

Assignment -

Theory

1. compare sql and nosql databases.

Aspect	SQL (Relational)	NoSQL (Non-relational)
Data Structure	Tables with rows and columns	Document-based, key-value, column-family, or graph-based
Schema	Fixed schema (predefined structure)	Flexible schema (dynamic and adaptable)
Scalability	Vertically scalable (upgrading hardware)	Horizontally scalable (adding more servers)
Data Integrity	ACID-compliant (strong consistency)	BASE-compliant (more available, less consistent)
Query Language	SQL (Structured Query Language)	Varies (e.g., MongoDB uses its own query language)
Performance	Efficient for complex queries and transactions	Better for large-scale data and fast read/write operations
Use Case	Best for transactional systems (banking, ERP, etc.)	Ideal for big data, real-time web apps, and data lakes
Examples	MySQL, PostgreSQL, Oracle, MS SQL Server	MongoDB, Cassandra, CouchDB, Neo4j

Practical-

1. Write a SQL query to find customers who are either from the city 'New York' or who do not have a grade greater than 100. Return customer_id, cust_name, city, grade, and salesman_id.

```
select customer_id,cust_name,city,grade,salesman_id
from customer
where city = 'newyork' or grade <= 100 OR grade IS NULL;
```

customer_id	cust_name	city	grade	salesman_id
3001	Brad Guzan	London	100	5005
3002	Nick Rimando	New York	100	5001
3009	Geoff Cameron	Berlin	100	5003
3008				

2. Write a SQL query to find all the customers in 'New York' city who have a grade value above 100. Return customer_id, cust_name, city, grade, and salesman_id.

```
select customer_id, cust_name, city, grade, salesman_id
from customer
```

where city = 'new york' and grade >= 100;

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3007	Brad Davis	New York	200	5001
3008		New York	300	5001

3. Write a SQL query that displays order number, purchase amount, and the achieved and unachieved percentage (%) for those orders that exceed 50% of the target value of 6000.

```
select ord_no, round((purch_amt /6000)*100,2) as archived, round((100-purch_amt /6000)*100,2)
as unarchived
from orders
where purch_amt > 3000;
```

ord_no	archived	unarchived
70008	96.00	9904.00
70013	50.76	9949.24

4. Write a SQL query to calculate the total purchase amount of all orders. Return total purchase amount.

```
select sum(purch_amt) as totalpurchase
from orders;
```

totalpurchase
17541.18

5. Write a SQL query to find the highest purchase amount ordered by each customer. Return customer_id, maximum purchase amount.

```
select customer_id,max(purch_amt) as maxpurchase
from orders
group by customer_id
order by maxpurchase;
```

customer_id	maxpurchase
3003	75.29
3008	250.45
3001	270.45
3005	948.50
3004	1983.43
3007	2400.60
3009	2480.40
3002	5760.00

6. Write a SQL query to calculate the average product price. Return average product price.

```
select round(avg(pro_price),2) as avgprice
from item_mast;
```

avgprice
1435.00

7. Write a SQL query to find those employees whose department is located at 'Toronto'. Return first name, last name, employee_id, job_id.

```
SELECT e.employee_id,e.first_name,e.last_name,e.job_id
FROM employees e
join departments d
on e.department_id = d.department_id
```

join locations l

on d.location_id = l.location_id

where l.city='toronto';

employee_id	first_name	last_name	job_id
201	Michael	Hartstein	MK_MAN
202	Pat	Fay	MK_REP

8. Write a SQL query to find those employees whose salary is lower than that of employees whose job title is "MK_MAN". Exclude employees of the job title "MK_MAN". Return employee_id, first name, last name, job_id.

select employee_id,first_name,last_name,job_id

from employees

where salary <

(select min(salary) lowersalary

from employees

where job_id = 'MK_MAN')

and job_id != 'MK_MAN';



The screenshot shows the SQL Developer interface with a query result grid. The grid contains the following data:

employee_id	first_name	last_name	job_id
108	Nancy	Greenberg	AC_ACCOUNT
109	Daniel	Faviet	FI_ACCOUNT
111	Jamuel	Scarra	FI_ACCOUNT
112	Jose Manuel	Urman	FI_ACCOUNT
113	Luis	Popp	FI_ACCOUNT
176	Mohammed	Slayer	OT_MAN

9. Write a SQL query to find all those employees who work in department ID 80 or 40. Return first name, last name, department number, and department name.

select e.first_name,e.last_name,d.department_id,d.department_name

from employees e

join departments d

on e.department_id = d.department_id

where d.department_id = 80 or d.department_id= 40;



The screenshot shows the SQL Developer interface with a query result grid. The grid contains the following data:

first_name	last_name	department_id	department_name
Susan	Mavris	40	Human Resources

10. Write a SQL query to calculate the average salary, the number of employees receiving commissions in that department. Return department name, average salary and number of employees.

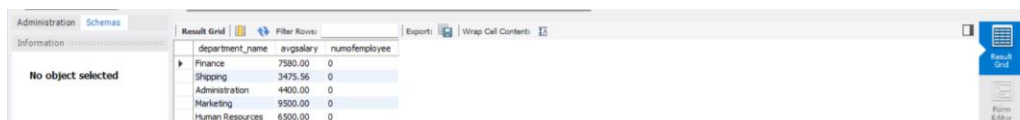
select d.department_name , round(avg(e.salary),2) avgsalary ,count(e.commission_pct) numofemployee

from employees e

join departments d

on e.department_id = d.department_id

group by d.department_name;



The screenshot shows the SQL Developer interface with a query result grid. The grid contains the following data:

department_name	avgsalary	numofemployee
Finance	7580.00	0
Shipping	3475.56	0
Administration	4400.00	0
Marketing	9500.00	0
Human Resources	6500.00	0

11. Write a SQL query to find out which employees have the same designation as the employee whose ID is 169. Return first name, last name, department ID, and job ID.

select*from employees;

```

select first_name,last_name,department_id,job_id
from employees
where job_id = (
    select job_id
    from employees
    where employee_id = 169);

```

The screenshot shows a SQL query result grid with the following data:

first_name	last_name	department_id	job_id
Peter	Tucker	80	SA_REP
David	Bernstein	80	SA_REP
Peter	Hall	80	SA_REP
Christopher	Olsen	80	SA_REP
Nanette	Cambrault	80	SA_REP
Oliver	Tunault	80	SA_REP
Janette	King	80	SA_REP
Patrick	Sully	80	SA_REP
Allan	McEwen	80	SA_REP
Lindsey	Smith	80	SA_REP
Louise	Doran	80	SA_REP
Sarah	Sewall	80	SA_REP
Clara	Vishney	80	SA_REP
Danielle	Greene	80	SA_REP
Mattea	Marvins	80	SA_REP
David	Lee	80	SA_REP
Sundar	Ande	80	SA_REP
Amit	Banda	80	SA_REP
Lisa	Ozer	80	SA_REP
Harrison	Bloom	80	SA_REP
Taylor	Fox	80	SA_REP
William	Smith	80	SA_REP
Elizabeth	Bates	80	SA_REP
Sundita	Kumar	80	SA_REP

12. Write a SQL query to find those employees who earn more than the average salary. Return employee ID, first name, last name.

```

select employee_id,first_name,last_name,salary
from employees
where salary >
(select avg(salary) avgsalary
from employees);

```

The screenshot shows a SQL query result grid with the following data:

employee_id	first_name	last_name	salary
108	Nancy	Greenberg	6500.00
109	Daniel	Faviet	9000.00
111	Ismael	Sciera	7700.00
112	Jose Manuel	Urman	7800.00
113	Luis	Popp	6900.00
114	Martinez	Vizcarra	8000.00

13. Write a SQL query to find all those employees who work in the Finance department. Return department ID, name (first), job ID, and department name.

```

select d.department_id, e.first_name,e.job_id,d.department_name
from employees e
join departments d
on e.department_id = d.department_id
where department_name = 'finance';

```

The screenshot shows a SQL query result grid with the following data:

department_id	first_name	job_id	department_name
100	Nancy	FI_MGR	Finance
100	Daniel	FI_ACCOUNT	Finance
100	Ismael	FI_ACCOUNT	Finance
100	Jose Manuel	FI_ACCOUNT	Finance
100	Luis	FI_ACCOUNT	Finance

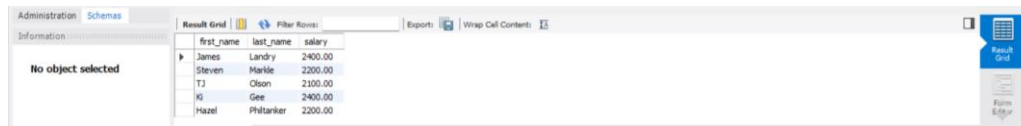
14. From the following table, write a SQL query to find the employees who earn less than the employee of ID 182. Return first name, last name, and salary.

```

select first_name,last_name, salary
from employees
where salary <
(select salary
from employees

```

where employee_id = 182);



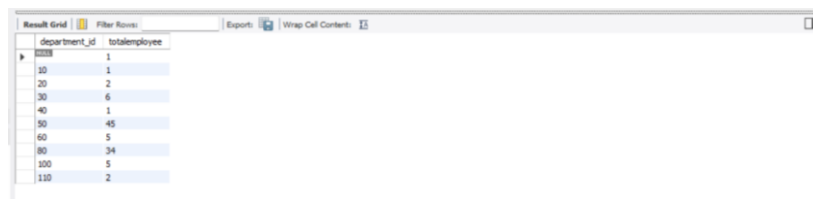
The screenshot shows the SQL Developer interface. On the left, the 'Administration' tab is active, showing 'No object selected'. The main area displays a 'Result Grid' with the following data:

first_name	last_name	salary
James	Landry	2400.00
Steven	Markle	2200.00
TJ	Olsen	2100.00
KJ	Gee	2400.00
Hazel	Philtanker	2200.00

stored procedure

15. Create a stored procedure CountEmployeesByDept that returns the number of employees in each department.

```
delimiter //
create procedure countemployee()
begin
    select department_id,count(*) as totalemployee
    from employees
    group by department_id;
end//
delimiter ;
call countemployee();
```



The screenshot shows the SQL Developer interface with a 'Result Grid' displaying the output of the stored procedure. The data is as follows:

department_id	totalemployee
10	1
20	1
30	2
40	6
50	1
60	45
70	5
80	34
90	5
100	2

16. Create a stored procedure AddNewEmployee that adds a new employee to the database.

```
delimiter //
create procedure addnewemployee(
    in employee_id INT UNSIGNED ,
    in first_name VARCHAR(20),
    in last_name VARCHAR(25) ,
    in email VARCHAR(25) ,
    in phone_number VARCHAR(20),
    in hire_date DATE ,
    in job_id VARCHAR(10) ,
    in salary DECIMAL(8, 2),
    in commission_pct DECIMAL(5, 2),
    in manager_id INT UNSIGNED,
    in department_id INT UNSIGNED
)
begin
    INSERT INTO employees
VALUES
(employee_id, first_name, last_name, email, phone_number, hire_date, job_id, salary, commission_pct, manager_id, department_id);
end //
delimiter ;
SET FOREIGN_KEY_CHECKS = 1;
call addNEWemployee(209, 'gargi', 'Baer', 'HBAER', '515.123.8888', '1994-06-08', 'PR_REP', 10000, NULL, 101, 70);
```

SET FOREIGN_KEY_CHECKS = 1;

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	commission_pct	manager_id	department_id
197	Kevin	Feeney	KFEENEY	650.507.9832	1998-05-23	SH_CLERK	3000.00	0.10	124	50
198	Donald	OConnell	DOCONNEL	650.507.9833	1999-06-21	SH_CLERK	2600.00	0.10	124	50
199	Douglas	Grant	DGRANT	650.507.9844	2000-01-13	SH_CLERK	2600.00	0.10	124	50
200	Jennifer	Whalen	JWHALEN	515.123.4444	1987-09-17	AD_ASST	4400.00	0.10	101	10
201	Michael	Hartstein	MHARTSTE	515.123.5555	1996-02-17	MK_MAN	13000.00	0.10	100	20
202	Pat	Fay	PFAY	603.123.6666	1997-08-17	MK_REP	6000.00	0.10	201	20
203	Susan	Mavris	SMAVRIS	515.123.7777	1994-06-07	HR_REP	6500.00	0.10	101	40
205	Shelley	Higgins	SHIGGINS	515.123.8000	1994-06-07	AC_MGR	12000.00	0.10	101	110
206	William	Gietz	WGIEZT	519.5.123.8181	1994-06-07	AC_ACCOUNT	8300.00	0.10	205	110
209	Gargi	Baer	HBAER	515.123.8888	1994-06-08	PR_REP	10000.00	0.10	101	70

17. Create a stored procedure DeleteEmployeesByDept that removes all employees from a specific department.

```
delimiter //

create procedure DeleteEmployeesByDept(in dept_id int unsigned)
begin
    delete
    from employees
    where department_id = dept_id;
end //

delimiter ;

SET FOREIGN_KEY_CHECKS = 0;

call DeleteEmployeesByDept(100);

SET FOREIGN_KEY_CHECKS = 1;
```

18. Create a stored procedure GetTopPaidEmployees that retrieves the highest-paid employee in each department.

```
DELIMITER //

CREATE PROCEDURE GetTopPaidEmployees()
BEGIN
    SELECT e.employee_id, e.first_name, e.last_name, e.department_id, e.salary
    FROM employees e
    WHERE e.salary = (
        SELECT MAX(salary)
        FROM employees
        WHERE department_id = e.department_id
    );
END //

DELIMITER ;

CALL GetTopPaidEmployees();
```

employee_id	first_name	last_name	department_id	salary
121	Adam	Frip	50	8200.00
200	Jennifer	Whalen	10	4400.00
201	Michael	Hartstein	20	13000.00
203	Susan	Mavris	40	6500.00

19. Create a stored procedure PromoteEmployee that increases an employee's salary and changes their job role.

```
delimiter //

create procedure PromoteEmployee(IN emp_id INT,
    IN new_salary DECIMAL(8,2),
    IN new_job_id VARCHAR(10)
)
```

```

begin
update employees
set salary = new_salary,
    job_id = new_job_id
WHERE employee_id = emp_id;
end //
delimiter ;
call PromoteEmployee(105, 5500.00, 'HR_REP');

```

66	17:38:54	call PromoteEmployee(105, 5500.00, 'HR_REP')	1 row(s) affected	0.000 sec
----	----------	----------------------------------------------	-------------------	-----------

20. Create a stored procedure AssignManagerToDepartment that assigns a new manager to all employees in a specific department.

```

delimiter //
create procedure AssignManagerToDepartment (in dept_id INT UNSIGNED,
in new_manager_id INT UNSIGNED)
begin
update employees
set manager_id = new_manager_id
where department_id = dept_id;
end //
delimiter ;
call AssignManagerToDepartment(50,114);

```

69	11:45:25	call AssignManagerToDepartment(40,202)	1 row(s) affected	0.000 sec
----	----------	----------------------------------------	-------------------	-----------

No-sql

1.retrieve all employee records.

```
db.collection.find()
```

```

employee> db.collection.find()
{
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a69'),
    name: 'Alice Johnson',
    age: 30,
    department: 'HR',
    salary: 60000,
    joining_date: '2019-05-15'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6a'),
    name: 'Bob Smith',
    age: 40,
    department: 'IT',
    salary: 80000,
    joining_date: '2015-08-20'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6b'),
    name: 'Charlie Brown',
    age: 35,
    department: 'Finance',
    salary: 70750,
    joining_date: '2018-11-30'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6c'),
    name: 'David White',
    age: 20,
    department: 'IT',
    salary: 72000,
    joining_date: '2021-01-10'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6d'),
    name: 'Ema Wilson',
    age: 32,
    department: 'Marketing',
    salary: 65000,
    joining_date: '2017-03-25'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6e'),
    name: 'Franklin Adams',
    age: 45,
    department: 'Finance',
    salary: 94500,
    joining_date: '2020-12-15'
  }
}

```

2.find employee who work in the IT department.

db.collection.find({department:'IT'});

```

employee> db.collection.find({department:'IT'});
{
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6a'),
    name: 'Bob Smith',
    age: 40,
    department: 'IT',
    salary: 80000,
    joining_date: '2015-08-20'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6c'),
    name: 'David White',
    age: 20,
    department: 'IT',
    salary: 72000,
    joining_date: '2021-01-10'
  },
  {
    _id: ObjectId('685d1125f2b99a1d8c748a60'),
    name: 'Bob Smith',
    age: 40,
    department: 'IT',
    salary: 80000,
    joining_date: '2015-08-20'
  },
  {
    _id: ObjectId('685d1125f2b99a1d8c748a62'),
    name: 'David White',
    age: 20,
    department: 'IT',
    salary: 72000,
    joining_date: '2021-01-10'
  },
  {
    _id: ObjectId('685d1125f2b99a1d8c748a66'),
    name: 'Henry Ford',
    age: 60,
    department: 'IT',
    salary: 95000,
    joining_date: '2000-12-15'
  }
}

```

3.find employee who have a salary greater than 70000.

db.collection.find({salary:{\$gt:70000}});


```

employee> db.collection.find({salary:{$gt:70000}});
[
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6a'),
    name: 'Bob Smith',
    age: 40,
    department: 'IT',
    salary: 80000,
    joining_date: '2015-08-20'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6b'),
    name: 'Charlie Brown',
    age: 35,
    department: 'Finance',
    salary: 70750,
    joining_date: '2018-11-30'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6c'),
    name: 'David White',
    age: 28,
    department: 'IT',
    salary: 72000,
    joining_date: '2021-01-10'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6d'),
    name: 'Franklin Adams',
    age: 45,
    department: 'Finance',
    salary: 94500,
    joining_date: '2010-07-12'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a72'),
    name: 'Jack Carter',
    age: 27,
    department: 'Finance',
    salary: 71400,
    joining_date: '2022-04-10'
  },
  {
    _id: ObjectId('685d1125f2b99ald8c748a60'),
    name: 'Bob Smith',
    age: 40,
    department: 'IT',
    salary: 80000,
  }
]

```

4. find employee who joined after 2018.

```
db.collection.find({joining_date:{$gt:'2018'}});
```

```

employee> db.collection.find({joining_date:{$gt:'2018'}});
[
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a69'),
    name: 'Alice Johnson',
    age: 30,
    department: 'HR',
    salary: 60000,
    joining_date: '2019-05-15'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6b'),
    name: 'Charlie Brown',
    age: 35,
    department: 'Finance',
    salary: 70750,
    joining_date: '2018-11-30'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6c'),
    name: 'David White',
    age: 28,
    department: 'IT',
    salary: 72000,
    joining_date: '2021-01-10'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a6f'),
    name: 'Grace Lee',
    age: 29,
    department: 'HR',
    salary: 58000,
    joining_date: '2020-06-05'
  },
  {
    _id: ObjectId('685bc5f8d1ebce75a6748a72'),
    name: 'Jack Carter',
    age: 27,
    department: 'Finance',
    salary: 71400,
    joining_date: '2022-04-10'
  },
  {
    _id: ObjectId('685d1125f2b99ald8c748a6f'),
    name: 'Alice Johnson',
    age: 30,
    department: 'HR',
    salary: 60000,
    joining_date: '2019-05-15'
  },
]

```

5.find employee between the ages of 30 and 40.

```
db.collection.find({age:{$gt:30,$lt:40}});
```

```

employee> db.collection.find({age:{$gt:30,$lt:40}});
{
  _id: ObjectId('685bc5f8d1ebce75a6748a6b'),
  name: 'Charlie Brown',
  age: 35,
  department: 'Finance',
  salary: 78750,
  joining_date: '2018-11-30'
},
{
  _id: ObjectId('685bc5f8d1ebce75a6748a6d'),
  name: 'Emma Wilson',
  age: 32,
  department: 'Marketing',
  salary: 65000,
  joining_date: '2017-03-25'
},
{
  _id: ObjectId('685bc5f8d1ebce75a6748a71'),
  name: 'Isabella Martinez',
  age: 38,
  department: 'Marketing',
  salary: 78000,
  joining_date: '2016-09-18'
},
{
  _id: ObjectId('685d1125f2b99ald9c748a61'),
  name: 'Charlie Brown',
  age: 35,
  department: 'Finance',
  salary: 75000,
  joining_date: '2018-11-30'
},
{
  _id: ObjectId('685d1125f2b99ald9c748a63'),
  name: 'Emma Wilson',
  age: 32,
  department: 'Marketing',
  salary: 65000,
  joining_date: '2017-03-25'
},
{
  _id: ObjectId('685d1125f2b99ald9c748a67'),
  name: 'Isabella Martinez',
  age: 38,
  department: 'Marketing',
  salary: 78000,
  joining_date: '2016-09-18'
}

```

6.increase the salary of all employee in the finance department by 5%.

```
db.collection.updateMany({department:'Finance'},{$mul:{salary:0.05}});
```

```

employee> db.collection.updateMany({department:'Finance'},{$mul:{salary:1.05}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 6,
  modifiedCount: 6,
  upsertedCount: 0
}

```

7.delete employee who joined before 2010.

```
db.collection.deleteMany({joining_date:{$lt:"2010"}});
```

```

employee> db.collection.deleteMany({joining_date:{$lt:"2010"}});
{ acknowledged: true, deletedCount: 1 }
employee>

```

8. find the highest-paid employee.

```
db.collection.find().sort({salary:-1}).limit(1);
```

```

employee> db.collection.find().sort({salary:-1}).limit(1);
{
  _id: ObjectId('685bc5f8d1ebce75a6748a6e'),
  name: 'Franklin Adams',
  age: 45,
  department: 'Finance',
  salary: 99225,
  joining_date: '2010-07-12'
}

```