SQL CHALLENGES

Scenario based data challenges with solutions

Rajanand Ilangovan

download.rajanand.org

Table of Contents

SQL CHALLENGES

Prepared by Rajanand Ilangovan

01. CRICKET TEAM SELECTION

Question:

The Indian Premier League (IPL) is planning to start a new T10 series and want to launch a pilot quickly. The management wants to select the players into three teams randomly instead of auction.

Write an SQL query to randomly group players into three teams. Each team should have one all rounder, spin bowler, and wicket keeper and two batsman, and fast bowler.

dbo.teams

team_id	team_name
1	Chennai Super Kings
2	Royal Challengers Bangalore
3	Kolkata Knight Riders

dbo.players

player_id	player_name	role
1	Virat Kohli	Batsman
2	Joe Root	Batsman
3	Steven Smith	Batsman
4	Babar Azam	Batsman
5	David Warner	Batsman
6	Jos Buttler	Batsman
7	Adam Gilchrist	Wicket Keeper
8	MS Dhoni	Wicket Keeper
9	Kumar Sangakkara	Wicket Keeper
10	Ravindra Jadeja	All Rounder
11	Hardik Pandya	All Rounder
12	Glenn Maxwell	All Rounder
13	Ravichandran Ashwin	Spin Bowler
14	Muttiah Muralitharan	Spin Bowler
15	Anil Kumble	Spin Bowler
16	Jasprit Bumrah	Fast Bowler
17	Umran Malik	Fast Bowler
18	Brett Lee	Fast Bowler
19	Shaun Tait	Fast Bowler
20	Shoaib Akthar	Fast Bowler
21	James Anderson	Fast Bowler

Each team should have 7 players

Role	Player Count
All Rounder	1
Batsman	2
Fast Bowler	2
Spin Bowler	1
Wicket Keeper	1
	7

example output

team_name	role	player_name
Chennai Super Kings	All Rounder	Hardik Pandya
Chennai Super Kings	Batsman	Jos Buttler
Chennai Super Kings	Batsman	Virat Kohli
Chennai Super Kings	Fast Bowler	Shaun Tait
Chennai Super Kings	Fast Bowler	James Anderson
Chennai Super Kings	Spin Bowler	Anil Kumble
Chennai Super Kings	Wicket Keeper	MS Dhoni
Kolkata Knight Riders	All Rounder	Glenn Maxwell
Kolkata Knight Riders	Batsman	Babar Azam
Kolkata Knight Riders	Batsman	Steven Smith
Kolkata Knight Riders	Fast Bowler	Umran Malik
Kolkata Knight Riders	Fast Bowler	Brett Lee
Kolkata Knight Riders	Spin Bowler	Muttiah Muralitharan
Kolkata Knight Riders	Wicket Keeper	Kumar Sangakkara
Royal Challengers Bangalore	All Rounder	Ravindra Jadeja
Royal Challengers Bangalore	Batsman	Joe Root
Royal Challengers Bangalore	Batsman	David Warner
Royal Challengers Bangalore	Fast Bowler	Jasprit Bumrah
Royal Challengers Bangalore	Fast Bowler	Shoaib Akthar
Royal Challengers Bangalore	Spin Bowler	Ravichandran Ashwin
Royal Challengers Bangalore	Wicket Keeper	Adam Gilchrist

SQL CHALLENGES

Prepared by Rajanand Ilangovan

01. CRICKET TEAM SELECTION

Solution:

```sql
;with team_selection as (
    select player_id,
    player_name,
    role,
    ntile(3) over(partition by role order by newid()) as team_id
    from players
)
select team_name,
role,
player_name
from team_selection as p
inner join teams as t on t.team_id = p.team_id
order by team_name,
role
```

This solution is implemented using NTILE ranking function. This function distributes the players into three groups for each role in random order.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

02. CONSISTENT PERFOMER

Question:

You have two tables called players and score_details. The player table contains player detail and score_details table contains each innings the players have played and runs scored.

Write an SQL query to list out the players who have consecutively scored 30+ runs at least 3 times.

dbo.players

player_id	player_name
1	Devon Conway
2	Ruturaj Gaikwad
3	Ambati Rayudu
4	Robin Uthappa

dbo.score_detail

innings_id	player_id	score
1	1	35
1	2	50
1	3	20
1	4	30
2	1	5
2	2	40
2	3	8
2	4	32
3	1	40
3	2	77
3	3	25
3	4	44
4	1	62
4	2	54
4	3	20
4	4	31
5	1	2
5	2	82
5	3	29
5	4	1

expected output

player_name	scored_at_least	consecutive_innings
Ruturaj Gaikwad	30	5
Robin Uthappa	30	4

SQL CHALLENGES

Prepared by Rajanand Ilangovan

02. CONSISTENT PERFOMER

Solution #1:

```sql
-- #solution 1
declare @min_score int = 30;
declare @consecutive_innings int = 3;

;with cte1 as (
    select player_id,
    innings_id,
    case when score ≥ @min_score then @min_score else 0 end as score
    from dbo.score_detail
), cte2 as (
    select player_id,
    score,
    (row_number() over(partition by player_id order by score, innings_id) - innings_id) as grp
    from cte1
)
select player_name, score as scored_at_least, count(1) as consecutive_innings
from cte2
inner join dbo.players as p on cte2.player_id = p.player_id
where score = @min_score
group by player_name, score, grp
having count(1) ≥ @consecutive_innings
order by consecutive_innings desc, player_name asc
```

cte1 - If the score is above 30, then 30 else 0.
cte2 - Create a row number for each player ordered by score and innings id. Then find the difference of their innings id.
Then group the result based on the difference calculated in cte2.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

02. CONSISTENT PERFOMER

Solution #2:

```sql
-- #solution 2
declare @min_score int = 30;
declare @consecutive_innings int = 3;

;with cte1 as (
    select player_id,
    innings_id,
    case when score ≥ @min_score then @min_score else 0 end as score
    from dbo.score_detail

), cte2 as (
    select player_id,
    innings_id,
    score,
    case when lag(score) over(partition by player_id order by innings_id) - score = 0 then 0 else 1 end as diff
    from cte1
), cte3 as (
    select player_id,
    score,
    sum(diff) over(partition by player_id order by innings_id) as grp
    from cte2
)
select player_name, score as scored_at_least, count(1) as consecutive_innings
from cte3
inner join dbo.players as p on cte3.player_id = p.player_id
where score = @min_score
group by player_name, score, grp
having count(1) ≥ @consecutive_innings
order by consecutive_innings desc, player_name asc
```

cte1 - If the score is above 30, then 30 else 0.
cte2 - Calculate the difference. If the previous innings score and current innings score is same then 0 else 1
cte3 - Find the running total of the difference for each player based on innings_id order.
Then group the result based on the running total calculated in cte3.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

03. TRAVELLER'S DILEMMA

Question:

You are planning to go for a summer vacation and decided on the cities you want to visit. But you have not finalized in which order you want to visit them yet.

Write a SQL query to list out all different possible order you can visit these cities.
Note that you neither want to visit the same city again nor skip any city in your travel plan.

dbo.city

id	city_name
1	Oslo
2	Helsinki
3	Stockholm
4	Copenhagen

expected output

id	travel_path
1	Copenhagen -> Helsinki -> Oslo -> Stockholm
2	Copenhagen -> Helsinki -> Stockholm -> Oslo
3	Copenhagen -> Oslo -> Helsinki -> Stockholm
4	Copenhagen -> Oslo -> Stockholm -> Helsinki
5	Copenhagen -> Stockholm -> Helsinki -> Oslo
6	Copenhagen -> Stockholm -> Oslo -> Helsinki
7	Helsinki -> Copenhagen -> Oslo -> Stockholm
8	Helsinki -> Copenhagen -> Stockholm -> Oslo
9	Helsinki -> Oslo -> Copenhagen -> Stockholm
10	Helsinki -> Oslo -> Stockholm -> Copenhagen
11	Helsinki -> Stockholm -> Copenhagen -> Oslo
12	Helsinki -> Stockholm -> Oslo -> Copenhagen
13	Oslo -> Copenhagen -> Helsinki -> Stockholm
14	Oslo -> Copenhagen -> Stockholm -> Helsinki
15	Oslo -> Helsinki -> Copenhagen -> Stockholm
16	Oslo -> Helsinki -> Stockholm -> Copenhagen
17	Oslo -> Stockholm -> Copenhagen -> Helsinki
18	Oslo -> Stockholm -> Helsinki -> Copenhagen
19	Stockholm -> Copenhagen -> Helsinki -> Oslo
20	Stockholm -> Copenhagen -> Oslo -> Helsinki
21	Stockholm -> Helsinki -> Copenhagen -> Oslo
22	Stockholm -> Helsinki -> Oslo -> Copenhagen
23	Stockholm -> Oslo -> Copenhagen -> Helsinki
24	Stockholm -> Oslo -> Helsinki -> Copenhagen

SQL CHALLENGES

Prepared by Rajanand Ilangovan

03. TRAVELLER'S DILEMMA

Solution #1:

```
-- solution 1
declare @total_cities int = (select count(1) from dbo.city);
;with travel (travel_path, level) as (
    select cast(city_name as varchar(200)),
    level = 1
    from dbo.city
    union all
    select cast(travel.travel_path + ' -> ' + city.city_name as varchar(200)),
    level = level + 1
    from dbo.city
    inner join travel on level < @total_cities
    where charindex(city.city_name, travel.travel_path) = 0
    )

select
id = row_number() over(order by travel_path),
travel_path
from travel
where level = @total_cities
order by id
```

This solution is implemented using recursive CTE as you need to find all the possible combination. As you need to have a plan with all the four cities, we are filtering only the plan that has hour cities.

As you should not visit the same city twice, we are using charindex function to find if the city is present in the travel plan already. If it is, then ignore that plan.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

03. TRAVELLER'S DILEMMA

Solution #2:

```
-- solution 2
;with bitmasks as (
    select cast(city_name as varchar(max)) as city_name,
    cast(power(2, row_number() over (order by city_name) - 1) as int) as bitmask
    from dbo.city
),
travel as (
    select city_name as travel_path,
    bitmask
    from bitmasks
    union all
    select p.travel_path + ' -> ' + b.city_name,
    p.bitmask ^ b.bitmask
    from travel p
    join bitmasks b on p.bitmask ^ b.bitmask > p.bitmask
)
select travel_path
from travel
where bitmask = power(2, (select count(*) from dbo.city)) - 1
order by travel_path
```

This solution is implemented using a recursive CTE, bitmask and bitwise exclusive OR (^) operator.

bitmasks - Create bitmask (1,2,4,8) for each city.
travel - Recursive CTE with exclusive OR to ignore the plan if the city is already in the travel plan.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

04. UNGROUP TABLE

Question:

You have an orders table with orders details. You have to ungroup the data based on the order quantity. The amount column in the output should be an amount of a single quantity.

Assume that the product's amount will be same regardless of the quantity ordered.

Write an SQL query to degroup the orders table.

dbo.orders

order_id	product	quantity	amount
1001	Laptop	1	75000
1001	Monitor	2	30000
1002	Speaker	4	12000

expected output

order_id	product	quantity	amount
1001	Laptop	1	75000
1001	Monitor	1	15000
1001	Monitor	1	15000
1002	Speaker	1	3000
1002	Speaker	1	3000
1002	Speaker	1	3000
1002	Speaker	1	3000

SQL CHALLENGES

Prepared by Rajanand Ilangovan

04. UNGROUP TABLE

Solution:

```
declare @max_num int = (select max(quantity) from order_details)

;with numbers(num) as (
    select 1
    union all
    select num+1
    from numbers
    where num ≤ @max_num
)

select order_id,
product,
1 as quantity,
cast(amount / quantity as decimal(18,2)) as amount
from order_details
cross join numbers
where quantity ≥ num
order by order_id,
product
```

This solution is implemented using number sequence and cross join. We have generated a number sequence using a recursive CTE and then cross join this number sequence table with the orders table based on the quantity in orders table.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

05. COUNTRY DROPDOWN

Question:

There is a retail company in US which majorly serves customers in US, UK and Canada. They want the shipping country drop down in the web application to list the countries in the below order.

US, UK, Canada and rest of the countries they serve in ascending order.

Write an SQL query to sort the dropdown country value in custom order.

dbo.country

country_code	country_name
AF	Afghanistan
BH	Bahrain
CA	Canada
DK	Denmark
EC	Ecuador
FO	Faroe Islands
DE	Germany
HT	Haiti
IS	Iceland
JM	Jamaica
KZ	Kazakhstan
LA	Laos
MG	Madagascar
NA	Namibia
OM	Oman
PK	Pakistan
QA	Qatar
RO	Romania
CH	Switzerland
TW	Taiwan
UK	United Kingdom
US	United States of America
VA	Vatican City
WF	Wallis and Futuna
YE	Yemen
ZM	Zambia

expected output

country_code	country_name
US	**United States of America**
UK	**United Kingdom**
CA	**Canada**
AF	Afghanistan
BH	Bahrain
DK	Denmark
EC	Ecuador
FO	Faroe Islands
DE	Germany
HT	Haiti
IS	Iceland
JM	Jamaica
KZ	Kazakhstan
LA	Laos
MG	Madagascar
NA	Namibia
OM	Oman
PK	Pakistan
QA	Qatar
RO	Romania
CH	Switzerland
TW	Taiwan
VA	Vatican City
WF	Wallis and Futuna
YE	Yemen
ZM	Zambia

SQL CHALLENGES

Prepared by Rajanand Ilangovan

05. COUNTRY DROPDOWN

Solution:

```sql
select * from dbo.country
order by case when country_name = 'United States of America' then '1'
              when country_name = 'United Kingdom' then '2'
              when country_name = 'Canada' then '3'
         else country_name end asc
```

You can use the CASE expression in order by to do the custom sorting. As you want US, UK, and Canada to be the first three values, we have assigned a string 1 to 3 respectively and for the rest of the countries, their country name.

You are not restricted to use only the string 1 to 3. You can use any string that would put these countries at the top. For example. 'aaa', 'aab' & 'aac'.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

06. EMPLOYEES EARN MORE THEN THEIR MANAGER

Question:

Write an SQL query to list out the employees who earns more than their manager.

Table details:

business_unit - Business unit details.

employee_details - Employee details with salary and manager information.

dbo.employee_details

emp_id	emp_name	bu_id	manager_id	salary
1	Samvel Josef	NULL	NULL	750000
2	Feliciano Chris	1	1	500000
3	Olaf Humberto	1	2	400000
4	Jonatan León	1	2	300000
5	Aron Flemming	1	2	550000
6	Bjørn Lars	2	1	600000
7	Alfredo Lasse	2	6	500000
8	Kristapor Jarl	2	6	250000
9	Silvio Leonardo	3	1	250000
10	Tom Cleto	3	9	150000
11	Emigdio Silvio	4	1	625000
12	Magnus Noé	4	11	600000
13	Simon Johan	4	11	550000
14	Lias Sebastian	4	13	600000
15	Jakob Ezequiel	4	13	450000
16	Lina Nisha	5	6	150000
17	Durga Pravina	5	9	220000
18	Gauri Dipti	5	11	470000
19	Vikram Veda	5	2	225000
20	Sakshi Singh	5	13	100000

dbo.business_unit

bu_id	bu_name
1	Insights and Data
2	Security and Compliance
3	Human Resource Management
4	Application Development
5	Project Management Office

expected output

emp_id	emp_name	manager_name	business_unit_name	emp_salary	manager_salary
5	Aron Flemming	Feliciano Chris	Insights and Data	550000	500000
14	Lias Sebastian	Simon Johan	Application Development	600000	550000

SQL CHALLENGES

Prepared by Rajanand Ilangovan

06. EMPLOYEES EARN MORE THEN THEIR MANAGER

Solution:

```sql
select e.emp_id,
e.emp_name,
m.emp_name as manager_name,
b.bu_name as business_unit_name,
e.salary as emp_salary,
m.salary as manager_salary
from dbo.employee_details as e
inner join dbo.employee_details as m on e.manager_id = m.emp_id
inner join dbo.business_unit as b on e.bu_id = b.bu_id
where e.salary > m.salary
```

This solution is implemented using a self join. The employee_details table is joined with itself to get the manager's salary.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

07. EMPLOYEES EARN MORE THEN THEIR BU AVERAGE

Question:

Write an SQL query to list out the employees who earns more than their business unit average.

Table details:

business_unit - Business unit details.

employee_details - Employee details with salary and manager information.

dbo.employee_details

emp_id	emp_name	bu_id	manager_id	salary
1	Samvel Josef	NULL	NULL	750000
2	Feliciano Chris	1	1	500000
3	Olaf Humberto	1	2	400000
4	Jonatan León	1	2	300000
5	Aron Flemming	1	2	550000
6	Bjørn Lars	2	1	600000
7	Alfredo Lasse	2	6	500000
8	Kristapor Jarl	2	6	250000
9	Silvio Leonardo	3	1	250000
10	Tom Cleto	3	9	150000
11	Emigdio Silvio	4	1	625000
12	Magnus Noé	4	11	600000
13	Simon Johan	4	11	550000
14	Lias Sebastian	4	13	600000
15	Jakob Ezequiel	4	13	450000
16	Lina Nisha	5	6	150000
17	Durga Pravina	5	9	220000
18	Gauri Dipti	5	11	470000
19	Vikram Veda	5	2	225000
20	Sakshi Singh	5	13	100000

dbo.business_unit

bu_id	bu_name
1	Insights and Data
2	Security and Compliance
3	Human Resource Management
4	Application Development
5	Project Management Office

expected output

emp_id	emp_name	business_unit_name	salary	avg_bu_salary
2	Feliciano Chris	Insights and Data	500000	437500
5	Aron Flemming	Insights and Data	550000	437500
6	Bjørn Lars	Security and Compliance	600000	450000
7	Alfredo Lasse	Security and Compliance	500000	450000
9	Silvio Leonardo	Human Resource Management	250000	200000
11	Emigdio Silvio	Application Development	625000	565000
12	Magnus Noé	Application Development	600000	565000
14	Lias Sebastian	Application Development	600000	565000
18	Gauri Dipti	Project Management Office	470000	233000

SQL CHALLENGES

Prepared by Rajanand Ilangovan

07. EMPLOYEES EARN MORE THEN THEIR BU AVERAGE

Solution:

```sql
;with bu_average as (
    select b.bu_id,
    avg(salary) as avg_bu_salary
    from dbo.employee_details as e
    inner join dbo.business_unit as b on b.bu_id = e.bu_id
    group by b.bu_id
)
select e.emp_id,
e.emp_name,
b.bu_name as business_unit_name,
e.salary,
a.avg_bu_salary
from dbo.employee_details as e
inner join bu_average as a on e.bu_id = a.bu_id
inner join dbo.business_unit as b on b.bu_id = e.bu_id
where e.salary > a.avg_bu_salary
```

We have calculated the average salary of each BU in bu_average CTE. Then compared the BU average salary with the employee salary.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

08. EMPLOYEES EARN MORE THEN THEIR PEERS

Question:

Write an SQL query to list out the employees who earns more than their peers. Employees with same managers are considered as peers.

Table details:

business_unit - Business unit details.

employee_details - Employee details with salary and manager information.

dbo.employee_details

emp_id	emp_name	bu_id	manager_id	salary
1	Samvel Josef	NULL	NULL	750000
2	Feliciano Chris	1	1	500000
3	Olaf Humberto	1	2	400000
4	Jonatan León	1	2	300000
5	Aron Flemming	1	2	550000
6	Bjørn Lars	2	1	600000
7	Alfredo Lasse	2	6	500000
8	Kristapor Jarl	2	6	250000
9	Silvio Leonardo	3	1	250000
10	Tom Cleto	3	9	150000
11	Emigdio Silvio	4	1	625000
12	Magnus Noé	4	11	600000
13	Simon Johan	4	11	550000
14	Lias Sebastian	4	13	600000
15	Jakob Ezequiel	4	13	450000
16	Lina Nisha	5	6	150000
17	Durga Pravina	5	9	220000
18	Gauri Dipti	5	11	470000
19	Vikram Veda	5	2	225000
20	Sakshi Singh	5	13	100000

dbo.business_unit

bu_id	bu_name
1	Insights and Data
2	Security and Compliance
3	Human Resource Management
4	Application Development
5	Project Management Office

expected output

emp_id	emp_name	business_unit_name	salary
5	Aron Flemming	Insights and Data	550000
7	Alfredo Lasse	Security and Compliance	500000
11	Emigdio Silvio	Application Development	625000
12	Magnus Noé	Application Development	600000
14	Lias Sebastian	Application Development	600000
17	Durga Pravina	Project Management Office	220000

SQL CHALLENGES

Prepared by Rajanand Ilangovan

08. EMPLOYEES EARN MORE THEN THEIR PEERS

Solution:

```sql
;with max_peer_salary as (
    select emp_id,
    emp_name,
    b.bu_name as business_unit_name,
    salary,
    max(salary) over(partition by manager_id) as max_peer_salary
    from dbo.employee_details as e
    inner join dbo.business_unit as b on e.bu_id = b.bu_id
    where manager_id is not null
)
select emp_id,
emp_name,
business_unit_name,
salary
from max_peer_salary
where salary = max_peer_salary
```

We have calculated the maximum salary of direct reportees of each manager using a window function. Then filter the employees with that maximum salary to list the employees.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

09. MANAGER AND DIRECT REPORTS - 1

Question:

Write an SQL query to list out the employees who earns more than their peers. Employees with same managers are considered as peers.

Table details:

business_unit - Business unit details.

employee_details - Employee details with salary and manager information.

dbo.employee_details

emp_id	emp_name	bu_id	manager_id	salary
1	Samvel Josef	NULL	NULL	750000
2	Feliciano Chris	1	1	500000
3	Olaf Humberto	1	2	400000
4	Jonatan León	1	2	300000
5	Aron Flemming	1	2	550000
6	Bjørn Lars	2	1	600000
7	Alfredo Lasse	2	6	500000
8	Kristapor Jarl	2	6	250000
9	Silvio Leonardo	3	1	250000
10	Tom Cleto	3	9	150000
11	Emigdio Silvio	4	1	625000
12	Magnus Noé	4	11	600000
13	Simon Johan	4	11	550000
14	Lias Sebastian	4	13	600000
15	Jakob Ezequiel	4	13	450000
16	Lina Nisha	5	6	150000
17	Durga Pravina	5	9	220000
18	Gauri Dipti	5	11	470000
19	Vikram Veda	5	2	225000
20	Sakshi Singh	5	13	100000

dbo.business_unit

bu_id	bu_name
1	Insights and Data
2	Security and Compliance
3	Human Resource Management
4	Application Development
5	Project Management Office

expected output

manager_id	manager_name	direct_reports
1	Samvel Josef	Feliciano Chris,Bjørn Lars,Silvio Leonardo,Emigdio Silvio
2	Feliciano Chris	Vikram Veda,Olaf Humberto,Jonatan León,Aron Flemming
6	Bjørn Lars	Alfredo Lasse,Kristapor Jarl,Lina Nisha
9	Silvio Leonardo	Durga Pravina,Tom Cleto
11	Emigdio Silvio	Gauri Dipti,Magnus Noé,Simon Johan
13	Simon Johan	Lias Sebastian,Jakob Ezequiel,Sakshi Singh

SQL CHALLENGES

Prepared by Rajanand Ilangovan

09. MANAGER AND DIRECT REPORTS - 1

Solution:

```sql
select
e.manager_id,
m.emp_name as manager_name,
string_agg(e.emp_name,',') as direct_reports
from dbo.employee_details as e
inner join dbo.employee_details as m on e.manager_id = m.emp_id
group by e.manager_id,
m.emp_name
```

We have used STRING_AGG function and GROUP BY clause to list the direct reports as comma separated value. There are other ways to achieve this same result using COALESCE, STUFF and SUBSTRING instead of STRING_AGG function.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

II. MANAGER AND DIRECT REPORTS - II

Question:

Write an SQL query to list out the employees who earns more than their peers. Employees with same managers are considered as peers.

Table details:

business_unit - Business unit details.

employee_details - Employee details with salary and manager information.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

II. MANAGER AND DIRECT REPORTS - II

Solution:

```sql
select
e.manager_id,
m.emp_name as manager_name,
string_agg(e.emp_name,',') as direct_reports
from dbo.employee_details as e
inner join dbo.employee_details as m on e.manager_id = m.emp_id
group by e.manager_id,
m.emp_name
```

We have used STRING_AGG function and GROUP BY clause to list the direct reports as comma separated value. There are other ways to achieve this same result using COALESCE, STUFF and SUBSTRING instead of STRING_AGG function.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

10. TRAVEL HOURS

Question:

You have a table called travel_detail and you need to calculate the total travel hours between the cities.

For example, the total travel hours between Oslo and Helsinki is 235 (i.e 125+110)

Write an SQL query to calculate the total travel hours between cities.

dbo.travel_detail

id	from_city	to_city	travel_time_hours
1	Oslo	Helsinki	125
2	Helsinki	Oslo	110
3	Stockholm	Oslo	132
4	Oslo	Stockholm	180
5	Copenhagen	Helsinki	148
6	Helsinki	Copenhagen	84
7	Stockholm	Copenhagen	116
8	Helsinki	Stockholm	124

expected output

city_1	city_2	total_travel_hours
Copenhagen	Helsinki	232
Copenhagen	Stockholm	116
Helsinki	Oslo	235
Helsinki	Stockholm	124
Oslo	Stockholm	312

SQL CHALLENGES

Prepared by Rajanand Ilangovan

10. TRAVEL HOURS

Solution 1:

```sql
-- solution 1
;with travel(city_1, city_2, travel_time_hours) as (
    select
    city_1 = case when from_city < to_city then from_city else to_city end,
    city_2 = case when from_city > to_city then from_city else to_city end,
    travel_time_hours
    from dbo.travel_detail
)
select
city_1,
city_2,
sum(travel_time_hours) as total_travel_hours
from travel
group by city_1, city_2
order by city_1, city_2
```

You have to consider the total travel hours between city1 to city2 and city2 to city1 as one pair. So we are comparing the city names with less than and greater than operator to swap to the same side. Then we just aggregate the trave time hours.

SQL CHALLENGES

Prepared by Rajanand Ilangovan

10. TRAVEL HOURS

Solution 2:

```sql
-- solution 2
;with travel(city_1, city_2, travel_time_hours) as (
    select from_city, to_city, travel_time_hours
    from dbo.travel_detail
    where from_city < to_city
    union all
    select to_city, from_city, travel_time_hours
    from dbo.travel_detail
    where from_city > to_city
)
select
city_1,
city_2,
sum(travel_time_hours) as total_travel_hours
from travel
group by city_1,city_2
order by city_1
```

This implementation also uses the same approach as the solution 1. But union all is used to combine the two result set. The from_city and to_city is swapped in the upper and lower part of the union all.

Interested in solution?

DOWNLOAD NOW

link.rajanand.org/sql-challenges