

SQL-Mongo Project – IBM HR Analytics Employee Attrition & Performance

BUAN 6320

Priyanka Police Reddy Gari

Activity
Prepared Data Model and Created Physical DB
Loaded Data into Database
Wrote SQL Queries
Prepared Mongo Database
Loaded data into Mongo DB
Wrote Mongo Queries
Prepared Report
Reviewed Report

Contents

Relational Data Model	4
Assumptions/Notes About Data Entities and Relationships	4
Entity-Relationship Diagram	7
Physical MySQL Database	8
Screen shot of Physical Database objects	9
SQL Queries.....	15
SQL Query 1.....	15
Question.....	15
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	15
Query	15
Translation	15
Screen Shot of SQL Query and Results.....	16
Result	16
SQL Query 2.....	17
Question.....	17
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	17
Query	17
Translation	17
Screen Shot of SQL Query and Results.....	18
Result	18
SQL Query 3.....	19
Question.....	19
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	19
Query	19
Translation	19
Screen Shot of SQL Query and Results.....	19
Result	20
SQL Query 4.....	20
Question.....	20
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	20
Translation	21
SQL Query.....	21

Screen Shot of SQL Query and Results.....	21
Result	21
SQL Query 5.....	22
Question.....	22
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	22
Query	22
Translation	22
Screen Shot of SQL Query and Results.....	22
SQL Query 6.....	23
Question.....	23
Notes/Comments About SQL Query and Results (Include # of Rows in Result)	23
Query	23
Translation	23
Screen Shot of SQL Query and Results.....	24
Result	24
Data Review for MongoDB	24
Assumptions/Notes About Data Collections, Attributes and Relationships between Collections.....	24
Physical Mongo Database.....	26
Assumptions/Notes About Data Set.....	26
Screen shot of Physical Database objects (Database, Collections and Attributes)	26
Data in the Database	29
MongoDB Queries/Code.....	30
Mongo Query 1	30
Question.....	30
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) ...	30
Translation	30
Query	30
Screen Shot of MongoDB Query/Code and Results	31
Result	31
Mongo Query 2	32
Question.....	32
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) ...	32
Translation	32

Query	32
Screen Shot of MongoDB Query/Code and Results	33
Result	33
Mongo Query 3	34
Question.....	34
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) ...	34
Translation	34
Query	34
Screen Shot of MongoDB Query/Code and Results	35
Result	35
Mongo Query 4	36
Question.....	36
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) ...	36
Translation	36
Query	36
Screen Shot of MongoDB Query/Code and Results	36
Result	37

Relational Data Model

Assumptions/Notes About Data Entities and Relationships

The database is divided into 13 tables. There are many partitioned tables, and a few index tables.

Notes on attributes of data:

We dropped three columns: Over18, EmployeeCount and Standard Hours because they had redundant data, with only one value for each column. So out of the 35 attributes mentioned in the Excel sheet, we have used 32.

Notes on entities:

1. Employee – The entity holds the personal data of employee.
2. Education – Holds the Educational details about employee.
3. Experience – holds the work history about employee.

4. Satisfaction – holds the data of the employee's satisfaction in the various segments.
5. Job details- holds the data about employee and the department, job role and job level they are working in.
6. Salary – Says about the salary earned by employee.
7. Satisfaction_index – holds the various levels of satisfactions
8. Work_life_balance – holds various levels of balance in work life
9. Performance_rating- holds various rating levels of performance.
10. Education_index – holds various levels of education such as college, masters etc.
11. Travel_index- has various options for travel.
12. Job role- has various roles that a company holds.
13. Department- has various departments of the company.

Notes on Relationships:

Employee table are one-to-one related with experience, education, salary, satisfaction, and job details tables. The relationship is made as identifying and mandatory.

The Performance rating index table has indexes for performance ratings, from Low to Outstanding. It has a one-to-many relationship with the experience table. The relationship is non-identifying and optional.

The Work life balance index table has indexes for work-life balance ratings, from Bad to Best. It has a one-to-many relationship with the satisfaction table. The relationship is non-identifying and optional.

The Education index table has indexes for Education names such as college, masters etc., and it has a one-to-many relationship with the Education table. The relationship is non-identifying and optional.

The satisfaction index table has indexes for satisfaction rating names, from Low to Very Bad. It has a one-to-many relationship with the satisfaction table. The satisfaction ID is used as a foreign key for four different attributes: Environment Satisfaction, Relationship Satisfaction, Job Satisfaction and Job Involvement. The relationship is non-identifying and optional.

The Travel index has indexes for Travel names, and it has a one-to-many relationship with the Employee details table. The relationship is non-identifying and optional.

The department table has indexes for department names, and it has a one-to-many relationship with the job details table. The relationship is non-identifying and optional.

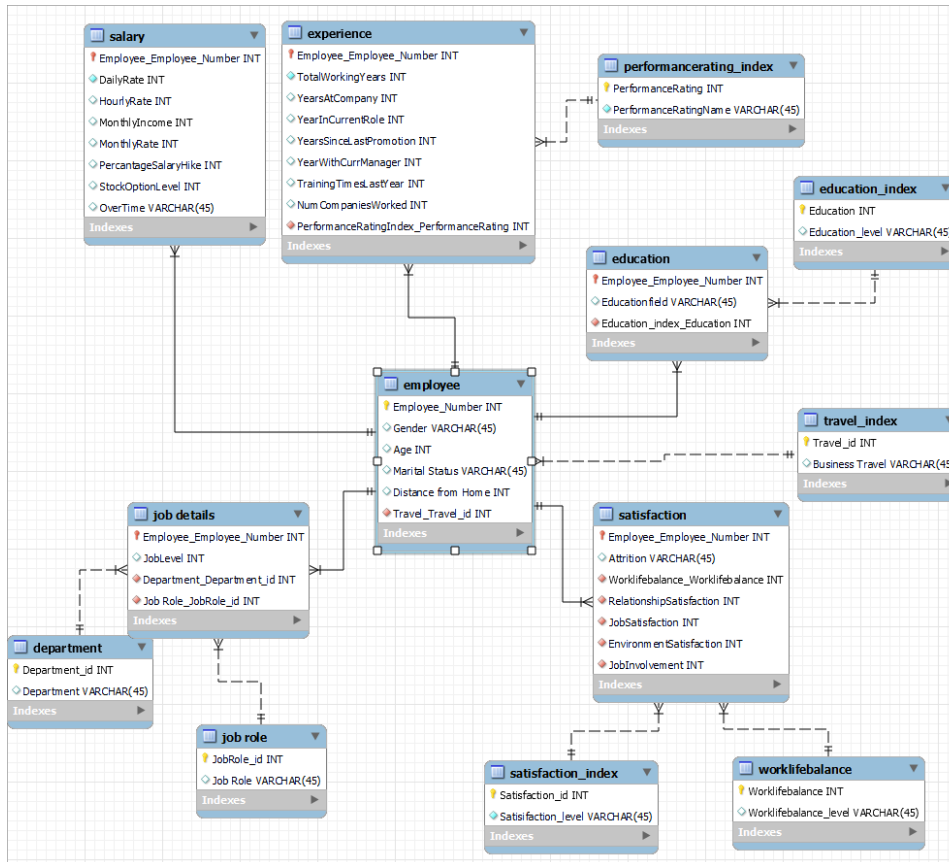
The Job role table has indexes for job roles such as Sales Executive and Research Scientist. It has a one-to-many relationship with the job details table. The relationship is non-identifying and optional.

The data model is in 3NF:

1. All the 13 tables have their primary key.
2. All the other columns in the respective table non-transitively depend on the primary key.

Any column's value can be derived from the primary key.

Entity-Relationship Diagram

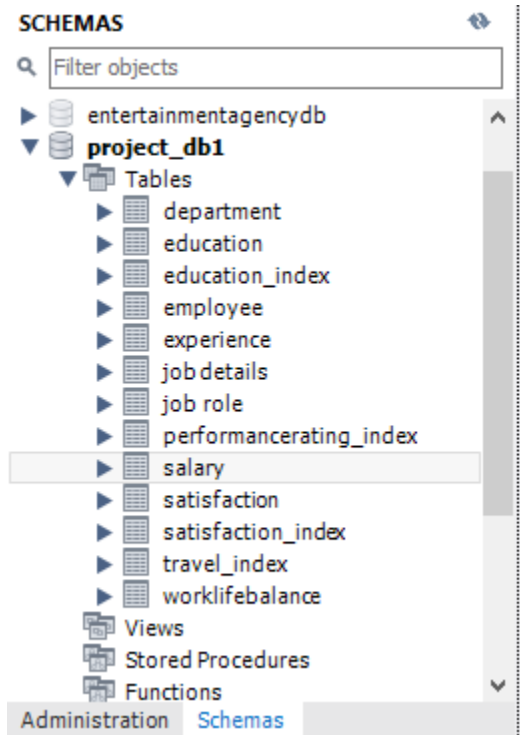


Physical MySQL Database

Assumptions/Notes About Data Set

1. All fields in the database are filled and do not contain any empty values (NULL values).
2. The attributes 'over18', 'employeecount' and 'StandardHours' are not utilized in the model, and their purpose or relevance is unclear.
3. The attributes 'Hourlyrate', 'Dailyrate', 'Monthlyrate', and 'Monthlyincome' are vaguely defined and have little logical connection with each other. They may be considered bad data or other factors affecting these attributes, not accounted for in the database.
4. Attrition refers to the termination of an employee's employment with the company.
5. If a person rehires into the company, they will be assigned a new Employee number and their details will be treated as those of a new employee.
6. All employees receive a monthly income, which is the salary paid to them monthly.
7. Each department has unique job roles that do not overlap with those of other departments.
8. JobLevel and JobRole are not interdependent, meaning that an employee's JobLevel cannot be determined based on their JobRole or vice versa. Instead, both JobLevel and JobRole independently rely on the Employee_Number (primary key) in the database. However, it can be deduced that each JobRole is associated with 4 different Job Levels.
9. Education and EducationField are not mutually dependent, meaning that an employee's Education cannot be determined based on their EducationField or vice versa (No Transitive dependence). Instead, both Education and EducationField independently rely on the Employee_Number (primary key) in the database. However, it can be inferred that each EducationField is associated with 5 different Education categories.
10. Age and over 18: The over 18 age criteria can be easily obtained by a where clause and thus the attribute over18 is removed.

Screen shot of Physical Database objects



Data in the Database

Table Number	Table Name	Primary Key	Foreign Key	# of Rows in Table
1	Employee	Employee_Number	Travel_id	1470
2	Experience	Employee_Number	PerformanceRating	1470
3	Education	Employee_Number	Education	1470
4	Salary	Employee_Number	-	1470
5	Satisfaction	Employee_Number	Worklifebalance RelationshipSatisfaction JobSatisfaction EnvironmentSatisfaction	1470

			JobInvolvement	
6	Job Details	Employee_Number	Department_id JobRole_id	1470
7	PerformanceRating_index	PerformanceRating	-	4
8	Worklifebalance	Worklifebalance	-	4
9	Education_index	Education	-	5
10	Satisfaction_index	Satisfaction_id	-	4
11	Travel_index	Travel_id	-	3
12	Department	Department_id	-	3
13	Job Role	JobRole_id	-	9

1.Employee

```
1 • SELECT count(*) FROM project_db1.employee;
```

<

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

count(*)
1470

2. Experience

```
1 • SELECT count(*) FROM project_db1.experience;
```



The screenshot shows a database query result grid. The top bar includes a back arrow, a 'Result Grid' tab, a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell' button. The table has one column labeled 'count(*)' and one row with the value '1470'.

count(*)
1470

3. Education

```
1 • SELECT count(*) FROM project_db1.education;
```



The screenshot shows a database query result grid. The top bar includes a back arrow, a 'Result Grid' tab, a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell' button. The table has one column labeled 'count(*)' and one row with the value '1470'.

count(*)
1470

4. Salary

```
1 • SELECT count(*) FROM project_db1.salary;
```



The screenshot shows a database query result grid. The top bar includes a back arrow, a 'Result Grid' tab, a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell' button. The table has one column labeled 'count(*)' and one row with the value '1470'.

count(*)
1470

5. Satisfaction

```
1 • SELECT count(*) FROM project_db1.satisfaction;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export:  Wrap Cell
count(*)	
▶	1470

6. Job details

```
1 • SELECT count(*) FROM project_db1.`job details`;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export:  Wrap Cell
count(*)	
▶	1470

7. Performancerating_index

```
1 • SELECT count(*) FROM project_db1.performancerating_index;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export:  Wrap Cell Content: 
count(*)	
▶	4

8. Worklifebalance

```
1 • SELECT count(*) FROM project_db1.worklifebalance;
```

Result Grid		Filter Rows:	Export:	Wrap Cell
	count(*)			
▶	4			

11. Education_index

```
1 • SELECT count(*) FROM project_db1.education_index;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Contents
	count(*)			
▶	5			

12. Satisfaction_index

```
1 • SELECT count(*) FROM project_db1.satisfaction_index;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Contents
	count(*)			
▶	4			

13. Travel_index

```
1 • SELECT count(*) FROM project_db1.travel_index;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export: Wrap Cell
count(*)	
▶ 3	

12. Department

```
1 • SELECT count(*) FROM project_db1.department;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export: Wrap Cell
count(*)	
▶ 3	

13. Job role

```
1 • SELECT count(*) FROM project_db1.`job role`;
```

<	
Result Grid	Filter Rows: <input type="text"/> Export: Wrap Cell
count(*)	
▶ 9	

SQL Queries

Pick 5 out of the 12 statements to write your queries in mySQL.

SQL Query 1

Question

The HR department feels they have the highest job satisfaction while Research & Development department feels their department has the highest environment satisfaction. Who is right? (4th in the List).

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

By calculating the average rating of each department in both job satisfaction and environment satisfaction. The satisfaction index considers a rating of 1 as bad and a rating of 5 as the best. This index applies to any measurement of satisfaction, whether it is job satisfaction or environment satisfaction.

Query

```
select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction",  
avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"
```

```
from department as de
```

```
inner join `job details` as jd on jd.Department_Department_id = de.Department_id
```

```
inner join satisfaction as sa on sa.Employee_Employee_Number =  
jd.Employee_Employee_Number group by department;
```

Translation

Group the data on department, look out the job satisfaction and sort the result in descending order base on the avarage. The Department table in the first row is the department with the highest job satisfaction.

Using the group by on the department, find avrage environment satisfaction and sort the result in descending order based on avrage.the departmenet in the first raw show the department with the highest environment satisfaction.

Select max of environmentsatisfaction and max of jobsatisfaction and case than to decide which department is right and wrong.

Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The SQL Editor contains the following query:

```
1 select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction", avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"
2 from department as de
3 inner join "job details" as jd on jd.Department_Department_Id = de.Department_Id
4 inner join satisfaction as sa on sa.Employee_Employee_Number = jd.Employee_Employee_Number group by department;
```

The Results window shows the output of the query:

department	Average Job Satisfaction	Average Environment Satisfaction
Sales	2.7511	2.6794
Research & Development	2.7583	2.7440
Human Resources	2.6032	2.6825

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	11:13:00	SELECT * FROM project_db1 (jobsat) LIMIT 0, 1000	3 rows returned	0.000 sec / 0.000 sec
2	17:43:24	select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction", avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"	3 rows returned	0.000 sec / 0.000 sec
3	17:45:23	select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction", avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"	3 rows returned	0.016 sec / 0.000 sec
4	17:46:41	select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction", avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"	3 rows returned	0.000 sec / 0.000 sec
5	17:45:49	select de.department, avg(sa.JobSatisfaction) as "Average Job Satisfaction", avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction"	3 rows returned	0.016 sec / 0.000 sec

Result

Research and Development is right here whereas Human Resources is wrong based on the query results.

SQL Query 2

Question

An employee in Sales department has complained to HR saying that females are paid less than males in the company, in all departments. What insight can you provide to prove or disprove that statement?(5th in the list)

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

To prove or disprove that statement we Selects the Departement Name, employee gender, and the average monthly salary of employees within that department, grouping the results by department and gender. It then sorts the results in ascending order by department and gender. The query achieves this by joining the "department," "job details," "employee," and "salary" tables, and calculating the average monthly salary for each group of employees within a department and gender. Based on the result We disprove the statement as we can see from the query results that females are paid more than male in any department.

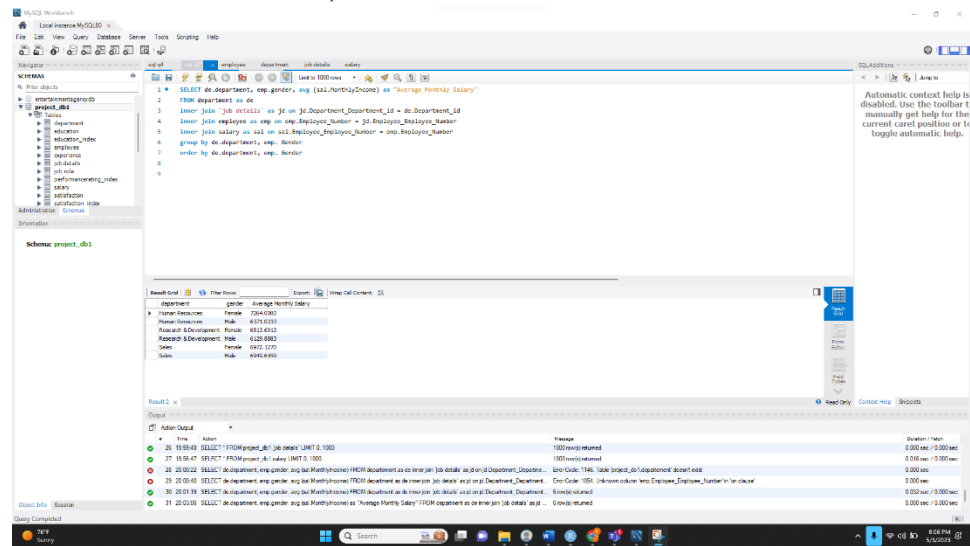
Number of rows in result: 6

Query

```
SELECT de.department, emp.gender, avg (sal.MonthlyIncome) as "Average Monthly Salary"
FROM department as de
inner join `job details` as jd on jd.Department_Department_id = de.Department_id
inner join employee as emp on emp.Employee_Employee_Number = jd.Employee_Employee_Number
inner join salary as sal on sal.Employee_Employee_Number = emp.Employee_Employee_Number
group by de.department, emp. Gender
order by de.department, emp. Gender
```

Translation

Selects the Departement Name, employee gender, and the average monthly salary of employees within that department, grouping the results by department and gender. It then sorts the results in ascending order by department and gender. The query achieves this by joining the "department," "job details," "employee," and "salary" tables, and calculating the average monthly salary for each group of employees within a department and gender.



Result

We disprove the statement as we can see from the query results that females are paid more than male in any department.

SQL Query 3

Question

A press article in a business magazine has said that at this company, married men have higher performance ratings than divorced or single men. What initial finding can you obtain from the data to help articulate the company's response in this regard? (6th in the list)

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

We can group employees by marital status, and compute average performance ratings for Married, Single and Divorced employees. We can then order by average performance rating to get the highest average performance rating first.

Query

```
select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) as "Average Performance" from employee as em
```

```
inner join experience as ex on ex.Employee_Employee_Number = em.Employee_Number  
where em.Gender = "Male"
```

```
group by `Marital Status` order by avg(ex.PerformanceRatingIndex_PerformanceRating) desc;
```

Translation

Select the Marital Status and the average performance index from employees table joined with experience table on employee number, group by marital status and order by average performance index.

Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The SQL Editor window contains the following query:

```
1 select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) as "Average Performance" from employee as em  
2 inner join experience as ex on ex.Employee_Employee_Number = em.Employee_Number where em.Gender = "Male"  
3 group by `Marital Status` order by avg(ex.PerformanceRatingIndex_PerformanceRating) desc;
```

The Results Grid shows the following data:

Marital Status	Average Performance
Single	3.3624
Married	3.1471
Divorced	3.1381

The Output window shows the execution log with the following details:

#	Time	Action	Message	Duration / Fetch
1	19:20:07	select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) as "Average Performance" from employee as em inner join experience	3 rows returned	0.031 sec / 0.000 sec
2	19:22:40	select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) as "Average Performance" from employee as em inner join experience	3 rows returned	0.000 sec / 0.000 sec
3	19:34:03	select `Marital Status`, count(em.Employee_Number) from employee as em inner join experience as ex on ex.Employee_Employee_Number = em.E	3 rows returned	0.000 sec / 0.000 sec
4	19:36:06	select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) from employee as em inner join experience as ex on ex.Employee_E	3 rows returned	0.000 sec / 0.000 sec
5	19:37:26	select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) from employee as em inner join experience as ex on ex.Employee_E	3 rows returned	0.000 sec / 0.000 sec
6	19:41:21	select `Marital Status`, avg(ex.PerformanceRatingIndex_PerformanceRating) as "Average Performance" from employee as em inner join experience	3 rows returned	0.000 sec / 0.000 sec

Result

We see that the Average performance rating of Single men is the highest at 3.1624. Although the other two are not too far, the data still validates the statement is wrong. Thus, **the Single men show better performance than married and divorced.**

SQL Query 4

Question

The company has been paying gas expenses for miles traveled by employees between their home and work. If they want to increase the per mile compensation, which department's employees will gain the most? (8th in the list)

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

It is said that the company decided to increase the compensation per mile. Thus, the department with the highest total distance between home and work will benefit the most as the compensation is directly proportional to the total distance. Thus, we need to find how much total distance is travelled by each department to travel from home to work.

This SQL query selects the department name and the total miles traveled by each department's employees. It then groups the results by department and sorts them in descending order by the total miles traveled.

The query achieves this by joining the "employee," "job details," and "department" tables, and summing the "Distance From Home" column of the "employee" table for each employee within a department.

Here's a breakdown of each part of the query:

`select de.Department, sum(emp.Distance From Home) As "Miles Travelled by Department"`
selects the department name and the total miles traveled by each department's employees.

`from employee as emp inner join 'job details' as jd on emp.Employee_Number =
jd.Employee_Employee_Number inner join department as de on de.Department_id =
jd.Department_Department_id` joins the "employee," "job details," and "department" tables on the appropriate keys to allow the query to access all the necessary data.

`group by de.Department` groups the results by department.

`order by sum(emp.Distance From Home) desc` orders the results in descending order by the total miles traveled.

Translation

Select from Department, sum(emp.`Distance From Home`) As "Miles Travelled by Department" two tables - employee and department - using inner joins. The query calculates the sum distance traveled by employees in each department and orders the results in descending order of distance traveled. Select the max of the output.

SQL Query

select de.Department, sum(emp.`Distance From Home`) As "Miles Travelled by Department" from employee as emp

inner join `job details` as jd on emp.Employee_Number = jd.Employee_Employee_Number

inner join department as de on de.Department_id = jd.Department_Department_id

group by de.Department order by sum(emp.`Distance From Home`) desc;

Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The SQL editor contains the following query:

```
1. select de.Department, sum(emp.`Distance From Home`) As "Miles Travelled by Department" from employee as emp
2. inner join `job details` as jd on emp.Employee_Number = jd.Employee_Employee_Number
3. inner join department as de on de.Department_id = jd.Department_Department_id
4. group by de.Department order by sum(emp.`Distance From Home`) desc;
```

The Results grid shows the output of the query:

Department	Miles Travelled by Department
Research & Development	8788
Sales	6177
Human Resources	510

The Output tab shows the execution details of the query, including the time taken and the number of rows returned.

Result

The Query says that the R&D department employees top the criteria by travelling a total distance of

8788 miles. Thus, with the decision to increase the per mile compensation, the **Research & Development** department gains the most.

SQL Query 5

Question

A new employee from the Life Sciences education field wants to work in Sales. Do you believe the company might be able to give him a chance to work in Sales? Why or why not? (9th in the list)

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

To check if the company gives a chance to a new candidate from Life Science, the best way is to see the percentage of employees that are already working in the department who are from Life Science. Thus, the following queries are used to find out the percentage of candidates in sales, and to see how impactful the selection of life sciences is among the different departments.

Query

```
select de.Department_Department_id, dep.Department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life Sciences in each department" from education as eh inner join `job details` as de on eh.Employee_Employee_Number = de.Employee_Employee_Number inner join department as dep on dep.Department_id = de.Department_Department_id where eh.Educationfield = "Life Sciences" group by de.Department_Department_id;
```

Number of rows in result: 3

Translation

Select department id and department, and percentage of life science educated employees in each department, computed by getting the count of life science educated employees from each department, and dividing it by total count. Data is retrieved from education table inner joined with job details on employee number, inner joined with department on department id. Query is filtered for where education field is life sciences. Query is grouped by department.

Screen Shot of SQL Query and Results

The screenshot shows a SQL IDE interface. The query editor contains the following SQL query:

```
select de.Department_Department_id, dep.Department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life Sciences in each department" from education as eh inner join `job details` as de on eh.Employee_Employee_Number = de.Employee_Employee_Number inner join department as dep on dep.Department_id = de.Department_Department_id where eh.Educationfield = "Life Sciences" group by de.Department_Department_id;
```

The results pane shows the following data:

Department_Department_id	Department	Percentage of Life Sciences in each department
1	Sales	24.7623%
2	Research & Development	72.4075%
3	Human Resources	2.4493%

The bottom pane shows the execution log with the following messages:

- 70 171530 select de.department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life S..." Error Code: 1054. Unknown column 'de.department' in 'field list' 0.000 sec
- 71 171530 select de.department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life S..." 0.010 sec / 0.000 sec
- 72 171724 select de.department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life S..." Error Code: 1054. Unknown column 'de.department' in 'field list' 0.000 sec
- 73 171747 select de.department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life S..." 0.000 sec / 0.000 sec
- 74 171840 SELECT COUNT(*) FROM education WHERE Educationfield = "Life Sciences" 0.000 sec / 0.000 sec
- 75 172201 select de.department, concat(count(*)/ (select count(*) from education where Educationfield = "Life Sciences") *100, "%") as "Percentage of Life S..." 0.000 sec / 0.000 sec

Result

We see that employees with Life Sciences education background is in Sales (24.7525%) in the company and is the second department who recruits from the same education level after the R&D. Thus, we conclude that **there are good chances for a new employee from Life sciences background to get into Sales Department.**

SQL Query 6

Question

HR feels that their environment satisfaction score is higher than Sales, but HR job satisfaction score is lower than Research & Development. Are they right? (10th in the List).

Notes/Comments About SQL Query and Results (Include # of Rows in Result)

By calculating the average rating of each department in both environment satisfaction and job satisfaction. The satisfaction index considers a rating of 1 as bad and a rating of 5 as the best. This index applies to any measurement of satisfaction, whether it is job satisfaction or environment satisfaction.

Query

```
select de.department, avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction",  
avg(sa.JobSatisfaction) as "Average Job Satisfaction"
```

```
from department as de
```

```
inner join `job details` as jd on jd.Department_Department_id = de.Department_id
```

```
inner join satisfaction as sa on sa.Employee_Employee_Number =  
jd.Employee_Employee_Number group by department;
```

Translation

Group the data on department, look out the job satisfaction and sort the result in descending order base on the avarage. The Department table in the first row is the department with the highest job satisfaction.

Using the group by on the department, find avrage environment satisfaction and sort the result in descending order based on avrage.the departmenet in the first raw show the department with the highest environment satisfaction.

Screen Shot of SQL Query and Results

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
1 select de.department, avg(sa.EnvironmentSatisfaction) as "Average Environment Satisfaction", avg(sa.JobSatisfaction) as "Average Job Satisfaction"
2 from department as de
3 inner join "job details" as jd on jd.Department_Department_Id = de.Department_Id
4 inner join satisfaction as sa on sa.Employee_Employee_Number = jd.Employee_Employee_Number group by department;
5
```

The Results window displays the following data:

department	Average Environment Satisfaction	Average Job Satisfaction
Sales	2.6794	2.7511
Research & Development	2.7440	2.7263
Human Resources	2.6825	2.6032

The Action Output window shows the execution progress of the query, including the time taken and the number of rows returned.

Result

HR feels that their environment satisfaction score is higher than Sales which is right as HRs average environment score is 2.6825 which is greater than sales environment satisfaction which is 2.6794 but HR job satisfaction score is lower than Research & Development as HRs average job satisfaction 2.6032 is lower than average job satisfaction of research and development is 2.7263. **Thus, they are right.**

Data Review for MongoDB

Assumptions/Notes About Data Collections, Attributes and Relationships between Collections

Data is loaded into Mongo using Compass and visualized using Compass and Mongo Shell.

The raw dataset is loaded into Mongo Server because it is capable of handling unstructured data too. This is now also used to compare result sets fetched from Mongo with SQL result sets. The collection for the raw dataset is named IBM.

Four index tables are added as collections to the database, to provide indexes for values used in the raw data collection IBM.

The satisfaction_id collection a common index for several columns in the IBM collection, such as EnvironmentSatisfaction, JobInvolvement, JobSatisfaction, PerformanceRating, and RelationshipSatisfaction.

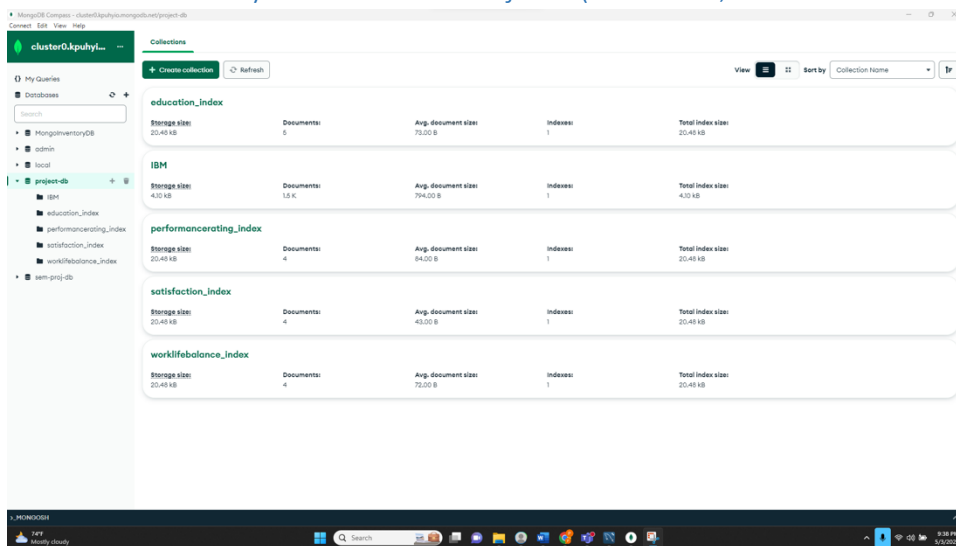
Execution of queries is done through the Mongo Shell.

Physical Mongo Database

Assumptions/Notes About Data Set

1. The dataset which is originally in csv format is loaded into the mongoDB with the database name as project and collection named as IBM.
2. No normalization: This holds all the attributes , unlike SQL.
3. The database still has indexes for certain fields as four separate collections, which are there for reference reasons. This makes a total of 5 collections in the database.
4. All the Collections are present as reference collections instead of embedded collections as there is no use of the collection other than IBM to solve the problems of this project.

Screen shot of Physical Database objects (Database, Collections and Attributes)



MongoDB Compass - cluster0.mongodb.net/project-db-IBM

cluster0.kpuhyt...

Documents project-db-IBM

1.5k DOCUMENTS 1 INDEXES

Filter Type a query: { field: "value" } [Reset] [Find] [More Options]

ADD DATA EXPORT COLLECTION

1 - 20 of 1470

```
{
  "_id": "ObjectID('64531a2a7f3aedd0b062a0c')",
  "Age": 41,
  "Activity": "yes",
  "BusinessTravel": "Travel_Rarely",
  "DailyRate": 1100,
  "Department": "Sales",
  "DistanceFromHome": 1,
  "Education": 2,
  "EducationField": "Life Sciences",
  "EmployeeCount": 1,
  "EmployeeNumber": 1,
  "EnvironmentSatisfaction": 2,
  "Gender": "Female",
  "HourlyRate": 94,
  "JobInvolvement": 3,
  "JobLevel": 2,
  "JobRole": "Sales_Executive",
  "JobSatisfaction": 4,
  "MaritalStatus": "Single",
  "MonthlyIncome": 10900,
  "MonthlyRate": 10470,
  "NewContractHired": 0,
  "Over18": "yes",
  "OverTime": "yes",
  "PercentSalaryHike": 11,
  "Education_Index": 1,
  "PerformanceRating_Index": 1,
  "Satisfaction_Index": 1,
  "WorkLifeBalance_Index": 1
}
```

SHOW 11 MORE FIELDS

```
{
  "_id": "ObjectID('64531a2a7f3aedd0b062a0c')",
  "Age": 49,
  "Activity": "no",
  "BusinessTravel": "Travel_Frequently",
  "DailyRate": 270,
  "Department": "Research & Development",
  "DistanceFromHome": 8,
  "Education": 1,
  "Education_Index": 1,
  "PerformanceRating_Index": 1,
  "Satisfaction_Index": 1,
  "WorkLifeBalance_Index": 1
}
```

MongoDB Compass - cluster0.mongodb.net/project-db-education_index

cluster0.kpuhyt...

Documents project-db-education_index

5 DOCUMENTS 1 INDEXES

Filter Type a query: { field: "value" } [Reset] [Find] [More Options]

ADD DATA EXPORT COLLECTION

1 - 5 of 5

```
{
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "Education_Index": 1,
  "Education_Level": "Below College"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "Education_Index": 2,
  "Education_Level": "College"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "Education_Index": 3,
  "Education_Level": "Bachelor"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "Education_Index": 4,
  "Education_Level": "Master"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "Education_Index": 5,
  "Education_Level": "Doctor"
}
```

MongoDB Compass - cluster0.mongodb.net/project-db-performance_rating_index

cluster0.kpuhyt...

Documents project-db-performance_rating_index

4 DOCUMENTS 1 INDEXES

Filter Type a query: { field: "value" } [Reset] [Find] [More Options]

ADD DATA EXPORT COLLECTION

1 - 4 of 4

```
{
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "PerformanceRating_Index": 1,
  "PerformanceRatingName": "Low"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "PerformanceRating_Index": 2,
  "PerformanceRatingName": "Good"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "PerformanceRating_Index": 3,
  "PerformanceRatingName": "Excellent"
}, {
  "_id": "ObjectID('64531b037f3aedd0b062a0c')",
  "PerformanceRating_Index": 4,
  "PerformanceRatingName": "Outstanding"
}
```

MongoDB Compass - cluster0.kpuhyt.mongodb.net/project-db.satisfaction_index

Connect Edit View Collection Help

cluster0.kpuhyt... Documents project-db.satisf...

My Queries Databases

Search

- MongoInventoryDB
- admin
- local
- project-db
 - ibm
 - education_index
 - performance_rating_index
 - satisfaction_index
 - worklifebalance_index
- iam-proj-db

project-db.satisfaction_index

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: "value" } [Reset] [Find] [More Options]

ADD DATA EXPORT COLLECTION 1 - 4 of 4

_id: ObjectId("64515d97f3aedd0802a7")	Satisfaction_id: 1
_id: ObjectId("64515d97f3aedd0802a8")	Satisfaction_id: 2
_id: ObjectId("64515d97f3aedd0802a9")	Satisfaction_id: 3
_id: ObjectId("64515d97f3aedd0802aa")	Satisfaction_id: 4



MongoDB Compass - cluster0.kpuhyt.mongodb.net/project-db.worklifebalance_index

Connect Edit View Collection Help

cluster0.kpuhyt... Documents project-db.worklf...

My Queries Databases

Search

- MongoInventoryDB
- admin
- local
- project-db
 - ibm
 - education_index
 - performance_rating_index
 - satisfaction_index
 - worklifebalance_index
- iam-proj-db

project-db.worklifebalance_index

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: "value" } [Reset] [Find] [More Options]

ADD DATA EXPORT COLLECTION 1 - 4 of 4

_id: ObjectId("64515f7ff3aedd0802a4")	Worklifebalance_id: 1	Worklifebalance: "Bad"
_id: ObjectId("64515f7ff3aedd0802a5")	Worklifebalance_id: 2	Worklifebalance: "Good"
_id: ObjectId("64515f7ff3aedd0802a6")	Worklifebalance_id: 3	Worklifebalance: "Better"
_id: ObjectId("64515f7ff3aedd0802a7")	Worklifebalance_id: 4	Worklifebalance: "Best"



```
x_MONGOSH
> use project-db
< 'already on db project-db'
> show collections
education_index
ibm
performance_rating_index
satisfaction_index
worklifebalance_index
Atlas atlas-rv2ix-shard-0 [armory] project-db
```

Data in the Database

Collection Name	Relationships With Other Collections (if any)	# of Documents in Collection
IBM		1470
worklifebalance_index	Index for Worklifebalance in IBM	4
education_index	Index for EducationLevel in IBM	5
satisfaction_index	Index for EnvironmentSatisfaction, JobInvolvement, JobSatisfaction, PerformanceRating, and RelationshipSatisfaction in IBM	4
performancerating_index	Index for PerformanceRating in IBM	4

MongoDB Queries/Code

Pick 3 queries and write them in MongoDB

Mongo Query 1

Question

Assume that the company has several branch offices around the country and employee morale is down. The company feels that not enough employees travel frequently between offices and that more employees should travel frequently to the branch offices to improve morale amongst each other. Is the company correct in feeling this way? (1st in the list)

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

For the travel choices of the employees, we can extract the travel choice and the number of employees bound to it out of the total number of employees in the company.

of Documents in Result: 3

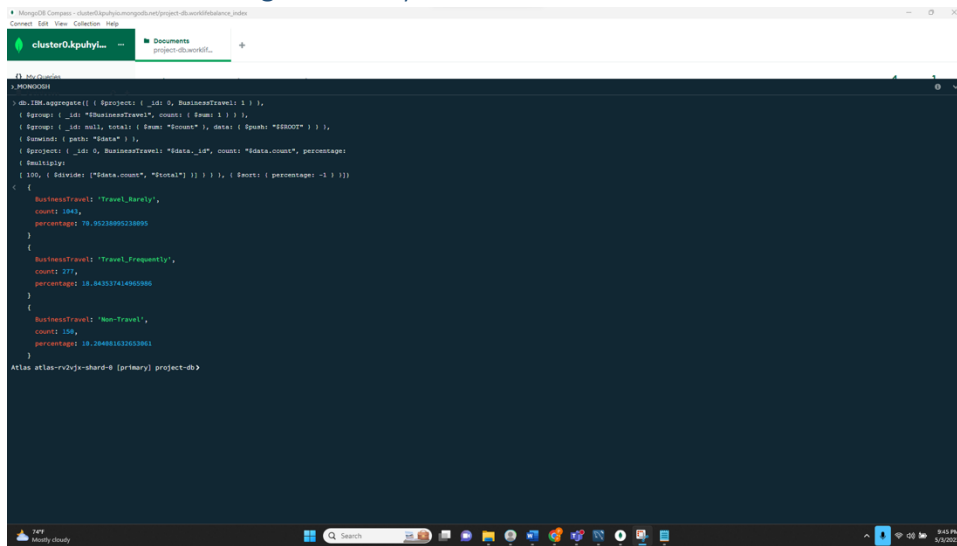
Translation

Group the data by BusinessTravel and find the number of employees travel behavior and the percentage contribution.

Query

```
db.IBM.aggregate([ { $project: { _id: 0, BusinessTravel: 1 } }, { $group: { _id: "$BusinessTravel",  
count: { $sum: 1 } } }, { $group: { _id: null, total: { $sum: "$count" }, data: { $push: "$$ROOT" } }  
}, { $unwind: { path: "$data" } }, { $project: { _id: 0, BusinessTravel: "$data._id", count:  
"$data.count", percentage: { $multiply:  
[ 100, { $divide: ["$data.count", "$total"] } ] } } }, { $sort: { percentage: -1 } } ])
```

Screen Shot of MongoDB Query/Code and Results



The screenshot shows the MongoDB Shell interface with a query and its results. The query is an aggregation pipeline that groups data by business travel frequency and calculates the percentage of employees in each group. The results are displayed as JSON documents.

```
> use project
> db.employees.aggregate([
  { $project: { _id: 0, BusinessTravel: 1 } },
  { $group: { _id: "BusinessTravel", count: { $sum: 1 } } },
  { $group: { _id: null, count: { $sum: "$count" }, data: { $push: "$_id" } } },
  { $unwind: { path: "$data" } },
  { $project: { _id: 0, BusinessTravel: "$data._id", count: "$data.count", percentage:
    { $multiply:
      [ 100, { $divide: ["$data.count", "$total"] } ] }, { $sort: { percentage: -1 } } } }
])
```

```
{
  "BusinessTravel": "Travel_Rarely",
  "count": 1843,
  "percentage": 70.95238095238095
},
{
  "BusinessTravel": "Travel_Frequently",
  "count": 277,
  "percentage": 10.843750000000001
},
{
  "BusinessTravel": "Non-Travel",
  "count": 104,
  "percentage": 10.204081632653061
}
```

Atlas atlas-rv2rjs-shard-0 [primary] project-db

Result

18.84% of employees travel frequently whereas 70.95238% of the employees travel Rarely and 10.204% don't travel. Thus, **the company about the employees travel choice is right.**

Mongo Query 2

Question

A new employee from a Medical-related education field wants to work in Sales. Do you believe the company might be able to give her a chance to work in Sales? Why or why not?(3rd in the list)

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

To determine whether Medical-related education field is given importance for staff members working in sales, we can group the IBM collection by Department of employees, filter for the employees that have a Medical related education and calculate the average salary for all employees. We can use the aggregation framework to accomplish this.

From the results of this query, we can see that medically educated staff have the highest average salary by department in the sales department, compared to Human Resources and Research & Development. We can safely say that the employee from a medical-related education field will do well in the sales department.

of Documents in Result: 3

Translation

Using the aggregate pipeline, group IBM collection by Department, filter by EducationField as "Medical", and get the AvgMonthlyIncome for each Department.

Query

```
db.IBM.aggregate([{$group: {"_id": {Department:"$Department",  
EducationField:"Medical"},AvgMonthlyIncome: {$avg: "$MonthlyIncome"}}}])
```

```

Last login: Fri May 5 20:24:18 on ttys001
(base) viyangshah@Viyangshah-MacBook-Pro ~ % mongosh "mongodb+srv://cluster0.kpuhyio.mongodb.net" --apiVersion 1 --username viyangshah
Enter password: *****
Current Mongosh Log ID: 64558ac19a46c8e90cd51447
Connecting to:      mongodb+srv://<credentials>@cluster0.kpuhyio.mongodb.net/?appName=mongosh+1.6.2
Using MongoDB:      6.0.5 (API Version 1)
Using Mongosh:      1.6.2

For mongosh info see: https://docs.mongodb.com/mongosh-shell/

Atlas atlas-rv2vjx-shard-0 (primary) test> db.IBM.aggregate([{$group: {"_id": {"Department": "$Department", EducationField: "Medical"}, AvgMonthlyIncome: {$avg: "$MonthlyIncome"}}}])
{
  "_id": { Department: 'Human Resources', EducationField: 'Medical' },
  AvgMonthlyIncome: 6654.507936507936
},
{
  "_id": { Department: 'Sales', EducationField: 'Medical' },
  AvgMonthlyIncome: 6959.17264573991
},
{
  "_id": { Department: 'Research & Development', EducationField: 'Medical' },
  AvgMonthlyIncome: 6281.252861602497
}
Atlas atlas-rv2vjx-shard-0 (primary) project-db>

```

For the Human Resources department, medical-educated staff have an average monthly salary of \$6654.50. The Sales department has \$6959.17, and the Research & Development department has \$6281.25. The Sales department has the highest pay to staff with medical education compared to all other departments, so it is considerable that a new employee with a medical-related education background be put in Sales.

For the Human Resources department, medical-educated staff have an average monthly salary of \$6654.50. The Sales department has \$6959.17, and the Research & Development department has \$6281.25. The Sales department has the highest pay to staff with medical education compared to all other departments, so it is considerable that a new employee with a medical-related education background be put in Sales.

Mongo Query 3

Question

If the company wants to cut travel costs, which department should the company focus on? (7th in the list)

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

The company aims to reduce travel costs which can occur due to frequent travel by the same employee or rare travel by many employees. To identify these departments with high travel expenses, the query determines the number of employees who travel frequently and rarely in each department.

of Documents in Result: 3, 3

Translation

Compare the data with the attribute business travel as travel frequently group the resultant data and count the number of employees who travel oftenly in each department. Output in descending order based on the count.

Compare the data with the attribute business travel as travel rarely, group the resultant data and count number of employees who travel rarely in each department. Output in descending order based on the count.

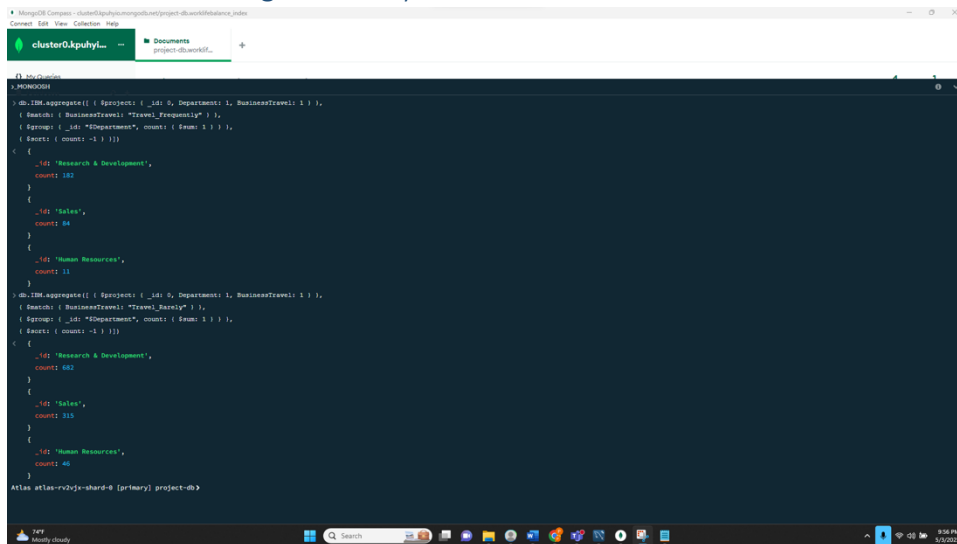
Decide which Department the company needs to focus on with respect to travel costs.

Query

```
db.IBM.aggregate([ { $project: { _id: 0, Department: 1, BusinessTravel: 1 } }, { $match: { BusinessTravel: "Travel_Frequently" } }, { $group: { _id: "$Department", count: { $sum: 1 } } }, { $sort: { count: -1 } } ] )
```

```
db.IBM.aggregate([ { $project: { _id: 0, Department: 1, BusinessTravel: 1 } }, { $match: { BusinessTravel: "Travel_Rarely" } }, { $group: { _id: "$Department", count: { $sum: 1 } } }, { $sort: { count: -1 } } ] )
```

Screen Shot of MongoDB Query/Code and Results



```
> use project
> db.employees.aggregate([
  { $project: { _id: 0, Department: 1, BusinessTravel: 1 } },
  { $match: { BusinessTravel: "Travel Frequently" } },
  { $group: { _id: "$Department", count: { $sum: 1 } } },
  { $sort: { count: -1 } }
])
{
  "_id": "Research & Development",
  "count": 182
}
{
  "_id": "Sales",
  "count": 84
}
{
  "_id": "Human Resources",
  "count": 11
}
> db.employees.aggregate([
  { $project: { _id: 0, Department: 1, BusinessTravel: 1 } },
  { $match: { BusinessTravel: "Travel Rarely" } },
  { $group: { _id: "$Department", count: { $sum: 1 } } },
  { $sort: { count: -1 } }
])
{
  "_id": "Research & Development",
  "count": 682
}
{
  "_id": "Sales",
  "count": 318
}
{
  "_id": "Human Resources",
  "count": 49
}
Atlas atlas-rv2xj-shard-0 [primary] project-db>
```

Result

The Research and Development Department has more employees who travel in both areas, with more than double the number of employees compared to the next department. Therefore, the Research and Development department is the department the company must focus on if it wants to reduce travel costs.

Mongo Query 4

Question

A press article in a business magazine has said that at this company, single women in Sales have worked at the company longer than divorced or married women. What initial finding can you obtain from the data to help articulate the company's response in this regard? (12th in the list)

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

We are going to find the avg yearsatcompany of married women, divorced women, and single women separately to help articulate the company's response. We can observe that divorced or married women have worked at the company longer than single women in Sales.

of Documents in Result: 3

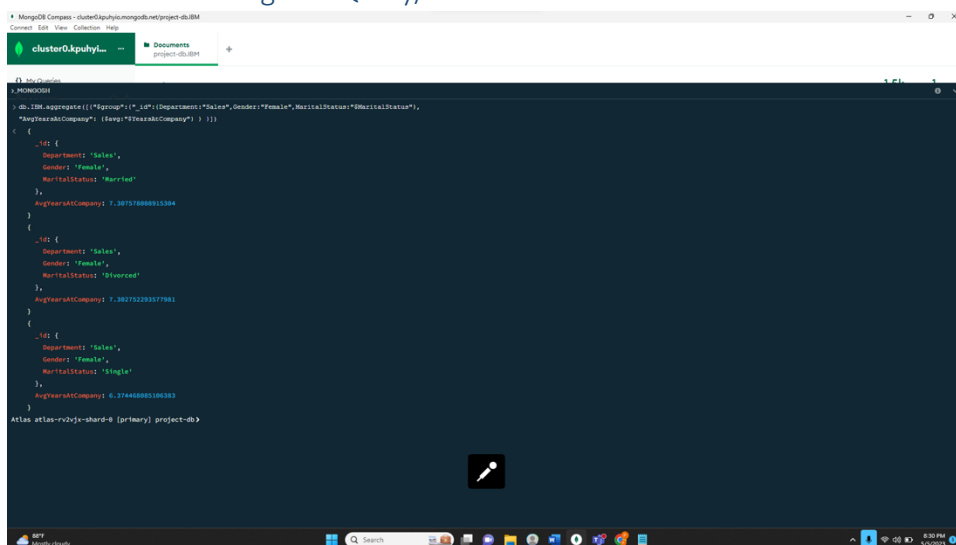
Translation

Using IBM collection utilize aggregation framework and group the documents by MaritalStatus where employees are female and from Sales department and calculate the avg yearsatcompany.

Query

```
db.IBM.aggregate([{"$group":{"_id":{"Department":"Sales",Gender:"Female",MaritalStatus:"$MaritalStatus"},"AvgYearsAtCompany":{"$avg":"$YearsAtCompany"}}}])
```

Screen Shot of MongoDB Query/Code and Results



The screenshot shows the MongoDB Compass interface. The query editor contains the following aggregation pipeline:

```
> db.IBM.aggregate([{"$group":{"_id":{"Department":"Sales",Gender:"Female",MaritalStatus:"$MaritalStatus"},"AvgYearsAtCompany":{"$avg":"$YearsAtCompany"}}}])
```

The results pane shows three documents, each representing a group of employees by marital status:

Department	Gender	MaritalStatus	AvgYearsAtCompany
Sales	Female	Married	7.88757888881384
Sales	Female	Divorced	7.88273228257781
Sales	Female	Single	6.87448888108383

Result

The press article is wrong as per the results married women worked longer in the Sales department than both divorced and single women.