC;

-- 14. LIST DOWN TOTAL GOLD, SILVER AND BROZE MEDALS WON BY EACH COUNTRY.

SELECT country

, coalesce(gold, 0) as gold

, coalesce(silver, 0) as silver

, coalesce(bronze, 0) as bronze

FROM CROSSTAB('SELECT nr.region as country

, medal

, count(1) as total\_medals

FROM olympics\_history oh

JOIN olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

where medal <> ''NA''

GROUP BY nr.region,medal

order BY nr.region,medal',

'values (''Bronze''), (''Gold''), (''Silver'')')

AS FINAL\_RESULT(country varchar, bronze bigint, gold bigint, silver bigint)

order by gold desc, silver desc, bronze desc;

-- PIVOT

In Postgresql, we can use crosstab function to create pivot table.

crosstab function is part of a PostgreSQL extension called tablefunc.

To call the crosstab function, you must first enable the tablefunc extension by executing the following SQL command:

CREATE EXTENSION TABLEFUNC;

-- 15. LIST DOWN TOTAL GOLD, SILVER AND BROZE MEDALS WON BY EACH COUNTRY CORRESPONDING TO EACH OLYMPIC GAMES.

SELECT substring(games,1,position(' - ' in games) - 1) as games

, substring(games,position(' - ' in games) + 3) as country

, coalesce(gold, 0) as gold

, coalesce(silver, 0) as silver

, coalesce(bronze, 0) as bronze

FROM CROSSTAB('SELECT concat(games, '' - '', nr.region) as games

, medal

, count(1) as total\_medals

FROM olympics\_history oh

JOIN olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

where medal <> ''NA''

GROUP BY games,nr.region,medal

order BY games,medal',

'values (''Bronze''), (''Gold''), (''Silver'')')

AS FINAL\_RESULT(games text, bronze bigint, gold bigint, silver bigint);

-- PIVOT

In Postgresql, we can use crosstab function to create pivot table.

crosstab function is part of a PostgreSQL extension called tablefunc.

To call the crosstab function, you must first enable the tablefunc extension by executing the following SQL command:

CREATE EXTENSION TABLEFUNC;

-- 16. IDENTIFY WHICH COUNTRY WON THE MOST GOLD, MOST SILVER AND MOST BRONZE MEDALS IN EACH OLYMPIC GAMES.

WITH temp as

(SELECT substring(games, 1, position(' - ' in games) - 1) as games

, substring(games, position(' - ' in games) + 3) as country

, coalesce(gold, 0) as gold

, coalesce(silver, 0) as silver

, coalesce(bronze, 0) as bronze

FROM CROSSTAB('SELECT concat(games, '' - '', nr.region) as games

, medal

, count(1) as total\_medals

FROM olympics\_history oh

JOIN olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

where medal <> ''NA''

GROUP BY games,nr.region,medal

order BY games,medal',

'values (''Bronze''), (''Gold''), (''Silver'')')

AS FINAL\_RESULT(games text, bronze bigint, gold bigint, silver bigint))

select distinct games

, concat(first\_value(country) over(partition by games order by gold desc)

, ' - '

, first\_value(gold) over(partition by games order by gold desc)) as Max\_Gold

, concat(first\_value(country) over(partition by games order by silver desc)

, ' - '

, first\_value(silver) over(partition by games order by silver desc)) as Max\_Silver

, concat(first\_value(country) over(partition by games order by bronze desc)

, ' - '

, first\_value(bronze) over(partition by games order by bronze desc)) as Max\_Bronze

from temp

order by games;

-- PIVOT

In Postgresql, we can use crosstab function to create pivot table.

crosstab function is part of a PostgreSQL extension called tablefunc.

To call the crosstab function, you must first enable the tablefunc extension by executing the following SQL command:

CREATE EXTENSION TABLEFUNC;

-- 17. IDENTIFY WHICH COUNTRY WON THE MOST GOLD, MOST SILVER, MOST BRONZE MEDALS AND THE MOST MEDALS IN EACH OLYMPIC GAMES.

with temp as

(SELECT substring(games, 1, position(' - ' in games) - 1) as games

, substring(games, position(' - ' in games) + 3) as country

, coalesce(gold, 0) as gold

, coalesce(silver, 0) as silver

, coalesce(bronze, 0) as bronze

FROM CROSSTAB('SELECT concat(games, '' - '', nr.region) as games

, medal

, count(1) as total\_medals

FROM olympics\_history oh

JOIN olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

where medal <> ''NA''

GROUP BY games,nr.region,medal

order BY games,medal',

'values (''Bronze''), (''Gold''), (''Silver'')')

AS FINAL\_RESULT(games text, bronze bigint, gold bigint, silver bigint)),

tot\_medals as

(SELECT games, nr.region as country, count(1) as total\_medals

FROM olympics\_history oh

JOIN olympics\_history\_noc\_regions nr ON nr.noc = oh.noc

where medal <> 'NA'

GROUP BY games,nr.region order BY 1, 2)

select distinct t.games

, concat(first\_value(t.country) over(partition by t.games order by gold desc)

, ' - '

, first\_value(t.gold) over(partition by t.games order by gold desc)) as Max\_Gold

, concat(first\_value(t.country) over(partition by t.games order by silver desc)

, ' - '

, first\_value(t.silver) over(partition by t.games order by silver desc)) as Max\_Silver

, concat(first\_value(t.country) over(partition by t.games order by bronze desc)

, ' - '

, first\_value(t.bronze) over(partition by t.games order by bronze desc)) as Max\_Bronze

, concat(first\_value(tm.country) over (partition by tm.games order by total\_medals desc nulls last)

, ' - '

, first\_value(tm.total\_medals) over(partition by tm.games order by total\_medals desc nulls last)) as Max\_Medals

from temp t

join tot\_medals tm on tm.games = t.games and tm.country = t.country

order by games;

-- PIVOT

In Postgresql, we can use crosstab function to create pivot table.

crosstab function is part of a PostgreSQL extension called tablefunc.

To call the crosstab function, you must first enable the tablefunc extension by executing the following SQL command:

CREATE EXTENSION TABLEFUNC;

-- 18. WHICH COUNTRIES HAVE NEVER WON GOLD MEDAL BUT HAVE WON SILVER/BRONZE MEDALS?

select \* from (

SELECT country, coalesce(gold,0) as gold, coalesce(silver,0) as silver, coalesce(bronze,0) as bronze

FROM CROSSTAB('SELECT nr.region as country

, medal, count(1) as total\_medals

FROM OLYMPICS\_HISTORY oh

JOIN OLYMPICS\_HISTORY\_NOC\_REGIONS nr ON nr.noc=oh.noc

where medal <> ''NA''

GROUP BY nr.region,medal order BY nr.region,medal',

'values (''Bronze''), (''Gold''), (''Silver'')')

AS FINAL\_RESULT(country varchar,

bronze bigint, gold bigint, silver bigint)) x

where gold = 0 and (silver > 0 or bronze > 0)

order by gold desc nulls last, silver desc nulls last, bronze desc nulls last;

-- 19. IN WHICH SPORT/EVENT, INDIA HAS WON HIGHEST MEDALS.

with t1 as

(select sport, count(1) as total\_medals

from olympics\_history

where medal <> 'NA'

and team = 'India'

group by sport

order by total\_medals desc),

t2 as

(select \*, rank() over(order by total\_medals desc) as rnk

from t1)

select sport, total\_medals

from t2

where rnk = 1;

-- 20. BREAK DOWN ALL OLYMPIC GAMES WHERE INDIA WON MEDAL FOR HOCKEY AND HOW MANY MEDALS IN EACH OLYMPIC GAMES.

select team, sport, games, count(1) as total\_medals

from olympics\_history

where medal <> 'NA'

and team = 'India' and sport = 'Hockey'

group by team, sport, games

order by total\_medals desc;