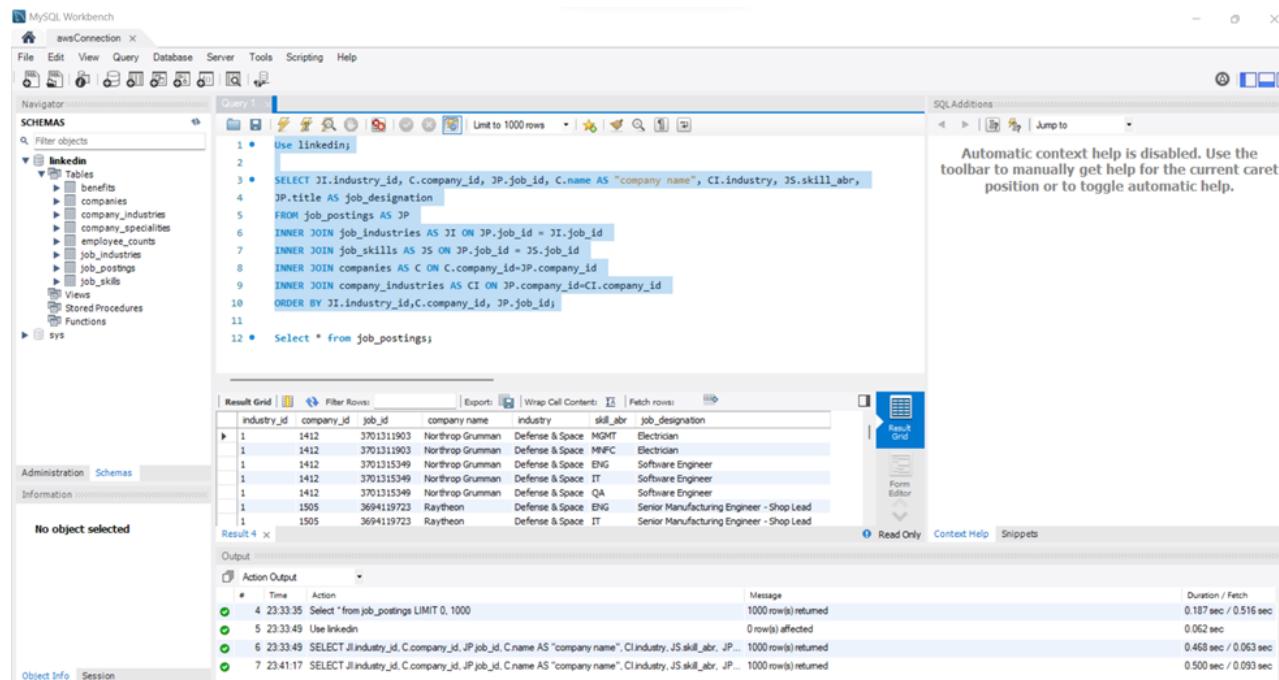


Functional Analysis

1. Job Categorization based on industries, companies and job functions.

Use linkedin;

```
SELECT JI.industry_id, C.company_id, JP.job_id, C.name AS "company name", CI.industry, JS.skill_abr,  
JP.title AS job_designation  
  
FROM job_postings AS JP  
  
INNER JOIN job_industries AS JI ON JP.job_id = JI.job_id  
  
INNER JOIN job_skills AS JS ON JP.job_id = JS.job_id  
  
INNER JOIN companies AS C ON C.company_id=JP.company_id  
  
INNER JOIN company_industries AS CI ON JP.company_id=CI.company_id  
  
ORDER BY JI.industry_id,C.company_id, JP.job_id;
```



The screenshot shows the MySQL Workbench interface with the following details:

- Connection:** awsConnection
- Query Editor:** Contains the SQL query provided above.
- Result Grid:** Displays the results of the query. The columns are industry_id, company_id, job_id, company_name, industry, skill_abr, and job_designation. The data shows multiple rows for Northrop Grumman, Raytheon, and other companies across various industries like Defense & Space, Manufacturing, and IT.
- Output Tab:** Shows the execution log with actions, times, messages, and durations.

industry_id	company_id	job_id	company_name	industry	skill_abr	job_designation
1	1412	3701311903	Northrop Grumman	Defense & Space	MQMT	Electrician
1	1412	3701311903	Northrop Grumman	Defense & Space	MVPC	Electrician
1	1412	3701315349	Northrop Grumman	Defense & Space	ENG	Software Engineer
1	1412	3701315349	Northrop Grumman	Defense & Space	IT	Software Engineer
1	1412	3701315349	Northrop Grumman	Defense & Space	QA	Software Engineer
1	1505	3694119723	Raytheon	Defense & Space	ENG	Senior Manufacturing Engineer - Shop Lead
1	1505	3694119723	Raytheon	Defense & Space	IT	Senior Manufacturing Engineer - Shop Lead

Action	Time	Message	Duration / Fetch
Select * from job_postings	4 23:33:35	1000 row(s) returned	0.187 sec / 0.516 sec
Use linkedin	5 23:33:49	0 row(s) affected	0.062 sec
6 23:33:49 SELECT JI.industry_id, C.company_id, JP.job_id, C.name AS "company name", CI.industry, JS.skill_abr, JP... 1000 row(s) returned			0.468 sec / 0.063 sec
7 23:41:17 SELECT JI.industry_id, C.company_id, JP.job_id, C.name AS "company name", CI.industry, JS.skill_abr, JP... 1000 row(s) returned			0.500 sec / 0.093 sec

2. Geographical distribution of Benefits and job roles provided by companies.

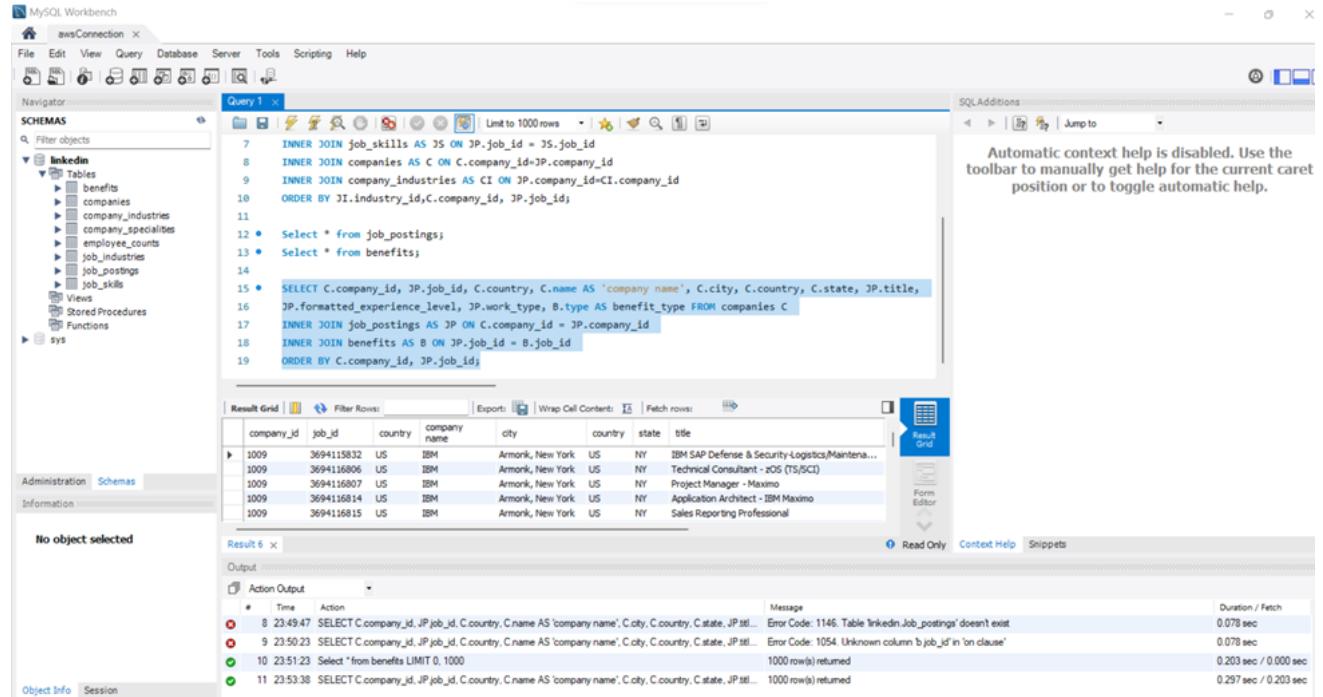
```
SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.state,  
JP.title,
```

```
JP.formatted_experience_level, JP.work_type, B.type AS benefit_type FROM companies C
```

```
INNER JOIN job_postings AS JP ON C.company_id = JP.company_id
```

```
INNER JOIN benefits AS B ON JP.job_id = B.job_id
```

```
ORDER BY C.company_id, JP.job_id;
```



The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar displays the Navigator with the SCHEMAS section expanded, showing the linkedin schema with its tables: benefits, companies, company_industries, company_specialties, employee_counts, job_industries, job_postings, and job_skills. Below the Navigator are tabs for Administration and Schemas, with Schemas selected. The main area contains a Query Editor window titled 'Query 1' with the following SQL code:

```
7 INNER JOIN job_skills AS JS ON JP.job_id = JS.job_id  
8 INNER JOIN companies AS C ON C.company_id=JP.company_id  
9 INNER JOIN company_industries AS CI ON JP.company_id=CI.company_id  
10 ORDER BY CI.industry_id,C.company_id, JP.job_id  
11  
12 • Select * from job_postings;  
13 • Select * from benefits;  
14  
15 • SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.state, JP.title,  
JP.formatted_experience_level, JP.work_type, B.type AS benefit_type FROM companies C  
16 INNER JOIN job_postings AS JP ON C.company_id = JP.company_id  
17 INNER JOIN benefits AS B ON JP.job_id = B.job_id  
18 ORDER BY C.company_id, JP.job_id;
```

The Result Grid below the query editor shows the following data:

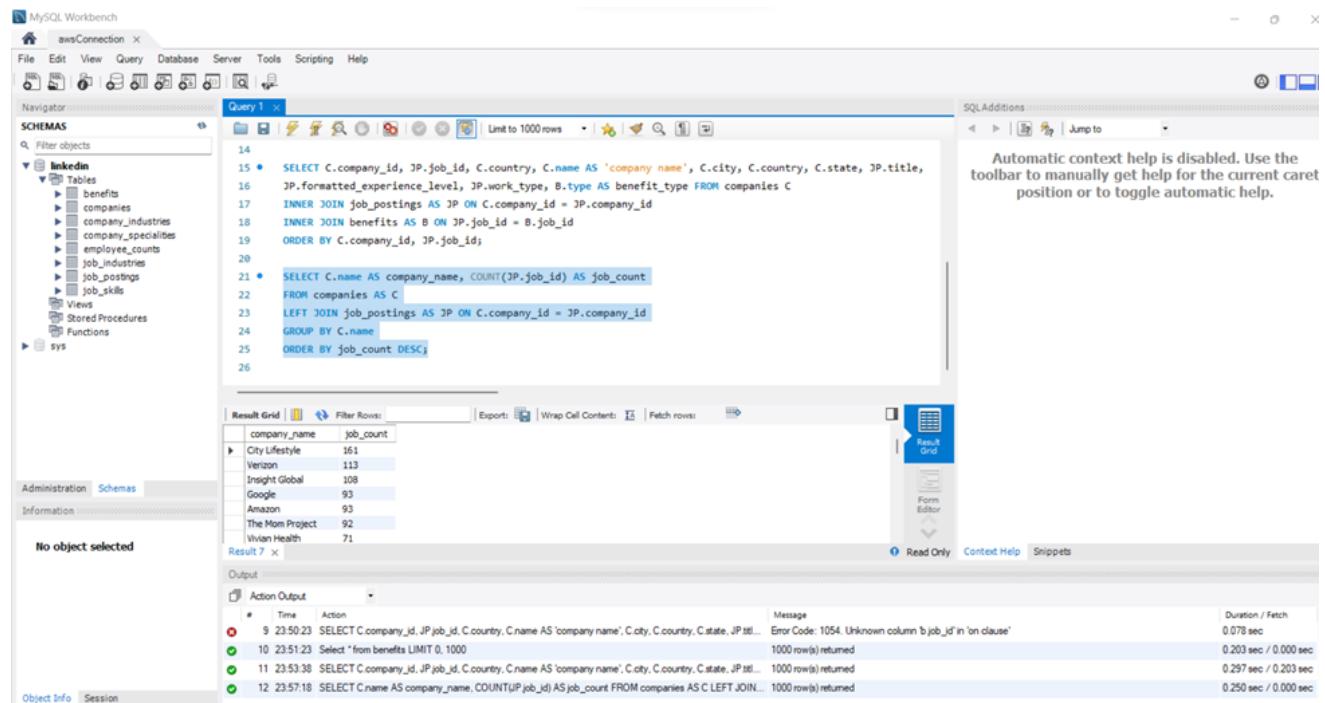
company_id	job_id	country	company_name	city	state	title
1009	3694115832	US	IBM	Armonk, New York	US	NY IBM SAP Defense & Security/Logistics/Maintena...
1009	3694116806	US	IBM	Armonk, New York	US	NY Technical Consultant - xOS (TS/SC)
1009	3694116807	US	IBM	Armonk, New York	US	NY Project Manager - Maximo
1009	3694116814	US	IBM	Armonk, New York	US	NY Application Architect - IBM Maximo
1009	3694116815	US	IBM	Armonk, New York	US	NY Sales Reporting Professional

The bottom pane shows the Action Output log:

#	Time	Action	Message	Duration / Fetch
8	23:49:47	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.state, JP.tit...	Error Code: 1146. Table 'linkedin.Job_postings' doesn't exist	0.078 sec
9	23:50:23	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.state, JP.tit...	Error Code: 1054. Unknown column 'b.job_id' in 'on clause'	0.078 sec
10	23:51:23	Select * from benefits LIMIT 0, 1000	1000 row(s) returned	0.203 sec / 0.000 sec
11	23:53:38	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.state, JP.tit...	1000 row(s) returned	0.297 sec / 0.203 sec

3.Total job postings listed by each company.

```
SELECT C.name AS company_name, COUNT(JP.job_id) AS job_count
FROM companies AS C
LEFT JOIN job_postings AS JP ON C.company_id = JP.company_id
GROUP BY C.name
ORDER BY job_count DESC;
```



The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema structure under the **linkedin** database, including tables like `benefits`, `companies`, `company_industries`, etc.
- Query Editor:** Contains the SQL query provided above.
- Result Grid:** Displays the results of the query, showing company names and their corresponding job counts.
- Output:** Shows the execution log with actions, times, messages, and durations.

company_name	job_count
City Lifestyle	161
Verizon	113
Insight Global	108
Google	93
Amazon	93
The Mom Project	92
Vivan Health	71

#	Time	Action	Message	Duration / Fetch
9	23:50:23	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.country, C.state, JP.title, JP.formatted_experience_level, JP.work_type, B.type AS benefit_type FROM companies C INNER JOIN job_postings AS JP ON C.company_id = JP.company_id INNER JOIN benefits AS B ON JP.job_id = B.job_id ORDER BY C.company_id, JP.job_id;	Error Code: 1054. Unknown column 'b.job_id' in 'on clause'	0.078 sec
10	23:51:23	Select * from benefits LIMIT 0, 1000	1000 row(s) returned	0.203 sec / 0.000 sec
11	23:53:38	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.country, C.state, JP.title, JP.formatted_experience_level, JP.work_type, B.type AS benefit_type FROM companies C LEFT JOIN job_postings AS JP ON C.company_id = JP.company_id GROUP BY C.name ORDER BY job_count DESC;	1000 row(s) returned	0.297 sec / 0.203 sec
12	23:57:18	SELECT C.name AS company_name, COUNT(JP.job_id) AS job_count FROM companies AS C LEFT JOIN job_postings AS JP ON C.company_id = JP.company_id GROUP BY C.name ORDER BY job_count DESC;	1000 row(s) returned	0.250 sec / 0.000 sec

4. Job posting with maximum no. of views and total number of applications.

```
SELECT JP.job_id, JS.skill_abr, JP.company_id, C.name, max_views, JP.applies
FROM job_postings JP
JOIN companies C ON JP.company_id = C.company_id
JOIN job_skills JS ON JP.job_id = JS.job_id
JOIN (
    SELECT MAX/views) AS max_views
    FROM job_postings
) AS max_views_subquery ON JP.views = max_views_subquery.max_views
JOIN (
    SELECT job_id, SUM/applies) AS applies
    FROM job_postings
    GROUP BY job_id
) AS applies_subquery ON JP.job_id = applies_subquery.job_id;
```

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigators

SCHEMAS

linkedin

- Tables
 - benefits
 - companies
 - company_industries
 - company_specialties
 - employee_counts
 - job_industries
 - job_postings
 - job_skills
 - Views
 - Stored Procedures
 - Functions
- sys

Query 1

```

26
27   SELECT JP.job_id, JS.skill_abr, JP.company_id, C.name, max_views, JP.applies
28   FROM job_postings JP
29   JOIN companies C ON JP.company_id = C.company_id
30   JOIN job_skills JS ON JP.job_id = JS.job_id
31   JOIN (
32     SELECT MAX/views) AS max_views
33     FROM job_postings
34   ) AS max_views_subquery ON JP.views = max_views_subquery.max_views
35   JOIN (
36     SELECT job_id, SUM/applies) AS applies
37     FROM job_postings
38   GROUP BY job_id
39   ) AS applies_subquery ON JP.job_id = applies_subquery.job_id;
40
  
```

Result Grid

job_id	skill_abr	company_id	name	max_views	applies
3693044557	IT	2503130	Noom	5656	1420

Administration Schemas

Information

No object selected

Result 2

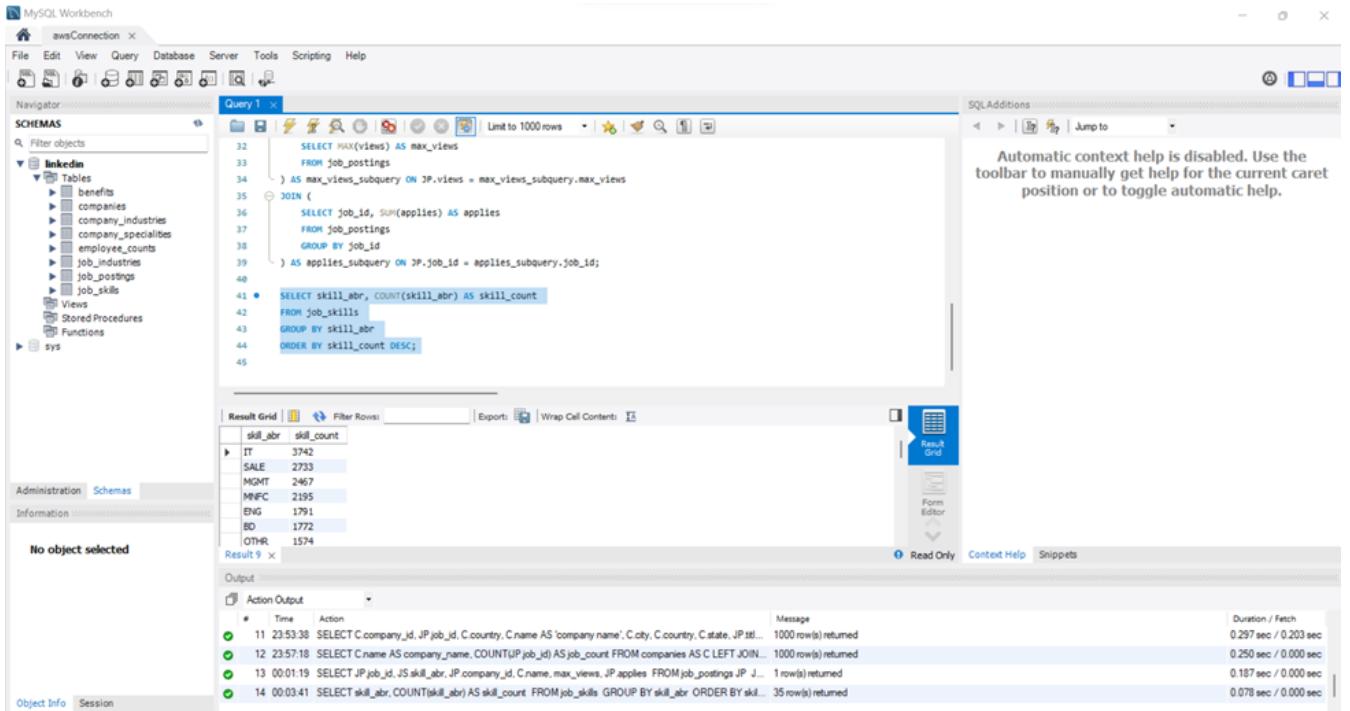
Action Output

#	Time	Action	Message	Duration / Fetch
10	23:51:23	Select *from benefits LIMIT 0,1000	1000 row(s) returned	0.203 sec / 0.000 sec
11	23:53:38	SELECT C.company_id, JP.job_id, C.country, C.name AS 'company name', C.city, C.country, C.state, JP.tl...	1000 row(s) returned	0.297 sec / 0.203 sec
12	23:57:18	SELECT C.name AS company_name, COUNT(JP.job_id) AS job_count FROM companies AS C LEFT JOIN...	1000 row(s) returned	0.250 sec / 0.000 sec
13	00:01:19	SELECT JP.job_id, JS.skill_abr, JP.company_id, C.name, max_views, JP.applies FROM job_postings JP J...	1 row(s) returned	0.187 sec / 0.000 sec

Object Info Session

5. Identifying skills in demand.

```
SELECT skill_abr, COUNT(skill_abr) AS skill_count  
FROM job_skills  
GROUP BY skill_abr  
ORDER BY skill_count DESC;
```



6. Company having highest employee count.

```
SELECT EC.company_id, C.name, MAX(EC.employee_count) AS highest_employee_count
FROM employee_counts EC
INNER JOIN companies AS C ON C.company_id=EC.company_id
GROUP BY EC.company_id, C.name
ORDER BY highest_employee_count desc
LIMIT 1;
```

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- linkedin
 - Tables
 - benefits
 - companies
 - company_industries
 - company_specialties
 - employee_counts
 - job_industries
 - job_postings
 - job_skills
- Views
- Stored Procedures
- Functions
- sys

Query 1 ×

```

34 ) AS max_views_subquery ON JP.views = max_views_subquery.max_views
35 JOIN (
36   SELECT job_id, SUM(applies) AS applies
37   FROM job_postings
38   GROUP BY job_id
39 ) AS applies_subquery ON JP.job_id = applies_subquery.job_id;
40
41 ● SELECT EC.company_id, C.name, MAX(EC.employee_count) AS highest_employee_count
42 FROM employee_counts EC
43 INNER JOIN companies AS C ON C.company_id=Ec.company_id
44 GROUP BY EC.company_id, C.name
45 ORDER BY highest_employee_count desc
46
47 LIMIT 1;
  
```

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Filter Rows | Export: | Wrap Cell Content: | Fetch rows: | Result Grid | Form Editor | Read Only | Context Help | Snippets

company_id	name	highest_employee_count
1586	Amazon	82911

Result 10 ×

Action Output

#	Time	Action	Message	Duration / Fetch
12	23:57:18	SELECT C.name AS company_name, COUNT(JP.job_id) AS job_count FROM companies AS C LEFT JOIN JP.job_postings AS JP ON C.company_id = JP.company_id WHERE JP.job_id IN (SELECT job_id FROM applies_subquery WHERE applies > 1000) GROUP BY C.name ORDER BY job_count DESC LIMIT 1;	1 row(s) returned	0.250 sec / 0.000 sec
13	00:01:19	SELECT JP.job_id, JS.skill_abr, JP.company_id, C.name, max_views, JP.applies FROM job_postings JP JOIN skill_abr JS ON JP.skill_abr = JS.skill_abr JOIN companies C ON JP.company_id = C.company_id WHERE JP.job_id = 1586;	1 row(s) returned	0.187 sec / 0.000 sec
14	00:03:41	SELECT skill_abr, COUNT(skill_abr) AS skill_count FROM job_skills GROUP BY skill_abr ORDER BY skill_abr;	35 row(s) returned	0.078 sec / 0.000 sec
15	00:06:43	SELECT EC.company_id, C.name, MAX(EC.employee_count) AS highest_employee_count FROM employee_counts EC JOIN companies C ON EC.company_id = C.company_id WHERE EC.highest_employee_count = 82911;	1 row(s) returned	0.203 sec / 0.000 sec

Object Info Session

7. Job postings that are currently active, not expired or closed.

```

SELECT job_id, company_id, title, work_type, pay_period, compensation_type, listed_time, expiry,
closed_time, job_posting_url, application_url
FROM linkedin.job_postings
WHERE job_id NOT IN (
SELECT job_id
FROM linkedin.job_postings
WHERE closed_time < CURRENT_TIMESTAMP()
OR expiry < CURRENT_TIMESTAMP()
);

```

The screenshot shows the AWS RDS Query Editor interface. The left sidebar displays the database schema with the 'linkedin' schema selected, showing tables like benefits, companies, Company_Activity_logs, company_industries, company_specialties, employee_counts, job_industries, job_postings, and job_skills. The main area is titled 'Query 7' and contains the SQL code provided above. Below the code, the 'Result Grid' shows the results of the query, listing numerous job postings with columns for job_id, company_id, title, work_type, pay_period, compensation_type, listed_time, expiry, closed_time, job_posting_url, and application_url. The results grid includes various filters and export options. At the bottom, the 'Action Output' section shows the execution log with three entries: a CREATE PROCEDURE statement and two subsequent calls to it, all completed successfully with very low duration times.

job_id	company_id	title	work_type	pay_period	compensation_type	listed_time	expiry	closed_time	job_posting_url	application_url
85008768	NULL	Licensed Insurance Agent	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
133114754	77768602	Sales Manager	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
133196985	1089558	Model Risk Auditor	CONTRACT	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
381055942	96654609	Business Manager	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
529257371	1244539	NY Studio Assistant	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
903408693	3894635	Office Associate	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
967848246	18995316	Education Manager	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	https://...
1004740969	882349	Civil Engineer	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
1028078768	61469	Registered Nurse (RN) Vaccinator	PART_TIME	HOURLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
1053342128	NULL	Company Owner	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
1418862485	NULL	Commercial Property Manager/Senior Pr...	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
148353774	NULL	Video Editor	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	NULL
1535497235	NULL	Administrative Assistant	PART_TIME	HOURLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
1657978824	89330959	REMOTE STEEL BUILDING SALES MA...	CONTRACT	MONTHLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	https://...
1928027033	11544533	Construction Project Manager	CONTRACT	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2024-03-09 16:00:00.000000	NULL	https://www.linkedin.co...	NULL
2148434014	1016	Virtual Sales Associate Account Manager...	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	https://...
2148434019	1016	Virtual Sales Associate Account Manager...	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	https://...
2148434032	17109	Tax Analyst	FULL_TIME	YEARLY	BASE_SALARY	2023-07-22 04:26:40.000000	2023-11-14 22:13:20.000000	NULL	https://www.linkedin.co...	https://...

Action Output

Time	Action	Response	Duration / Fetch Time
00:51:51	CREATE PROCEDURE GET_JOB_FROM_INDUSTRY(IN industry_name VARCHAR(30)) BEGIN SELECT jp.job_id, jp...	0 row(s) affected	0.098 sec
00:51:51	CALL GET_JOB_FROM_INDUSTRY('Retail')	968 row(s) returned	0.138 sec / 0.189 sec
00:59:08	SELECT job_id, company_id, title, work_type, pay_period, compensation_type, listed_time, expiry, closed_time, job_po...	1000 row(s) returned	0.153 sec / 0.187 sec

Query Completed

8.Location wise top 10 companies with the most job postings

```

SELECT j.company_id, c.name AS company_name, COUNT(*) AS job_posting_count, location, j.title,
j.med_salary
FROM job_postings AS j
JOIN companies AS c ON j.company_id = c.company_id

```

```

where j.med_salary != "NULL"

GROUP BY j.company_id, company_name, j.title, location, j.med_salary

ORDER BY COUNT(*) DESC

LIMIT 10;

```

The screenshot shows the AWS RDS Connection interface with the following details:

- Query Editor:** Contains the SQL query provided above.
- Result Grid:** Displays the results of the query, showing 10 rows of data from the job_postings table. The columns are: company_id, company_name, job_posting_count, location, title, and med_salary.
- Output:** Shows the execution log with two entries:

 - Line 2: 16:37:49 SELECT j.company_id, c.name AS company_name, COUNT(*) AS job_posting_count, location, j.title, j.med_salary 10 rows(s) returned
 - Line 3: 16:40:39 SELECT j.company_id, c.name AS company_name, COUNT(*) AS job_posting_count, location, j.title, j.med_salary 10 rows(s) returned

9. Analysis of Job Postings with on Location, Company, Title, and Application Duration and filtered through Inferred through text by linkedin

```

SELECT j.job_id, j.location, c.name, j.title, datediff(expiry, listed_time) as Days_to_apply

FROM job_postings_old AS j

JOIN companies AS c ON j.company_id = c.company_id

JOIN benefits AS jben

ON j.job_id = jben.job_id

where jben.inferred = 1

ORDER BY j.location, j.closed_time DESC, j.max_salary DESC;

```

AIWS RDS Connection ×

Edit View Query Database Server Tools Scripting Help

Navigator: IEMAS

Query 1 ×

```

1 • use linkedin;
2 • SELECT j.job_id,j.location, c.name, j.title, datediff(expiry,listed_time) as Days_to_apply
3 FROM job_postings AS j
4 JOIN companies AS c ON j.company_id = c.company_id
5 JOIN benefits AS jben
6 ON j.job_id = jben.job_id
7 where jben.inferred = 1
8 ORDER BY j.location,j.closed_time DESC, j.max_salary DESC;
9
10

```

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows:

job_id	location	name	title	Days_to_apply
3676480207	Abbeville, SC	Hira Industries LLC	Outside Sales Representative	231
3699084307	Aberdeen Proving Ground, MD	Booz Allen Hamilton	Executive Assistant	115
3697389286	Aberdeen, MD	Parsons Corporation	RF Network Engineer	115
3699417213	Abilene, TX	LocalEdge	Outside Sales - Digital Media Executive	115
3699055661	Abilene, TX	Raising Cane's Chicken Fingers	Restaurant Crewmember - Cook, Cashier, Cust...	115
3701368304	Addison, IL	Parts Town	Senior Marketplace Operations Specialist	115
3701368304	Addison, IL	Parts Town	Senior Marketplace Operations Specialist	115
3701143721	Addison, TX	Salad and Go	Senior Accountant	115
3701143721	Addison, TX	Salad and Go	Senior Accountant	115
3693058043	Addison, TX	Innova Solutions	Social Media Specialist	115
3693058043	Addison, TX	Innova Solutions	Social Media Specialist	115
3699053672	Addison, TX	Raising Cane's Chicken Fingers	Restaurant Crewmember - Late Night/Closing Shift	115
3699054767	Addison, TX	Raising Cane's Chicken Fingers	Restaurant Crewmember	115
3699429382	Adrian, MI	American Trucking Group	CDL A Flatbed Truck Driver	115
3699429382	Adrian, MI	American Trucking Group	CDL A Flatbed Truck Driver	115

Result 4 ×

Output:

Action Output	#	Time	Action	Message
6 16:47:58 use linkedin				0 row(s) affected
7 16:48:01 SELECT j.job_id,j.location, c.name, j.title, datediff(expiry,listed_time) as Days_to_apply FROM job_postings AS...				1000 row(s) returned

Object Info Session

10. Companies and their competitors sharing the same industry

SELECT

```

ci1.company_id AS company_id,
ci2.company_id AS competitor_company_id,
c1.name AS company_name,
c2.name AS competitor_company_name,
ci1.industry AS shared_industry

```

FROM company_industries AS ci1

JOIN company_industries AS ci2 ON ci1.industry = ci2.industry AND ci1.company_id < ci2.company_id

JOIN companies AS c1 ON ci1.company_id = c1.company_id

JOIN companies AS c2 ON ci2.company_id = c2.company_id;

Alv/RDS Connection x

Edit View Query Database Server Tools Scripting Help

gator: MAS

Iter objects

- linkedin
- Tables
- Views
- Stored Procedures
- Functions
- sys

migration Schemas

information

o object selected

Query 1

```

1 • use linkedin;
2 • SELECT
3     c11.company_id AS company_id,
4     c12.company_id AS competitor_company_id,
5     c1.name AS company_name,
6     c2.name AS competitor_company_name,
7     c11.industry AS shared_industry
8 FROM company_industries AS c11
9 JOIN company_industries AS c12 ON c11.industry = c12.industry AND c11.company_id < c12.company_id
10 JOIN companies AS c1 ON c11.company_id = c1.company_id
11 JOIN companies AS c2 ON c12.company_id = c2.company_id;
12

```

Result Grid

company_id	competitor_company_id	company_name	competitor_company_name	shared_industry
1009	1025	IBM	Hewlett Packard Enterprise	Information Technology & Services
1025	1028	Hewlett Packard Enterprise	Oracle	Information Technology & Services
1009	1028	IBM	Oracle	Information Technology & Services
1028	1044	Oracle	PwC	Information Technology & Services
1025	1044	Hewlett Packard Enterprise	PwC	Information Technology & Services
1009	1044	IBM	PwC	Information Technology & Services
1044	1060	PwC	Ericsson	Information Technology & Services
1028	1060	Oracle	Ericsson	Information Technology & Services
1025	1060	Hewlett Packard Enterprise	Ericsson	Information Technology & Services
1009	1060	IBM	Ericsson	Information Technology & Services
1060	1103	Ericsson	Verizon	Information Technology & Services
1044	1103	PwC	Verizon	Information Technology & Services

Output

Action Output

#	Time	Action	Message
1	02:45:05	use linkedin	0 row(s) affected
2	02:45:08	SELECT c11.company_id AS company_id, c12.company_id AS competitor_company_id, c1.name AS com...	1000 row(s) returned

ct Info Session

SQLAddition Autom disabled manual current toggle

Result grid Form Editor Field Types Read Only Context Help

11. Analyze Job Demand Over month:

SELECT

Month(expiry) AS month, SUM/views) AS total_views,

COUNT(*) AS job_count

FROM job_postings_old

GROUP BY month

ORDER BY month;

The screenshot shows the AWS RDS Connection interface with the following details:

- Top Bar:** AWS RDS Connection, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Left Sidebar:** Shows database objects under 'aws' and 'linkedin' schemas, including Tables, Views, Stored Procedures, Functions, sys, and Test.
- Query Editor (Query 1):**

```

14
15 •  SELECT
16   Month(expiry) AS month, SUM/views) AS total_views,
17   COUNT(*) AS job_count
18   FROM job_postings
19   GROUP BY month
20   ORDER BY month;
21
22
23
24
    
```
- Result Grid:**

month	total_views	job_count
3	169545	1237
7	4576	22
11	902493	14627
- Result 7:**

Output

#	Time	Action	Message
10	16:55:53	SELECT Month(expiry) AS month, SUM/views) AS total_views, COUNT(*) AS job_count FROM job_postings ...	3 row(s) returned
11	16:55:58	SELECT Month(expiry) AS month, SUM/views) AS total_views, COUNT(*) AS job_count FROM job_postings ...	3 row(s) returned

12. Partition Method to get the Annual salary on the max_salary specifying range in where clause.

SELECT

```

j.company_id,
j.formatted_work_type,
j.pay_period,
j.max_salary AS max_salary,
max(round(j.max_salary * 12, 3)) OVER (PARTITION BY j.company_id) AS Annual_Salary
FROM job_postings j
where (company_id BETWEEN 1500 AND 2500) and pay_period ='MONTHLY';
    
```

```

32 • SELECT
33     j.company_id,
34     j.formatted_work_type,
35     j.pay_period,
36     j.max_salary AS max_salary,
37
38     MAX(ROUND(j.max_salary * 12, 3)) OVER (PARTITION BY j.company_id) AS Annual_Salary
39
40     FROM job_postings j
41
42     WHERE (company_id BETWEEN 1500 AND 2500) AND pay_period = 'MONTHLY';
43
44
45
46

```

Result Grid

company_id	formatted_work_type	pay_period	max_salary	Annual_Salary
2477	Full-time	MONTHLY	5833.33	69999.96

Action Output

Time	Action	Response	Duration / Fetch Time
02:39:19	SELECT company_id, formatted_work_type, pay_period, max_salary AS max_salary, max(round(max_salary * 12, 3)) OVER (PARTITION BY company_id) AS Annual_... rows(s) returned	0.103 sec / 0.00011...	

13. Analyzing the count of different work types such as Full-Time, Part-Time and Contractor Roles based on their pay_period

SELECT

```

pay_period,
sum(formatted_work_type = 'Full-time') AS "Full-Time Roles",
sum(formatted_work_type = 'Part-time') AS "Part-Time Roles",
sum(formatted_work_type = 'Contract') AS "Contract Roles"
FROM job_postings
GROUP BY pay_period
ORDER BY pay_period;

```

```

57 • SELECT
58     pay_period,
59
60     sum(formatted_work_type = 'Full-time') AS "Full-Time Roles",
61
62     sum(formatted_work_type = 'Part-time') AS "Part-Time Roles",
63
64     sum(formatted_work_type = 'Contract') AS "Contract Roles"
65
66     FROM job_postings
67
68     GROUP BY pay_period
69
70     ORDER BY pay_period;
71
72
73
74

```

Result Grid

pay_period	Full-Time Roles	Part-Time Roles	Contract Roles
HOURLY	1515	281	802
MONTHLY	68	6	7
YEARLY	11261	723	1130

Action Output

Time	Action	Response	Duration / Fetch Time
02:42:41	SELECT pay_period, sum(formatted_work_type = 'Full-time') AS "Full-Time Roles", sum(formatted_work_type = 'Part-time') AS "Part-Time Roles", sum(formatted_work_type = 'Contract') AS "Contract Roles" rows(s) returned	0.119 sec / 0.00013...	

14. Analyzing the number of Full-Time opportunities offered for Monthly pay_period by each company

```
SELECT  
company_id,  
pay_period,  
COUNT(formatted_work_type) AS "Total Full-Time Roles"  
FROM job_postings  
WHERE formatted_work_type = 'Full-time' and pay_period= 'MONTHLY'  
GROUP BY company_id  
ORDER BY company_id;
```

The screenshot shows a database query results interface. At the top, there is a code editor with the following SQL query:

```
88 •  SELECT  
89     company_id,  
90     pay_period,  
91     COUNT(formatted_work_type) AS "Total Full-Time Roles"  
92   FROM job_postings  
93   WHERE formatted_work_type = 'Full-time' and pay_period= 'MONTHLY'  
94 GROUP BY company_id  
95 ORDER BY company_id; |
```

Below the code editor is a results grid with the following data:

company_id	pay_period	Total Full-Time Roles
88	MONTHLY	1
247	MONTHLY	1
264	MONTHLY	1
266	MONTHLY	1
266	MONTHLY	1
1188	MONTHLY	1
1719	MONTHLY	5
2017	MONTHLY	1
2622	MONTHLY	8
4523	MONTHLY	1
53009	MONTHLY	1
73681	MONTHLY	1

At the bottom of the interface, there is a status bar showing the following information:

Result Grid Filter Rows: Search Export:

Time Action Response Duration / Fetch Time

14 02:45:17 SELECT company_id, pay_period, COUNT(formatted_work_type) AS "Total Full-Time Roles" FROM job_postings WHERE formatted_work_type = 'Full-time' and pay_period= 'MONTHLY' 36 row(s) returned 0.182 sec / 0.00072 s..

15. Display the Total Roles in each roles types by every company.

```
SELECT  
company_id,  
formatted_work_type,  
pay_period,
```

```

COUNT(*) OVER (PARTITION BY company_id ORDER BY pay_period) AS "Total Roles"
FROM job_postings
WHERE pay_period != 'NA' and company_id != 0
ORDER BY pay_period;

```

The screenshot shows the execution of a SQL query in a database environment. The code is as follows:

```

128 • SELECT
129     company_id,
130     formatted_work_type,
131     pay_period,
132     COUNT(*) OVER (PARTITION BY company_id ORDER BY pay_period) AS "Total Roles"
133
134     FROM job_postings
135
136 WHERE pay_period != 'NA' and company_id != 0
137
138 ORDER BY pay_period;
139
140
141
142
143
144

```

The results table has columns: company_id, formatted_work_type, pay_period, and Total Roles. The data includes rows for various company IDs and work types, with their respective counts and total roles.

company_id	formatted_work_type	pay_period	Total Roles
96180361	Full-time	HOURLY	5
93749998	Contract	HOURLY	1
89715243	Part-time	HOURLY	1
94203498	Full-time	HOURLY	1
94141082	Full-time	HOURLY	1
94181897	Full-time	HOURLY	1
89721071	Contract	HOURLY	1
95761072	Full-time	HOURLY	1
96081672	Full-time	HOURLY	1
96159230	Full-time	HOURLY	1
96180361	Full-time	HOURLY	5
96180361	Full-time	HOURLY	5

Below the results table is a timeline showing the execution details:

Time	Action	Response	Duration / Fetch Time
02:50:20	SELECT company_id, formatted_work_type, pay_period, COUNT(*) OVER (PARTITION BY company_id ORDER BY pay_period) AS "Total Roles" FROM job_postings WHERE...	15470 row(s) returned	0.272 sec / 0.343 sec

16. Partition Method to get the Annual salary on the max_salary specifying range in where clause.

```

SELECT
j.company_id,
j.formatted_work_type,
j.pay_period,
j.max_salary AS max_salary,
max(round(j.max_salary * 12, 3)) OVER (PARTITION BY j.company_id) AS Annual_Salary
FROM job_postings j
where (company_id BETWEEN 1500 AND 2500) and pay_period ='MONTHLY';

```

```

161 • | SELECT
162 |     j.company_id,
163 |     j.formatted_work_type,
164 |     j.pay_perid,
165 |     j.max_salary AS max_salary,
166 |     max(round(j.max_salary * 12, 3)) OVER (PARTITION BY j.company_id) AS Annual_Salary
167 | FROM job_postings j
168 | WHERE (company_id BETWEEN 1500 AND 2500) AND pay_perid = 'MONTHLY';
169 |
170 |
100% 08:168
Result Grid Filter Rows: Search Export:
company_id formatted_work_type pay_perid max_salary Annual_Salary
2477 Full-time MONTHLY 5833.33 69999.96
Result 11
Action Output: 0
Time Action Response Duration / Fetch Time
19 02:58:52 SELECT j.company_id, j.formatted_work_type, j.pay_perid, j.max_salary AS max_salary, maxround(j.max_salary * 12, 3)) OVER (PARTITION BY j.company_id) AS Annual_Salary Rows(0) returned 0.090 sec / 0.00000...

```

STORED PROCEDURES:

1. Companies with vision Insurance as benefit

DELIMITER //

```

CREATE PROCEDURE GetCompanieswithspecificBenefit(BenefitType VARCHAR(100))
BEGIN
SELECT DISTINCT j.job_id, c.name AS company_name
FROM job_postings AS j
join companies as c on c.company_id = j.company_id
JOIN benefits AS b ON j.job_id = b.job_id
WHERE b.type = BenefitType;
END;
```

```

//  

DELIMITER ;  

call GetCompanieswithspecificBenefit('Vision insurance');

```

The screenshot shows the MySQL Workbench interface with the following details:

- Connection:** AWS RDS Connection
- Query Editor:** Query 1 contains the stored procedure code and a call to it.
- Result Grid:** Shows the output of the query, listing company names corresponding to the specified benefit type.
- Output Window:** Displays the execution log with two entries: the creation of the procedure and the execution of the call.

job_id	company_name
133114754	CargoLogin.
529257371	Ken Fulk Inc
2148434137	Genesys
2634247615	The Reserves Network
2730408164	KISS ABA
3047547711	Franklin Law
3242289926	Compass Family Services
3266185281	J&J Companies
3373346816	Razny Jewelers
3395626440	Sensibly Sprouted
3422104915	TruEar
...	...

Result 8 | Output
Action Output

#	Time	Action	Message
12	16:57:33	CREATE PROCEDURE GetCompanieswithspecificBenefit(BenefitType VARCHAR(100)) BEGIN SELECT DI...	0 row(s) affected
13	16:57:33	call GetCompanieswithspecificBenefit('Vision insurance')	1912 row(s) returned

2. Analysing Industry wise employee count

```

DELIMITER //  

CREATE PROCEDURE CalculateEmployeeCountByIndustry()  

BEGIN  

SELECT ci.industry, c.name, ec.employee_count  

FROM company_industries AS ci  

INNER JOIN companies AS c ON ci.company_id = c.company_id  

INNER JOIN employee_counts AS ec ON ci.company_id = ec.company_id  

GROUP BY ci.industry, c.name, ec.employee_count;  

END;  

//  

DELIMITER ;  

call CalculateEmployeeCountByIndustry();

```

```

AWS RDS Connection x
Edit View Query Database Server Tools Scripting Help
Query 1 x
object selected
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:
industry      name          employee_count
Information Technology & Services IBM           316130
Hospital & Health Care   GE HealthCare    53495
Renewables & Environment GE Power        26963
Information Technology & Services Hewlett Packard Enterprise 70995
Information Technology & Services Oracle        202019
Information Technology & Services Oracle        202050
Management Consulting       Deloitte        437314
Industrial Automation       Siemens        213470
Result 9 x
Output:
Action Output
# Time Action
14 16:59:57 CREATE PROCEDURE CalculateEmployeeCountByIndustry() BEGIN SELECT ci.industry, c.name, ec.employee_count
15 16:59:57 call CalculateEmployeeCountByIndustry()

```

3. Get Job Postings With Metrics by Company Size for given type of work

```

DELIMITER //
CREATE PROCEDURE GetJobPostingsWithMetricsAndSortByCompanySize(type_of_work
VARCHAR(100))
BEGIN
SELECT jp.job_id,c.name AS company_name, jp.title, jp.description, jp.pay_period,
jp.work_type,
c.company_size

FROM job_postings AS jp
INNER JOIN companies AS c ON jp.company_id = c.company_id
where jp.work_type = type_of_work
ORDER BY c.company_size desc;
END;
//
DELIMITER ;
call GetJobPostingsWithMetricsAndSortByCompanySize('PART_TIME');

```

AWS RDS Connection X

Edit View Query Database Server Tools Scripting Help

Query 1 X

Limit to 1000 rows

```

36  DELIMITER //
37  • CREATE PROCEDURE GetJobPostingsWithMetricsAndSortByCompanySize(type_of_work VARCHAR(100))
38  BEGIN
39      SELECT jp.job_id,c.name AS company_name, jp.title, jp.description, jp.pay_period,
40      jp.work_type,
41      c.company_size
42
43      FROM job_postings AS jp
44      INNER JOIN companies AS c ON jp.company_id = c.company_id
45      where jp.work_type = type_of_work
46      ORDER BY c.company_size desc;
47  END;
48  //
49  DELIMITER ;
50  • call GetJobPostingsWithMetricsAndSortByCompanySize('PART_TIME');

```

Result Grid | Filter Rows: Export: Wrap Cell Content: □

job_id	company_name	title	description	pay_period	work_type	company_size
3701372513	99 Cents Only Stores	Store Associate El Segundo #377	At 99 Cents Only Stores, LLC, we are recruiting...	HOURLY	PART_TIME	7
3701369552	99 Cents Only Stores	Store Associate San Bernardino #68	At 99 Cents Only Stores, LLC, we are recruiting...	HOURLY	PART_TIME	7
3701369531	Speedway LLC	Store Leader (Manager) Trainee	Start Your Story at Speedway! We're committe...	YEARLY	PART_TIME	7
3701370883	SCA Health	Radiology Tech, Pool - Carolina Coast Surgery ...	Overview Today, SCA Health has grown to 11,...	HOURLY	PART_TIME	7
3701371520	99 Cents Only Stores	Store Associate San Leandro #180	Our Promise to you is OPPORTUNITY. Join Us a...	HOURLY	PART_TIME	7
3701371520	99 Cents Only Stores	Store Associate San Leandro #180	Our Promise to you is OPPORTUNITY. Join Us a...	HOURLY	PART_TIME	7

Result 10 X

Output

Action Output

#	Time	Action	Message
10	16:55:53	SELECT Month(expiry) AS month, SUM/views) AS total_views, COUNT() AS job_count FROM job_postings ...	3 row(s) returned
11	16:55:58	SELECT Month(expiry) AS month, SUM/views) AS total_views, COUNT() AS job_count FROM job_postings ...	3 row(s) returned
12	16:57:33	CREATE PROCEDURE GetCompanieswithASpecificBenefit(BenefitType VARCHAR(100)) BEGIN SELECT DI...	0 row(s) affected
13	16:57:33	call GetCompanieswithASpecificBenefit('Vision insurance')	1912 row(s) returned

Info Session

4. Find all jobs from a particular industry

DELIMITER \$\$

```

CREATE PROCEDURE GET_JOB_FROM_INDUSTRY(
IN industry_name VARCHAR(30))
BEGIN
SELECT jp.job_id, jp.company_id, jp.title, jp.work_type, jp.location, jp.expiry, jp.closed_time,
jp.posting_domain, jp.job_posting_url, jp.application_url
FROM linkedin.job_postings AS jp
JOIN linkedin.company_industries AS ci
ON jp.company_id = ci.company_id
WHERE ci.industry = industry_name;
END $$
```

DELIMITER ;

```
CALL GET_JOB_FROM_INDUSTRY('Retail');
```

```

DELIMITER $$

CREATE PROCEDURE GET_JOB_FROM_INDUSTRY(
    IN industry_name VARCHAR(30)
)
BEGIN
    SELECT jp.job_id, jp.company_id, jp.title, jp.work_type, jp.location, jp.expiry, jp.closed_time, jp.posting_domain, jp.job_posting_url, jp.
    FROM linkedin.job_postings AS jp
    JOIN linkedin.company_industries AS ci
    ON jp.company_id = ci.company_id
    WHERE ci.industry = industry_name;
END $$

DELIMITER ;

CALL GET_JOB_FROM_INDUSTRY('Retail');

```

job_id	company_id	title	work_type	location	expiry	closed_time	posting_domain	job_posting_url
3693075405	1512	Inbound (Stocking) 4AM Team Members	PART_TIME	Scotts Valley, CA	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699405993	2251	Selling Associate-Cielo Vista	PART_TIME	El Paso, TX	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699406652	2251	Selling Associate-University Mall	PART_TIME	Tuscaloosa, AL	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699407157	2251	Selling Associate-Valley Mall	FULL_TIME	Baltimore, MD	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699407708	2251	Regional Asset Protection Manager (Colorado)	FULL_TIME	Denver, CO	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699407709	2251	Executive Assistant - Legal	FULL_TIME	Reynoldsburg, OH	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699406615	2251	Asset Protection Investigator (San Francisco, California)	FULL_TIME	Daly City, CA	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699406621	2251	Selling Associate-Alien Premium Outlets	PART_TIME	Allen, TX	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699409493	2251	Seasonal Associate-Valley Mall	PART_TIME	Hagerstown, MD	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699409496	2251	Seasonal Associate-Riverchase Galleria	PART_TIME	Hoover, AL	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699410368	2251	Selling Associate-Twelve Oaks Mall	PART_TIME	Novi, MI	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699412260	2251	Selling Associate-Hillside Village	PART_TIME	Cedar Hill, TX	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699412861	2251	Seasonal Associate-Grapevine Mills	PART_TIME	Grapevine, TX	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.victoriassecret.com
3699410451	2252	Retail Assistant Manager-Crossroads Shopping Center	FULL_TIME	Statesville, NC	2023-11-14 22:13:20.000000		HULL	https://www.linkedin.com/jobs/careers.bathandbodyworks.com

Action Output

Action	Time	Action	Response	Duration / Fetch Time
1	00:51:51	CREATE PROCEDURE GET_JOB_FROM_INDUSTRY(IN industry_name VARCHAR(30)) BEGIN SELECT jp.job_id, jp...	0 row(s) affected	0.098 sec
2	00:51:51	CALL GET_JOB_FROM_INDUSTRY('Retail')	968 row(s) returned	0.138 sec / 0.189 sec

5. Display companies offering remote work.

```

DELIMITER //

CREATE PROCEDURE GetRemoteJobs()
BEGIN
SELECT DISTINCT C.name AS 'company name', JP.location, JP.title, JP.job_id
FROM companies C
INNER JOIN job_postings JP ON C.company_id = JP.company_id
WHERE JP.remote_allowed = 1;
END;
// 
DELIMITER ;

```

The screenshot shows the MySQL Workbench interface. In the Navigator pane, under the 'linkedin' schema, there is a 'Tables' section containing tables like 'benefits', 'companies', 'company_industries', 'company_specialties', 'employee_counts', 'job_industries', 'job_postings', 'job_skills'. The 'Stored Procedures' section contains 'GetRemoteJobs()'. The 'Functions' section is empty.

In the Query Editor (Query 1), the following SQL code is displayed:

```

47
48
49  DELIMITER //
50  CREATE PROCEDURE GetRemoteJobs()
51  BEGIN
52      SELECT DISTINCT C.name AS 'company_name', JP.location, JP.title, JP.job_id
53      FROM companies C
54      INNER JOIN job_postings JP ON C.company_id = JP.company_id
55      WHERE JP.remote_allowed = 1;
56  END;
57  //
58  DELIMITER ;
59
60  CALL GetRemoteJobs();
61

```

The Result Grid shows the output of the stored procedure:

company name	location	title	job_id
Paradigm Senior Services	United States	Education Manager	967848246
American Steel Builders	Texas, United States	REMOTE STEEL BUILDING SALES MAKE \$1,000 ...	1657978824
GE HealthCare	Maine, United States	Virtual Sales Associate Account Manager, Wom...	2148434014
GE HealthCare	Connecticut, United States	Virtual Sales Associate Account Manager, Wom...	2148434019
Elastic	Richmond, VA	IT Operations ServiceNow Admin	214843427
Multi Media, LLC	United States	Lead Software Engineer	2649106600
Zero Punk Textile Recycling	Daly City, CA	Internship Offering: Fashion Upcycling Intern	2764358929

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
15	00:06:43	SELECT EC.company_id, C.name, MAX(EC.employee_count) AS highest_employee_count FROM employ...	1 row(s) returned	0.203 sec / 0.000 sec
16	00:11:59	SELECT company_id, title, pay_period, max_salary FROM (2 row(s) returned	0.188 sec / 0.000 sec
17	00:16:29	CREATE PROCEDURE GetRemoteJobs() BEGIN	0 row(s) affected	0.172 sec
18	00:19:01	CALL GetRemoteJobs()	2266 row(s) returned	0.313 sec / 0.234 sec

6. Procedure to insert values to companies and company_specialities table and get the latest count of total specialities with its corresponding company_id.
DELIMITER //

CREATE PROCEDURE ProcessCompanySpecialitiesWithDeduplicateCompanyID(

IN p_company_id BIGINT,
IN p_speciality VARCHAR(256))
BEGIN

INSERT INTO company_specialities (company_id, speciality)
VALUES (p_company_id, p_speciality)
ON DUPLICATE KEY UPDATE speciality = p_speciality;

SELECT company_id, COUNT(*) AS total_specialities
FROM company_specialities
WHERE company_id = p_company_id
GROUP BY company_id;
END //

DELIMITER ;

Calling the function:

```
call ProcessCompanySpecialitiesWithDeduplicateCompanyID(999999999,'ZZZ 22 TEST Speciality');
```

```

1 * use linkedin;
2
3
4 DELIMITER //
5 * CREATE PROCEDURE ProcessCompanySpecialitiesWithDeduplicateCompanyID(
6   IN p_company_id BIGINT,
7   IN p_speciality VARCHAR(256)
8 )
9 *
10 BEGIN
11   INSERT INTO company_specialties (company_id, speciality)
12   VALUES (p_company_id, p_speciality)
13
14   ON DUPLICATE KEY UPDATE speciality = p_speciality;
15
16   SELECT company_id, COUNT(*) AS total_specialties
17   FROM company_specialties
18
19   WHERE company_id = p_company_id
20
21   GROUP BY company_id;
22
23
24
25 END //
26
27
28
29 * /* Calling the function: */
30
31 call ProcessCompanySpecialitiesWithDeduplicateCompanyID(999999999,'ZZZ 22 TEST Speciality');
32
33 * call ProcessCompanySpecialitiesWithDeduplicateCompanyID(999999999,'ZZZ 23 TEST Speciality');
34

```

Action Output: 0

Time	Action	Response	Duration / Fetch Time
1 23:42:08	use linkedin	0 rows affected	0.091 sec
2 02:08:57	CREATE PROCEDURE ProcessCompanySpecialitiesWithDeduplicateCompanyID(_IN p_company_id BIGINT, _IN p_speciality VARCHAR(256)) BEGIN INSERT INTO company_special... 0 rows affected		0.120 sec

7. Procedure to get the exact user desired count as total_specialities_param to produce the company_id, speciality and total_specialities

```

DELIMITER //
CREATE PROCEDURE GetTotalSpecialities(total_specialities_param int)
BEGIN
select * from (
SELECT *, 
COUNT(*) OVER (PARTITION BY company_id) AS total_specialities
FROM company_specialties
order by (COUNT(*) OVER (PARTITION BY company_id)) desc
) A
where total_specialities =total_specialities_param;
      a.
END;
      b.
// DELIMITER ;
Calling the function:
call GetTotalSpecialities(20);

```

```

1 *  use linkedin;
2
3
4  DELIMITER //
5
6 *  CREATE PROCEDURE GetTotalSpecialities(total_specialities_param int)
7
8  BEGIN
9
10  select * from (
11
12    SELECT *,
13      COUNT(*) OVER (PARTITION BY company_id) AS total_specialities
14
15  FROM company_specialities
16
17  order by (COUNT(*)) OVER (PARTITION BY company_id) desc
18
19  ) A
20
21  where total_specialities >total_specialities_param;
22
23
24
25
26  END;
27
28
29
30  // DELIMITER ;

```

The screenshot shows the MySQL Workbench interface. At the top, there is a status bar with '100%' and '16:30'. Below it is a 'Action Output' tab showing the creation of the stored procedure:

Time	Action	Response	Duration / Fetch Time
02-26-17 16:30:45	CREATE PROCEDURE GetTotalSpecialities(total_specialities_param int) BEGIN select * from (SELECT *, COUNT(*) OVER (PARTITION BY company_id) AS total_specialities FROM company_specialities order by (COUNT(*)) OVER (PARTITION BY company_id) desc) A where total_specialities >total_specialities_param;	0 rows(s) affected	0.093 sec

Below this is a 'Result Grid' tab showing the data returned by the stored procedure:

company_id	speciality	total_specialities
79590	accessories	20
79590	accessories	20
79590	advertising	20
79590	bodding	20
79590	consumer products	20
79590	coach	20
79590	e-commerce	20
79590	finance	20
79590	furnishings	20
79590	furniture	20
79590	information techn...	20
79590	logistics	20
79590	machines	20
79590	media	20
79590	merchandising	20
79590	real estate	20
79590	retail	20
79590	sofa	20
79590	supplies / busi...	20
79590	warehousing	20
91305	campaign design	20
91305	communication re...	20
91305	data collection	20
91305	data visualization	20
91305	focus group mod...	20

At the bottom, another 'Action Output' tab shows the execution of the stored procedure:

Time	Action	Response	Duration / Fetch Time
02-26-17 16:31:05	call GetTotalSpecialities(20)	1860 row(s) returned	0.572 sec / 0.085 sec

TRIGGERS:

1. Inserting value in Company activity log table after a deletion of company in companies table

```

DELIMITER $$

CREATE TRIGGER Delete_job_posting_with_company
AFTER DELETE ON Companies FOR EACH ROW
BEGIN
UPDATE job_postings SET Company_id = NULL WHERE company_id = old.company_id;
INSERT INTO Company_Activity_logs VALUES ( old.Company_id, CONCAT('Row has been deleted
from Job_postings for comapny ', old.Company_id));
END $$

DELIMITER ;

```

2 Creating a trigger to log changes to the job_postings table

```
DELIMITER //  
  
CREATE TRIGGER JobPostingsAuditTrigger  
AFTER UPDATE ON job_postings FOR EACH ROW  
  
BEGIN  
  
INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)  
VALUES (OLD.job_id, 'title', OLD.title, NEW.title);  
  
INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)  
VALUES (OLD.job_id, 'work_type', OLD.work_type, NEW.work_type);  
  
INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)  
VALUES (OLD.job_id, 'location', OLD.location, NEW.location);  
  
END;  
  
//  
  
DELIMITER ;  
  
UPDATE job_postings  
  
SET title = 'Sales Manager', work_type ='FULL_TIME', location = 'Santa Clarita'  
WHERE job_id = 133114754;  
  
SELECT * FROM job_postings_audit;
```

Code Worksheet

AWS RDS Connection x

Edit View Query Database Server Tools Scripting Help

gator

EMAS

Query 1 x

```

89 -- Create the job_postings_audit table to store audit records
90 • CREATE TABLE job_postings_audit (
  audit_id INT AUTO_INCREMENT PRIMARY KEY,
  job_id INT,
  updated_column VARCHAR(50),
  old_value VARCHAR(50),
  new_value VARCHAR(50) );

```

Code Worksheet

AWS RDS Connection x

Edit View Query Database Server Tools Scripting Help

gator

EMAS

Query 1 x

```

96 -- Create the trigger to log changes to the job_postings table
97 DELIMITER //
98 • CREATE TRIGGER JobPostingsAuditTrigger
99 AFTER UPDATE ON job_postings FOR EACH ROW
100 BEGIN
101   INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)
102   VALUES (OLD.job_id, 'title', OLD.title, NEW.title);
103   INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)
104   VALUES (OLD.job_id, 'work_type', OLD.work_type, NEW.work_type);
105   INSERT INTO job_postings_audit (job_id, updated_column, old_value, new_value)
106   VALUES (OLD.job_id, 'location', OLD.location, NEW.location);
107 END;
108 //
109 DELIMITER ;
110 • UPDATE job_postings
111 SET title = 'Sales Manager', work_type = 'FULL_TIME', location = 'Santa Clarita'
112 WHERE job_id = 133114754;
113 • SELECT * FROM job_postings_audit;

```

Automat disabled. | manually current c toggle

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid

audit_id	job_id	updated_column	old_value	new_value
2	133114754	title	Sales Manager	Sales Manager
3	133114754	work_type	FULL_TIME	FULL_TIME
4	133114754	location	Santa Clarita, CA	Santa Clarita
*	HULL	HULL	HULL	HULL

job_postings_audit 44 x

Output:

Action Output

#	Time	Action	Message
141	00:44:21	UPDATE job_postings SET title = 'Sales Manager', work_type = 'FULL_TIME', location = 'Santa Clarita' WHERE job_id = 133114754;	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
142	00:44:26	SELECT * FROM job_postings_audit LIMIT 0, 1000	3 row(s) returned

