Q1) Given a table TBL with the following schema

ID (INT32) Parent\_ID (INT32)

31 3

32 3

33 4

34 31

35 32

3 NULL

4 NULL

Write a query which will produce a result like this {id, root\_parent } where root\_parent != NULL

And another query to produce output {id, level} where level shows the maximum level of depth of id.

Q2) A dataset contains dates in the following formats

MM/DD/YYYY or M/D/YYYY or M/D/YY

where M for say month of May is 5 and MM is 05, similar for D, DD, YY ….

Translate this into format YYYY-MM-DD ---- Use GoogleSheet RegEX functions to achieve the same

Q3) We have a table Acct (id, name) and a table Contact (id, acct\_id, name, email, role)

An account can have multiple contacts.

The role can be one or more of { Admin, Sales\_Rep, Support\_Person } as a delimited string “;” in any order e.g. “Admin; Support\_Person” or “Sales\_Rep, Admin”

1. Write a query to find all such accts that do not have any contact with Admin role
2. (hint: this may need a nested query or subquery)

Q4) Describe some advantages of Spanner over other RDBMS

Q5) Describe the differences between No-SQL and traditional RDBMS databases.

Q6) Given the table below, find Number of orders shipped in each year that are out of balance compared to line items (based on the amounts).

**ORDER**

Order\_id

customer\_id

Order\_amount

order\_status (‘IN-PROCESS’/’COMPLETED’/’CANCELLED’)

order\_date

shipment\_date

**ORDER\_LINE\_ITEM**

order \_id

Order\_line\_item\_id

line\_item\_description

Line\_item\_amount

Q7) Given two tables:

Table 1: Projects (Project ID, Manager, Project Name, Status)

Table 2: Projects Risks (Project ID, Risk Name, Date)

Write a query that lists each manager with their latest risk per project.

Q8) Given a table with 4 cols (Emp\_id, name, date\_of\_joining, dept\_id) write q query to show the number of employees in each department that have more than 50 employees sorted desc on strength of employees.

1. Generate a serial number for this result set as part of the query
2. b) The records now have a repeated field called Project, Role where both are enums. Enhance the query to find only those departments where number of managers is more than 10.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ANSWERS: do not look at it.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Q1) Given a table TBL with the following schema

ID (INT32) Parent\_ID (INT32)

31 3

32 3

33 4

34 31

35 32

3 NULL

4 NULL

Write a query which will produce a result like this {id, root\_parent } where root\_parent != NULL

And another query to produce output {id, level} where level shows the maximum level of depth of id.

-- GETTING EVERY CHILD RECORD

With child as

(

Select

t1.id as id,

t1.parent\_id as parent\_id

From

TBL as t1

Where t1.parent\_id is not null

),

-- getting root

root as

(

Select

t2.id as id,

t1.parent\_id as parent\_id

From TBL as t2

Where t2.parent\_id is null

)

-- getting parent of one step above

Select c1.id , c2.parent\_id

From

Child c1

inner Join child c2

on c1.parent\_id = c2.id

;

-- to loop till root parent using self join

With id\_root as (

Select

Id, parent\_id as root\_id

From tbl

Where parent\_id is null

Union all

Select

t1.id, rt.id as root\_id

From tbl as t1

Left join id\_root as rt

On t1.parent\_id = rt.id

)

Select \* from id\_root

;

-- to get the level

With id\_root as (

Select

Id, parent\_id as root\_id, 0 as level

From tbl

Where parent\_id is null

Union all

Select

t1.id, rt.id as root\_id, level + 1

From tbl as t1

Left join id\_root as rt

On t1.parent\_id = rt.id

)

Select \* from id\_root

;

Q2) A dataset contains dates in the following formats

MM/DD/YYYY or M/D/YYYY or M/D/YY

where M for say month of May is 5 and MM is 05, similar for D, DD, YY ….

Translate this into format YYYY-MM-DD ---- Use GoogleSheet RegEX functions to achieve the same

CASE 1:

=TEXT(DATE(REGEXREPLACE(A2,"^(\d+).(\d+).(\d+)$","$3"),REGEXREPLACE(A2,"^(\d+).(\d+).(\d+)$","$2"),REGEXREPLACE(A2,"^(\d+).(\d+).(\d+)$","$1")),"YYYY-MM-DD")

Assumption: when the year is YY can i assume it is from the 21st century

CASE TO HANDLE 2 digit year YY

=TEXT(DATE(IF(LEN(REGEXREPLACE(A3,"^(\d+).(\d+).(\d+)$","$3"))=2,TEXT(REGEXREPLACE(A3,"^(\d+).(\d+).(\d+)$","$3"),"2000"),REGEXREPLACE(A3,"^(\d+).(\d+).(\d+)$","$3")),REGEXREPLACE(A3,"^(\d+).(\d+).(\d+)$","$2"),REGEXREPLACE(A3,"^(\d+).(\d+).(\d+)$","$1")),"YYYY-MM-DD")

Q3) We have a table Acct (id, name) and a table Contact (id, acct\_id, name, email, role)

The role can be one or more of { Admin, Sales\_Rep, Support\_Person } as a delimited string “;”

1. Write a query to find all such accts that do not have any contact with Admin role

Select count(distinct(acct\_id))

From contact

where

lower(Role) like ‘%admin%’

-- find account with admins

With acct\_with\_admin as (

SELECT

ACCT\_ID as id

FROM

CONTACT

WHERE

LOWER(ROLE) LIKE ‘%admin%’

-- left join with the cte about where the id is null to find the accounts without admin

Select

accounts.\*

From

Accts as accounts

Left join

acct\_with\_admin

On

accounts.id = acct\_with\_admin.id

Where acct\_with\_admin.id is null

;

Q4) Describe some advantages of Spanner over other RDBMS

advantages:

* Highly available
* Horizontally scalable
* Automatic replication

Disadvantage:

* All tables need PK to be defined ?

Q5) Describe the differences between No-SQL and traditional RDBMS databases.

* No-sql data stores are easily and highly scalable compared to traditional RDBMS
* Data stored is stores unstructured, where as in rdbms data has defined set of columns
* No fixed data model - Can add or remove columns without changing historical data, where as in traditional rdbms we need to back populate and sync the no of columns to match the new data

Q6) Given a table with 4 cols (Emp\_id, name, date\_of\_joining, dept\_id)

1. write a query to show the number of employees in each department that have more than 50 employees sorted with most employees on the top.

Select dept\_id, count(emp\_id) as total\_emp

From

Emp

Group by 1

Having total\_emp > 50

Order by 2 desc

1. Generate a serial number for this result set as part of the query

Select a,dept\_id, total\_emp, rank() over( order by total\_emp desc) as snumber

from

(

Select dept\_id, count(emp\_id) as total\_emp

From

Emp

Group by 1

Having total\_emp > 50

) a

Version2:

Select dept\_id, count(emp\_id) as total\_emp , rank() over( order by count(emp\_id) desc) as snumber

From

Emp

Group by 1

Having total\_emp > 50

1. The records now have a repeated field called Project, Role where both are enums.
   1. Project is a string of the project name.
   2. Role could be values like manager, developer, analyst

Enhance the query to find only those departments where number of managers is more than 10.

SELECT

a.DEPT\_ID,

COUNT(\*) AS TOTAL\_managers

FROM

(

Select

Dept\_id,

PROJECT.NAME as name,

PROJECT.ROLE as role

From

Emp,

unnest(project) as PROJECT

) as a

WHERE

a.role = ‘manager’

Group by

a.DEPT\_ID,

Having

TOTAL\_managers > 10

**Q6) Answer:**

SELECT extract(year from shipment\_date) as shipping\_year, count(distinct(order\_id))

FROM order

INNER JOIN

ORDER\_LINE\_ITEM

ON order.order\_id = ORDER\_LINE\_ITEM.order \_id

WHERE order\_amount != SUM(Line\_item\_amount)

GROUP BY shipping\_year

ORDER BY shipping\_year

**Q7) Answer:**

SELECT Manager, Project Name, Risk Name,

RANK(Project ID) OVER (Partition by Project ID order by date desc limit 1)

FROM Projects as P

INNER JOIN Projects Risks as PR

ON P.Project ID = PR.Project ID

GROUP BY 1,2,3

ORDER BY PR.Date

**Q8) Answer:**

Query:

1)

SELECT count\_employees,

dept\_id

FROM

(

SELECT count(emp\_id) AS count\_employees,

dept\_id

FROM table1

GROUP BY dept\_id

)

WHERE count\_employees > 50

ORDER BY count\_employees DESC;

SELECT count(emp\_id) AS count\_employees,

dept\_id

FROM table1

GROUP BY dept\_id

HAVING count(emp\_id) >50

ORDER BY count\_employees DESC;

2)

SELECT count\_employees,

dept\_id,

RANK() OVER (ORDER BY count\_employees desc) AS serial\_number

FROM

(

SELECT COUNT(emp\_id) AS count\_employees,

dept\_id

FROM table1

GROUP BY dept\_id

HAVING count(emp\_id) >50

ORDER BY count\_employees DESC

);

3)

SELECT dept\_id

FROM (

SELECT emp\_id, dept\_id, project, role

FROM table1

) as t

CROSS JOIN UNNEST t.role as role\_enum

WHERE role\_enum.role = “Manager”

HAVING count(role\_enum.role)>10

SELECT dept\_id

FROM (

SELECT emp\_id, dept\_id, project, role

FROM table1

) as t, t.role AS role\_enum

WHERE role\_enum.role = “Manager”

HAVING COUNT(role\_enum.role)>10

Using t.role in the above query we are essentially creating a replica of the existing table to defragment the enumerated column called role, and then using it in condition to only fetch those rows for which the attribute matches to the title called Manager. This can also be done using CROSS JOIN UNNEST function as well.

Here, only dept\_id attribute is returned as the question asked only to return the department data for which the count of managers was more than 10.