

# Project 01: Employee Management System

## Database Design:

### Table: Employees

PropertiesDataER DiagramlocalhostDatabasesempTablesemployees

Table Name:employeesPartitioned

Engine:InnoDB

Auto Increment:0

Charset:utf8mb4

Collation:utf8mb4\_0900\_ai\_ci

Description:

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
Employee ID	1	int	[v]	[ ]	PRI				
First Name	2	varchar(100)	[v]	[ ]					
Last Name	3	varchar(100)	[v]	[ ]					
Department	4	varchar(100)	[v]	[ ]	MUL				
Job Title	5	varchar(100)	[v]	[ ]					
salary	6	int	[v]	[ ]					
Date of Joining	7	date	[v]	[ ]					

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

PropertiesDataER DiagramlocalhostDatabasesempTablesemployees

employeesEnter a SQL expression to filter results (use Ctrl+Space)

Grid

	Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
1	1	Arman	Uddin	D001	Software Engineer	70,000	2020-
2	2	Meharab	Hossain	D002	Project Manager	85,000	2019-
3	3	Anwar	Hossain	D003	Data Analyst	60,000	2020-
4	4	Walid	Dipto	D004	HR Specialist	55,000	2022-
5	5	Toukir	Ahmed	D005	Marketing Manager	78,000	2018-
6	6	Nazmul	Haque	D006	Sales Associate	45,000	2023-
7	7	Imam	Sourav	D007	Operations Manager	90,000	2017-
8	8	Jahidul	Islam	D008	System Analyst	65,000	2024-
9	9	Tamim	Ahmed	D009	UX Designer	70,000	2022-
10	10	Rakibuzzaman	Khan	D010	Financial Analyst	72,000	2023-
11	11	Masud	Ahmed	D011	Software Developer	75,000	2020-
12	12	Akik	Khan	D012	Brand Strategist	83,000	2019-
13	13	Sahedur	Rahman	D013	Recruiter	58,000	2022-
14	14	Nasif	Nahian	D014	Account Manager	67,000	2020-
15	15	Raiyan	Ahmed	D015	QA Engineer	62,000	2020-
16	16	Rezwana	Anjum	D016	Content Writer	54,000	2019-
17	17	Mosrur	Sunny	D017	Compliance Officer	71,000	2020-
18	18	Nahid	Karim	D018	Business Analyst	80,000	2022-
19	19	Talha	Howlader	D019	Web Developer	69,000	2020-
20	20	Towhidul	Islam	D020	Graphic Designer	64,000	2020-

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

Table: Departments

PropertiesDataER DiagramlocalhostDatabasesempTablesdepartments

Table Name:departmentsPartitioned

Engine:InnoDB

Auto Increment:0

Charset:utf8mb4

Collation:utf8mb4\_0900\_ai\_ci

Description:

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Comment
A-Z Department	1	varchar(100)	[v]	[ ]	PRI				
A-Z Department	2	varchar(100)	[v]	[ ]					

ConstraintsForeign KeysReferencesTriggersIndexesPartitionsStatisticsDDLVirtual

RefreshSave ...Revert

2 items

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

PropertiesDataER DiagramlocalhostDatabasesempTablesdepartments

departmentsEnter a SQL expression to filter results (use Ctrl+Space)

Grid

	A-Z Department ID	A-Z Department Name
1	D001	Developer
2	D002	Project Management
3	D003	Sales & Marketing
4	D004	Human Resources
5	D005	Sales & Marketing
6	D006	Sales & Marketing
7	D007	Operations
8	D008	Technical
9	D009	Developer
10	D010	Audit
11	D011	Developer
12	D012	Sales & Marketing
13	D013	Human Resources
14	D014	Customer Service
15	D015	Software Development
16	D016	Digital Marketing
17	D017	Human Resources
18	D018	Marketing & Strategies
19	D019	Software Development
20	D020	Designer

Value X  
D001

RefreshSave ...Cancel

Export data10020

20 row(s) fetched - 0.001s, on 2024-10-12 at 19:45:30

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

Table: Salaries

PropertiesDataER DiagramlocalhostDatabasesempTables salaries

Table Name: salariesPartitioned

Engine: InnoDB

Auto Increment: 0

Charset: utf8mb4

Collation: utf8mb4\_0900\_ai\_ci

Description:

	Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Commer
Columns	123 Employee ID	1	int	[v]	[ ]	PRI				
Constraints	123 salary	2	int	[v]	[ ]					
Foreign Keys	Date of Salary Update	3	date	[ ]	[ ]					
References										
Triggers										
Indexes										
Partitions										
Statistics										
DDL										
Virtual										

RefreshSave ...Revert

3 items

Database Tasks - GeneralTasks: type a part of task name hereTask executions: type a part of error message

BDT en

PropertiesDataER DiagramlocalhostDatabasesempTables salaries

salariesEnter a SQL expression to filter results (use Ctrl+Space)

	123 Employee ID	123 salary	Date of Salary Update
1	1	81,000	2021-01-05
2	2	85,000	2020-01-02
3	3	60,000	2021-01-03
4	4	55,000	2023-02-10
5	5	78,000	2019-02-06
6	6	45,000	2024-01-08
7	7	90,000	2018-12-12
8	8	65,000	2025-08-25
9	9	70,000	2023-05-30
10	10	72,000	2024-05-20
11	11	75,000	2022-08-25
12	12	83,000	2021-12-20
13	13	58,000	2023-12-22
14	14	67,000	2022-04-25
15	15	62,000	2021-10-25
16	16	54,000	2020-04-22
17	17	71,000	2022-04-04
18	18	80,000	2024-08-10
19	19	69,000	2022-07-14
20	20	64,000	2021-10-01

RefreshSave ...Cancel

Export data10020

20 row(s) fetched - 0.002s (0.001s fetch), on 2024-10-12 at 19:45:26

Database Tasks - GeneralTasks: type a part of task name hereTask executions: type a part of error message

BDT en

Value X

Edit value: 1

Dictionary (employee)

Type part of dictio

Value	Descri
1	Softw
2	Project
3	Data A
4	HR Spe
5	Market
6	Sales A
7	Operat
8	Systen
9	UX Des
10	Financ
11	Softwa
12	Brand

Table: Performance\_reviews

PropertiesDataER Diagram

localhostDatabasesempTablesperformance\_reviews

Table Name:performance\_reviews

Engine:InnoDB

Auto Increment:0

Charset:utf8mb4

Collation:utf8mb4\_0900\_ai\_ci

Description:

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Commer
123 Review ID	1	int	[v]	[ ]	PRI				
123 Employee I	2	int	[v]	[ ]	MUL				
A-Z Review Sco	3	varchar(100)	[v]	[ ]					
Review Dat	4	date	[v]	[ ]					

RefreshSave ...Revert

Review ID

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

PropertiesDataER Diagram

localhostDatabasesempTablesperformance\_reviews

Table Name:performance\_reviews

Engine:InnoDB

Auto Increment:0

Charset:utf8mb4

Collation:utf8mb4\_0900\_ai\_ci

Description:

Columns

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Extra	Expression	Commer
123 Review ID	1	int	[v]	[ ]	PRI				
123 Employee I	2	int	[v]	[ ]	MUL				
A-Z Review Sco	3	varchar(100)	[v]	[ ]					
Review Dat	4	date	[v]	[ ]					

RefreshSave ...Revert

Review ID

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

## Basic Queries:

--Retrieve all employees' data using SELECT \*--

```
SELECT * FROM emp.employees e ;
```

The screenshot displays a database management interface. At the top, a SQL query editor contains the command: `SELECT * FROM emp.employees e ;`. Below the editor, a table titled "employees 1" shows the results of the query. The table has 8 columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Joining. The data is as follows:

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
1	Arman	Uddin	D001	Software Engineer	70,000	2020-
2	Meharab	Hossain	D002	Project Manager	85,000	2019-
3	Anwar	Hossain	D003	Data Analyst	60,000	2020-
4	Walid	Dipto	D004	HR Specialist	55,000	2022-
5	Toukir	Ahmed	D005	Marketing Manager	78,000	2018-
6	Nazmul	Haque	D006	Sales Associate	45,000	2023-
7	Imam	Sourav	D007	Operations Manager	90,000	2017-

Below the table, there is a toolbar with options like Refresh, Save, Cancel, and Export data. At the bottom, a "Database Tasks - General" tab is visible, showing a search bar for tasks and a status bar with "BDT en Writable Smart Insert 1:32:31 Sel: 0 | 0".

--Filter employees by department or job title using the WHERE clause--

```
select *  
from emp.employees e  
where `Job Title` = 'Data Analyst'  
;
```

The screenshot shows a database management tool interface. At the top, a SQL query is entered in a text area:

```
SELECT * FROM emp.employees e ;  
select *  
from emp.employees e  
where `Job Title` = 'Data Analyst'  
;
```

Below the query editor, the results are displayed in a table grid. The table has 8 columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Joining. The first row of data shows an employee with ID 123, named Anwar Hossain, working in Department D003 as a Data Analyst, with a salary of 60,000 and a joining date of 2020-01-3.

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
123	Anwar	Hossain	D003	Data Analyst	60,000	2020-01-3

The interface also includes a sidebar with various tool icons, a bottom status bar with task information, and a search bar for tasks.

--Sort employees by salary or date of joining using `ORDER BY`--

```
select *  
from emp.employees e  
order by salary desc  
;
```

The screenshot shows a database management tool interface. The top panel displays a SQL query: `SELECT * FROM emp.employees e ;` followed by `select *`, `from emp.employees e`, `order by salary desc`, and a semicolon. The middle panel shows the results of the query in a table format. The table has columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Joining. The results are sorted by salary in descending order. The bottom panel shows the tool's interface with various buttons and a task bar.

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
7	Imam	Sourav	D007	Operations Manager	90,000	2017-
2	Meharab	Hossain	D002	Project Manager	85,000	2019-
12	Akik	Khan	D012	Brand Strategist	83,000	2019-
18	Nahid	Karim	D018	Business Analyst	80,000	2022-
5	Toukir	Ahmed	D005	Marketing Manager	78,000	2018-
11	Masud	Ahmed	D011	Software Developer	75,000	2020-
10	Rakibuzzaman	Khan	D010	Financial Analyst	72,000	2023-

--Use LIMIT to display only the top 5 highest-paid employees--

```
select *  
from emp.employees e  
order by salary desc  
limit 5  
;
```

The screenshot shows a database management tool interface. At the top, a SQL query is entered in a text area:

```
SELECT * FROM emp.employees e ;  
select *  
from emp.employees e  
order by salary desc  
limit 5  
;
```

Below the query editor, a table titled "employees 1" displays the results of the query. The table has 8 columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Joining. The results are sorted by salary in descending order, showing the top 5 highest-paid employees.

	Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
1	7	Imam	Sourav	D007	Operations Manager	90,000	2017-01-
2	2	Meharab	Hossain	D002	Project Manager	85,000	2019-01-
3	12	Akik	Khan	D012	Brand Strategist	83,000	2019-07-
4	18	Nahid	Karim	D018	Business Analyst	80,000	2022-02-
5	5	Toukir	Ahmed	D005	Marketing Manager	78,000	2018-01-

At the bottom of the interface, there is a "Database Tasks - General" section with a search bar for tasks and a "Task executions" section for error messages.



--Use JOIN statements to retrieve employees along with their department names or salaries--

**\*Inner join:**

```
select *  
from emp.employees e  
inner join emp.departments d on  
e.`Department ID` = d.`Department ID`  
;
```

The screenshot displays a database management interface. At the top, a SQL query is entered in a text area:

```
SELECT * FROM emp.employees e ;  
select *  
from emp.employees e  
inner join emp.departments d on  
e.`Department ID` = d.`Department ID`  
;
```

Below the query, the results are shown in a table grid. The table has 8 columns: Employee ID, A-Z First Name, A-Z Last Name, A-Z Department ID, A-Z Job Title, salary, and Date of Joining. The data is as follows:

Employee ID	A-Z First Name	A-Z Last Name	A-Z Department ID	A-Z Job Title	salary	Date of Joining
1	Arman	Uddin	D001	Software Engineer	70,000	2020-
2	Meharab	Hossain	D002	Project Manager	85,000	2019-
3	Anwar	Hossain	D003	Data Analyst	60,000	2020-
4	Walid	Dipto	D004	HR Specialist	55,000	2022-
5	Toukir	Ahmed	D005	Marketing Manager	78,000	2018-
6	Nazmul	Haque	D006	Sales Associate	45,000	2023-
7	Imam	Sourav	D007	Operations Manager	90,000	2017-

The interface includes a toolbar with various icons for grid, text, and record views, as well as a 'Refresh' button and a 'Save' button. The bottom status bar shows 'BDT en Writable Smart Insert 6:2:141 Sel: 0 | 0'.

### \*Left join:

```
select e.`Employee ID`,e.`First Name`,e.`Last Name`,e.`Department ID`,e.`Job Title`,
s.salary,s.`Date of Salary Update`
from emp.employees e
left join emp.salaries s on
e.`Employee ID` = s.`Employee ID`
;
```

The screenshot shows a database management tool interface. The top panel displays a SQL query for a left join between the 'employees' and 'salaries' tables. The query is as follows:

```
SELECT * FROM emp.employees e ;
select e.`Employee ID`,e.`First Name`,e.`Last Name`,e.`Department ID`,e.`Job Title`,s.salary,s.`Date of Salary Update`
from emp.employees e
left join emp.salaries s on
e.`Employee ID` = s.`Employee ID`
;
```

The bottom panel shows the results of the query in a table format. The table has 8 columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Salary Update. The results are as follows:

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Salary Update
1	Arman	Uddin	D001	Software Engineer	81,000	
2	Meharab	Hossain	D002	Project Manager	85,000	
3	Anwar	Hossain	D003	Data Analyst	60,000	
4	Walid	Dipto	D004	HR Specialist	55,000	
5	Toukir	Ahmed	D005	Marketing Manager	78,000	
6	Nazmul	Haque	D006	Sales Associate	45,000	
7	Imam	Sourav	D007	Operations Manager	90,000	

The interface also includes a sidebar with various tool icons, a bottom status bar with task information, and a search bar for tasks.

### \*Right join:

```
select e.`Employee ID`, e.`First Name`,e.`Last Name`,e.`Department ID`,e.`Job  
Title`,s.salary ,s.`Date of Salary Update`  
from emp.employees e  
right join emp.salaries s on  
e.`Employee ID` = s.`Employee ID`  
;
```

The screenshot displays a database management interface. At the top, a SQL query is entered in a text area:

```
SELECT * FROM emp.employees e ;  
select e.`Employee ID`, e.`First Name`,e.`Last Name`,e.`Department ID`,e.`Job Title`,s.salary ,s.`Date of Salary Update`  
from emp.employees e  
right join emp.salaries s on  
e.`Employee ID` = s.`Employee ID`  
;
```

Below the query editor, the results are shown in a table grid. The table has 8 columns: Employee ID, First Name, Last Name, Department ID, Job Title, salary, and Date of Salary Update. The data is as follows:

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Salary Update
1	Arman	Uddin	D001	Software Engineer	81,000	
2	Meharab	Hossain	D002	Project Manager	85,000	
3	Anwar	Hossain	D003	Data Analyst	60,000	
4	Walid	Dipto	D004	HR Specialist	55,000	
5	Toukir	Ahmed	D005	Marketing Manager	78,000	
6	Nazmul	Haque	D006	Sales Associate	45,000	
7	Imam	Sourav	D007	Operations Manager	90,000	

The interface includes a sidebar with icons for various database functions, a bottom status bar showing 'BDT en Writable Smart Insert 6:2:247 Sel: 0 | 0', and a 'Database Tasks - General' section at the bottom.

--Use COUNT to find the total number of employees in each department--

```
select `Department Name`, count(`Department ID`)
from emp.departments d
group by `Department Name`
;
```

The screenshot shows a database management tool interface. The top panel displays a SQL query:

```
SELECT * FROM emp.employees e ;
select `Department Name`, count(`Department ID`)
from emp.departments d
group by `Department Name`
;
```

The bottom panel shows the results of the query in a table with 2 columns: 'Department Name' and 'count(Department ID)'. The table has 8 rows of data:

Department Name	count(Department ID)
Developer	3
Project Management	1
Sales & Marketing	4
Human Resources	3
Operations	1
Technical	1
Audit	1
Customer Service	1

The interface also includes a sidebar with icons for various database operations, a 'Database Tasks - General' tab at the bottom, and a status bar showing 'BDT en Writable Smart Insert 6:1:141 Sel: 0 | 0'.

--Use AVG to calculate the average salary of employees--

```
select avg(salary)
from emp.employees e
;
```

The screenshot shows a database management tool interface. The top panel displays the SQL query: `SELECT * FROM emp.employees e ;` and `select avg(salary) from emp.employees e ;`. The bottom panel shows the results of the query. The first row of the results is highlighted, showing the average salary of 68,650. The interface includes a toolbar with various icons for navigation and execution, and a status bar at the bottom indicating the number of rows fetched and the execution time.

Grid	123 avg(salary)
1	68,650

Value: 68650.0000

1 row(s) fetched - 0.001s, on 2024-10-12 at 20:06:44

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

BDT en Writable Smart Insert 8:1:86 Sel: 0 | 0

--Identify the highest and lowest salaries using MAX and MIN--

```
select max(salary)
from emp.employees e
;
```

The screenshot shows a database IDE interface. The top panel displays a SQL script with three queries. The second query, `select max(salary) from emp.employees e ;`, is selected and highlighted. Below the script editor, the 'Results 1' tab is active, showing a single row of results in a grid. The grid has two columns: '123 max(salary)' and '90,000'. To the right of the grid, a 'Value' panel displays the value '90000'. The bottom panel shows the 'Database Tasks - General' tab with a search bar and a list of tasks. The status bar at the very bottom indicates 'BDT en Writable', 'Smart Insert', and '12:1:157'.

```
SELECT * FROM emp.employees e ;
select * from emp.departments d ;
select * from emp.salaries s ;

select max(salary)
from emp.employees e
;
```

123 max(salary)
90,000

Value X  
90000

Database Tasks - General X  
Tasks: type a part of task name here  
Task executions: type a part of error message

BDT en Writable Smart Insert 12:1:157 Sel: 0 | 0

```
select min(salary)
from emp.employees e
;
```

The screenshot shows a database management tool interface. The top panel displays a SQL script with three queries. The second query, `select min(salary) from emp.employees e ;`, is selected and highlighted. Below the script, the 'Results 1' tab is active, showing a single row of results in a grid. The grid has two columns: the first column contains the value '1' and the second column contains the value '45,000'. To the right of the grid, a 'Value' panel displays the result '45000'. The bottom panel shows the 'Database Tasks - General' tab with a search bar and a list of tasks. The status bar at the very bottom indicates 'BDT en Writable', 'Smart Insert', '10: 1: 153', and 'Sel: 0 | 0'.

```
SELECT * FROM emp.employees e ;
select * from emp.departments d ;
select * from emp.salaries s ;

select min(salary)
from emp.employees e
;
```

Results 1 X

select min(salary) from emp.employees e Enter a SQL expression to filter results (use Ctrl+Space)

Grid	123 min(salary)
1	45,000

Value X

45000

Refresh Save Cancel Export data 100 1

Database Tasks - General X

Tasks: type a part of task name here Task executions: type a part of error message

BDT en Writable Smart Insert 10: 1: 153 Sel: 0 | 0

--Summarize the total salaries of each department using SUM--

```
select sum(salary)
from emp.employees e
;
```

The screenshot shows a database management tool interface. The top panel displays a SQL query: `SELECT * FROM emp.employees e ;`, `select * from emp.departments d ;`, `select * from emp.salaries s ;`, and `select sum(salary) from emp.employees e ;`. The bottom panel shows the results of the query in a grid format. The grid has one column labeled `sum(salary)` and one row with the value `1,373,000`. The right side of the interface shows a 'Value' field with the value `1373000`. The bottom status bar shows 'Database Tasks - General' and 'Task executions: type a part of error message'.

Grid	123 sum(salary)
1	1,373,000

Value: 1373000

Database Tasks - General

Task executions: type a part of error message



--Use GROUP BY to group employees by department and calculate the number of employees in each department--

```
select `Department Name`, count(`Department ID`)
from emp.departments d
group by `Department Name`
;
```

The screenshot displays a database management interface. The top pane shows the SQL query: `SELECT * FROM emp.employees e ; select * from emp.departments d ; select * from emp.salaries s ;` followed by a collapsed query: `select `Department Name`, count(`Department ID`) from emp.departments d group by `Department Name` ;`. The bottom pane shows the results of the collapsed query in a table grid. The table has two columns: 'Department Name' and 'count(Department ID)'. The results are as follows:

Department Name	count(Department ID)
Developer	3
Project Management	1
Sales & Marketing	4
Human Resources	3
Operations	1
Technical	1
Audit	1

The interface also includes a status bar at the bottom indicating '12 row(s) fetched - 0.000s, on 2024-10-12 at 14:39:19' and a task pane on the right.

--Use HAVING to filter departments with more than 10 employees--

```
select `Department Name`, count(*) as Total_departments
from emp.departments d
group by `Department Name`
having count(*) > 10
;
```

The screenshot displays a database management interface. The top panel shows the SQL query being executed:

```
SELECT * FROM emp.employees e ;
select * from emp.departments d ;
select * from emp.salaries s ;

select `Department Name`, count(*) as Total_departments
from emp.departments d
group by `Department Name`
having count(*) >10
;
```

The bottom panel shows the results of the query in a grid view. The grid has two columns: "Department Name" and "Total\_departments". The first row shows "A-Z Department Name" and "123 Total\_departments". The rest of the grid is empty.

The interface also includes a toolbar with various icons for file operations, a status bar at the bottom showing "No data- 0.001s, on 2024-10-12 at 15:13:31", and a task execution panel at the very bottom.

In the datasheet, all department value is below 10. So, I add another value.

```
select `Department Name`, count(*) as Total_departments
from emp.departments d
group by `Department Name`
having count(*) >= 4
;
```

The screenshot displays a database management interface. The top pane shows a SQL query editor with the following code:

```
SELECT * FROM emp.employees e ;
select * from emp.departments d ;
select * from emp.salaries s ;

select `Department Name`, count(*) as Total_departments
from emp.departments d
group by `Department Name`
having count(*) >=4
;
```

The bottom pane shows the results of the query in a datasheet view. The table has two columns: 'Department Name' and 'Total\_departments'. The first row shows 'Sales & Marketing' with a value of 4. The interface includes a toolbar with various icons for navigation and data manipulation, and a status bar at the bottom indicating '1 row(s) fetched - 0.001s, on 2024-10-12 at 15:14:58'.

	Department Name	Total_departments
1	Sales & Marketing	4

--Use ROW\_NUMBER to rank employees based on their salary within each department--

```
select `Employee ID`, salary,  
row_number() over (order by salary desc) as row_num  
from emp.employees  
;
```

The screenshot displays a database management interface. The top panel shows the SQL query being executed: `SELECT * FROM emp.employees e ; select * from emp.departments d ; select * from emp.salaries s ; select * from emp.performance_reviews pr ;` followed by the specific query for ranking employees by salary: `select `Employee ID`, salary, row_number() over (order by salary desc) as row_num from emp.employees e ;`. The middle panel shows the results in a grid format with columns: Employee ID, salary, and row\_num. The bottom panel shows the 'Database Tasks - General' tab with a search bar and a list of tasks.

	Employee ID	salary	row_num
1	7	90,000	1
2	2	85,000	2
3	12	83,000	3
4	18	80,000	4
5	5	78,000	5
6	11	75,000	6
7	10	72,000	7
8	17	71,000	8

--Insert a new employee into the employees table--

```
insert into emp.employees
values
(21,'Nazmul','Hasan','D021','Principle Officer',95000,'2018-02-01')
;
```

The screenshot displays a database management interface. The top section shows a SQL script with three queries: a SELECT statement for employees and departments, an INSERT statement for a new department 'D021' named 'Treasury', and another INSERT statement for a new employee 'Nazmul' in department 'D021'. Below the script, a data grid titled 'employees 1' shows the results of a SELECT \* FROM emp.employees query. The grid contains 5 rows of employee data. The bottom section shows the 'Database Tasks - General' tab with a search bar and a list of tasks.

```
SELECT * FROM emp.employees e ;
select * from emp.departments d ;

insert into emp.departments (`Department ID`, `Department Name`)
values
('D021','Treasury')
;

insert into emp.employees
values
(21,'Nazmul','Hasan','D021','Principle Officer',95000,'2018-02-01')
;
```

Employee ID	First Name	Last Name	Department ID	Job Title	salary	Date of Joining
17	Mosrur	Sunny	D017	Compliance Officer	71,000	2020-
18	Nahid	Karim	D018	Business Analyst	80,000	2022-
19	Talha	Howlader	D019	Web Developer	69,000	2020-
20	Towhidul	Islam	D020	Graphic Designer	64,000	2020-
21	Nazmul	Hasan	D021	Principle Officer	95,000	2018-

Refresh Save Cancel Export data 100 21

... 21 row(s) fetched - 0.001s, on 2024-10-12 at 20:13:44

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

BDT en Writable Smart Insert 13:1:279 Sel: 0 | 0

--Update the salary of an employee in the salaries table--

```
update emp.salaries
set salary = 81000
where `Date of Salary Update` = '2021-01-05'
;
```

The screenshot shows a database management interface. The top pane displays SQL queries. The bottom pane shows the results of a query in a grid format. The grid has columns for Employee ID, salary, and Date of Salary Update. The data is as follows:

Employee ID	salary	Date of Salary Update
1	81,000	2021-01-05
2	85,000	2020-01-02
3	60,000	2021-01-03
4	55,000	2023-02-10
5	78,000	2019-02-06
6	45,000	2024-01-08
7	90,000	2018-12-12

The interface also includes a sidebar with icons for various database operations, a status bar at the bottom showing '20 row(s) fetched - 0.001s, on 2024-10-12 at 15:44:43', and a task execution area at the very bottom.

--Delete an employee who left the company from the employees' table--

```
delete from emp.employees
where salary = 95000
;
```

The screenshot displays a database management interface. The top pane shows a SQL script with three queries: a SELECT statement for employees, departments, and salaries; an INSERT statement for a new employee; and a DELETE statement for an employee with a salary of 95000. The bottom pane shows the execution results for the DELETE query.

Name	Value
Updated Rows	1
Query	delete from emp.employees where salary = 95000
Start time	Sat Oct 12 20:20:57 BDT 2024
Finish time	Sat Oct 12 20:20:57 BDT 2024

Database Tasks - General

Tasks: type a part of task name here

Task executions: type a part of error message

BDT en Writable Smart Insert 13: 1: 266 Sel: 0 | 0

