

## LAB-1

E-81 Date \_\_\_\_\_  
Page \_\_\_\_\_

- (Q) Illustrate an example to create a table Student which contains the following attributes stdid, stdname, dob, doi, fees, gender.

I/P CREATE DATABASE temp;  
USE temp;

CREATE TABLE Student(  
stdid INT PRIMARY KEY,  
stdname VARCHAR(50),  
dob DATE,  
doi DATE, fees INT, fees  
gender VARCHAR(100);

INSERT INTO Student VALUES (1)

(1, "Bheem", "2000-03-01", "2022-08-01",  
125000, "male"), (2, "Bhupender", "1999-09-12", "2022-09-03",  
25000, "male"), (3, "Dogelina", "2001-12-23", "2021-07-15",  
90000, "female");

(4, "Saranya", "2000-08-01", "2022-09-01", 100000, "female");

(5, "Rishabh", "2002-03-01", "2022-08-01", 25000, "male");

(6, "Aman", "2003-09-01", "2022-09-01", 15000, "male");

(7, "Vishal", "2004-05-01", "2022-09-01", 10000, "male");

(8, "Akash", "2005-11-01", "2022-09-01", 15000, "male");

(9, "Aniket", "2006-07-01", "2022-09-01", 10000, "male");

(10, "Pratik", "2007-01-01", "2022-09-01", 10000, "male");

(11, "Rishabh", "2008-04-01", "2022-09-01", 10000, "male");

(12, "Aayush", "2009-08-01", "2022-09-01", 10000, "male");

(13, "Vishal", "2010-12-01", "2022-09-01", 10000, "male");

(14, "Akash", "2011-06-01", "2022-09-01", 10000, "male");

(15, "Aniket", "2012-10-01", "2022-09-01", 10000, "male");

(16, "Pratik", "2013-02-01", "2022-09-01", 10000, "male");

(17, "Rishabh", "2014-08-01", "2022-09-01", 10000, "male");

(18, "Aayush", "2015-12-01", "2022-09-01", 10000, "male");

(19, "Vishal", "2016-06-01", "2022-09-01", 10000, "male");

(20, "Akash", "2017-10-01", "2022-09-01", 10000, "male");

(21, "Aniket", "2018-04-01", "2022-09-01", 10000, "male");

(22, "Pratik", "2019-08-01", "2022-09-01", 10000, "male");

(23, "Rishabh", "2020-12-01", "2022-09-01", 10000, "male");

(24, "Aayush", "2021-06-01", "2022-09-01", 10000, "male");

(25, "Vishal", "2022-10-01", "2022-09-01", 10000, "male");

(26, "Akash", "2023-04-01", "2022-09-01", 10000, "male");

(27, "Aniket", "2024-08-01", "2022-09-01", 10000, "male");

(28, "Pratik", "2025-12-01", "2022-09-01", 10000, "male");

(29, "Rishabh", "2026-06-01", "2022-09-01", 10000, "male");

(30, "Aayush", "2027-10-01", "2022-09-01", 10000, "male");

(31, "Vishal", "2028-04-01", "2022-09-01", 10000, "male");

(32, "Akash", "2029-08-01", "2022-09-01", 10000, "male");

(33, "Aniket", "2030-12-01", "2022-09-01", 10000, "male");

(34, "Pratik", "2031-06-01", "2022-09-01", 10000, "male");

(35, "Rishabh", "2032-10-01", "2022-09-01", 10000, "male");

(36, "Aayush", "2033-04-01", "2022-09-01", 10000, "male");

(37, "Vishal", "2034-08-01", "2022-09-01", 10000, "male");

(38, "Akash", "2035-12-01", "2022-09-01", 10000, "male");

(39, "Aniket", "2036-06-01", "2022-09-01", 10000, "male");

(40, "Pratik", "2037-10-01", "2022-09-01", 10000, "male");

(41, "Rishabh", "2038-04-01", "2022-09-01", 10000, "male");

(42, "Aayush", "2039-08-01", "2022-09-01", 10000, "male");

(43, "Vishal", "2040-12-01", "2022-09-01", 10000, "male");

(44, "Akash", "2041-06-01", "2022-09-01", 10000, "male");

(45, "Aniket", "2042-10-01", "2022-09-01", 10000, "male");

(46, "Pratik", "2043-04-01", "2022-09-01", 10000, "male");

(47, "Rishabh", "2044-08-01", "2022-09-01", 10000, "male");

(48, "Aayush", "2045-12-01", "2022-09-01", 10000, "male");

(49, "Vishal", "2046-06-01", "2022-09-01", 10000, "male");

(50, "Akash", "2047-10-01", "2022-09-01", 10000, "male");

(51, "Aniket", "2048-04-01", "2022-09-01", 10000, "male");

(52, "Pratik", "2049-08-01", "2022-09-01", 10000, "male");

(53, "Rishabh", "2050-12-01", "2022-09-01", 10000, "male");

(54, "Aayush", "2051-06-01", "2022-09-01", 10000, "male");

(55, "Vishal", "2052-10-01", "2022-09-01", 10000, "male");

(56, "Akash", "2053-04-01", "2022-09-01", 10000, "male");

(57, "Aniket", "2054-08-01", "2022-09-01", 10000, "male");

(58, "Pratik", "2055-12-01", "2022-09-01", 10000, "male");

(59, "Rishabh", "2056-06-01", "2022-09-01", 10000, "male");

(60, "Aayush", "2057-10-01", "2022-09-01", 10000, "male");

(61, "Vishal", "2058-04-01", "2022-09-01", 10000, "male");

(62, "Akash", "2059-08-01", "2022-09-01", 10000, "male");

(63, "Aniket", "2060-12-01", "2022-09-01", 10000, "male");

(64, "Pratik", "2061-06-01", "2022-09-01", 10000, "male");

(65, "Rishabh", "2062-10-01", "2022-09-01", 10000, "male");

(66, "Aayush", "2063-04-01", "2022-09-01", 10000, "male");

(67, "Vishal", "2064-08-01", "2022-09-01", 10000, "male");

(68, "Akash", "2065-12-01", "2022-09-01", 10000, "male");

(69, "Aniket", "2066-06-01", "2022-09-01", 10000, "male");

(70, "Pratik", "2067-10-01", "2022-09-01", 10000, "male");

(71, "Rishabh", "2068-04-01", "2022-09-01", 10000, "male");

(72, "Aayush", "2069-08-01", "2022-09-01", 10000, "male");

(73, "Vishal", "2070-12-01", "2022-09-01", 10000, "male");

(74, "Akash", "2071-06-01", "2022-09-01", 10000, "male");

(75, "Aniket", "2072-10-01", "2022-09-01", 10000, "male");

(76, "Pratik", "2073-04-01", "2022-09-01", 10000, "male");

(77, "Rishabh", "2074-08-01", "2022-09-01", 10000, "male");

(78, "Aayush", "2075-12-01", "2022-09-01", 10000, "male");

(79, "Vishal", "2076-06-01", "2022-09-01", 10000, "male");

(80, "Akash", "2077-10-01", "2022-09-01", 10000, "male");

(81, "Aniket", "2078-04-01", "2022-09-01", 10000, "male");

(82, "Pratik", "2079-08-01", "2022-09-01", 10000, "male");

(83, "Rishabh", "2080-12-01", "2022-09-01", 10000, "male");

(84, "Aayush", "2081-06-01", "2022-09-01", 10000, "male");

(85, "Vishal", "2082-10-01", "2022-09-01", 10000, "male");

(86, "Akash", "2083-04-01", "2022-09-01", 10000, "male");

(87, "Aniket", "2084-08-01", "2022-09-01", 10000, "male");

(88, "Pratik", "2085-12-01", "2022-09-01", 10000, "male");

(89, "Rishabh", "2086-06-01", "2022-09-01", 10000, "male");

(90, "Aayush", "2087-10-01", "2022-09-01", 10000, "male");

(91, "Vishal", "2088-04-01", "2022-09-01", 10000, "male");

(92, "Akash", "2089-08-01", "2022-09-01", 10000, "male");

(93, "Aniket", "2090-12-01", "2022-09-01", 10000, "male");

(94, "Pratik", "2091-06-01", "2022-09-01", 10000, "male");

(95, "Rishabh", "2092-10-01", "2022-09-01", 10000, "male");

(96, "Aayush", "2093-04-01", "2022-09-01", 10000, "male");

(97, "Vishal", "2094-08-01", "2022-09-01", 10000, "male");

(98, "Akash", "2095-12-01", "2022-09-01", 10000, "male");

(99, "Aniket", "2096-06-01", "2022-09-01", 10000, "male");

(100, "Pratik", "2097-10-01", "2022-09-01", 10000, "male");

(101, "Rishabh", "2098-04-01", "2022-09-01", 10000, "male");

(102, "Aayush", "2099-08-01", "2022-09-01", 10000, "male");

(103, "Vishal", "20100-12-01", "2022-09-01", 10000, "male");

(104, "Akash", "20101-06-01", "2022-09-01", 10000, "male");

(105, "Aniket", "20102-10-01", "2022-09-01", 10000, "male");

(106, "Pratik", "20103-04-01", "2022-09-01", 10000, "male");

(107, "Rishabh", "20104-08-01", "2022-09-01", 10000, "male");

(108, "Aayush", "20105-12-01", "2022-09-01", 10000, "male");

(109, "Vishal", "20106-06-01", "2022-09-01", 10000, "male");

(110, "Akash", "20107-10-01", "2022-09-01", 10000, "male");

(111, "Aniket", "20108-04-01", "2022-09-01", 10000, "male");

(112, "Pratik", "20109-08-01", "2022-09-01", 10000, "male");

(113, "Rishabh", "20110-12-01", "2022-09-01", 10000, "male");

(114, "Aayush", "20111-06-01", "2022-09-01", 10000, "male");

(115, "Vishal", "20112-10-01", "2022-09-01", 10000, "male");

(116, "Akash", "20113-04-01", "2022-09-01", 10000, "male");

(117, "Aniket", "20114-08-01", "2022-09-01", 10000, "male");

(118, "Pratik", "20115-12-01", "2022-09-01", 10000, "male");

(119, "Rishabh", "20116-06-01", "2022-09-01", 10000, "male");

(120, "Aayush", "20117-10-01", "2022-09-01", 10000, "male");

(121, "Vishal", "20118-04-01", "2022-09-01", 10000, "male");

(122, "Akash", "20119-08-01", "2022-09-01", 10000, "male");

(123, "Aniket", "20120-12-01", "2022-09-01", 10000, "male");

(124, "Pratik", "20121-06-01", "2022-09-01", 10000, "male");

(125, "Rishabh", "20122-10-01", "2022-09-01", 10000, "male");

(126, "Aayush", "20123-04-01", "2022-09-01", 10000, "male");

(127, "Vishal", "20124-08-01", "2022-09-01", 10000, "male");

(128, "Akash", "20125-12-01", "2022-09-01", 10000, "male");

(129, "Aniket", "20126-06-01", "2022-09-01", 10000, "male");

(130, "Pratik", "20127-10-01", "2022-09-01", 10000, "male");

(131, "Rishabh", "20128-04-01", "2022-09-01", 10000, "male");

(132, "Aayush", "20129-08-01", "2022-09-01", 10000, "male");

(133, "Vishal", "20130-12-01", "2022-09-01", 10000, "male");

(134, "Akash", "20131-06-01", "2022-09-01", 10000, "male");

(135, "Aniket", "20132-10-01", "2022-09-01", 10000, "male");

(136, "Pratik", "20133-04-01", "2022-09-01", 10000, "male");

(137, "Rishabh", "20134-08-01", "2022-09-01", 10000, "male");

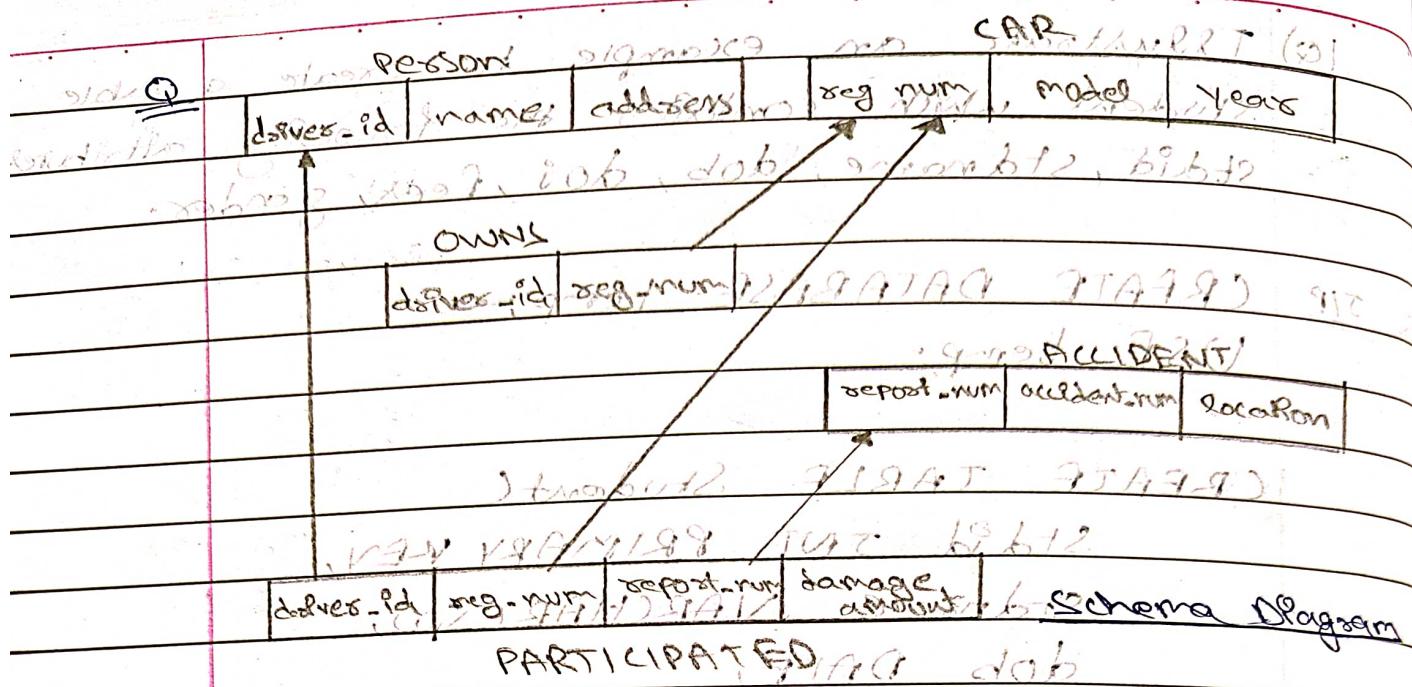
(138, "Aayush", "20135-12-01", "2022-09-01", 10000, "male");

(139, "Vishal", "20136-06-01", "2022-09-01", 10000, "male");

(140, "Akash", "20137-10-01", "2022-09-01", 10000, "male");</p

## LAB-2

Date \_\_\_\_\_  
Page \_\_\_\_\_



- (Q) update the damage amount to 125000 for the car with a specific reg. num'l 'KA05 3408' for which the accident report number was

- (b) UPDATE path & packed DATA TRAIN

10-80 -SBPL damage amount = 25000/-

WHERE reg\_num = 'KA053408' AND

"80-01-005" "deposit-number"; "subsequent"; "2" ("2011-0002")

- (E) Add a "new" accident "to the database"

- (d) INSERT INTO accident VALUES

(16, "2008-03-15", "Domus");

2 November 1987 \* T 07132

- (e) Display accident date and location.

~~Entnahmest~~ entnahm ab - abwimmeln 1912 810

- (f) SELECT accident\_date, location FROM

~~9-10-19~~ FROM accident: SI-10-PAPL after 10-10-81

Strong T + 900.0P 21-10-8200 22-21-1000C + 000.0P

- (g) Display drivers id who did the accident damage greater than or equal to Rs 25000

→ SELECT delivery\_order\_id, damage\_amount FROM ~~paid\_for~~ pending\_order WHERE damage\_amount >= 25000;

• Found 2 rows average damage amount.

→ SELECT avg(damage\_amount) FROM (SELECT damage\_amount FROM pending\_order WHERE damage\_amount >= 25000) AS T1;

• Average damage amount = 30000

• Standard deviation =  $\sqrt{\frac{1}{n} \sum (x_i - \bar{x})^2}$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

• Standard deviation =  $\sqrt{\frac{1}{2} (10000 + 10000)} = \sqrt{10000} = 100$

- List the entire participated relation in the descending order of damage amount.

→ `SELECT * FROM participated`

• ORDER BY damage\_amount DESC;

- find the average damage amount

→ `SELECT AVG(damage_amount) FROM participated;`

- find the max damage amount

→ `SELECT MAX(damage_amount) FROM participated;`

- List the name of drivers whose damage is greater than Avg damage Amount.

→ `SELECT DISTINCT name, damage_amount`

`FROM person p, participated pp`

`WHERE pp.driver_id = p.driver_id`

`AND pp.damage_amount > (SELECT AVG(damage_amount)`

`FROM participated);`

- Display the entire CAR relation in the ascending order of manufacturing year.

→ `SELECT * FROM car`

`ORDER BY year ASC;`

- Find the number of accidents in which cars belonging to a specific model were involved.

→ SELECT count(report\_num)

~~FROM car c, passed Spotted p~~

WHERE C.xog-num = P.xog-num and  
C.model = "Janlex";

- Find the total numbers of people who owned cars that were involved in accidents in 2008.

$\rightarrow \text{SELECT count(DISTINCT driver_id)}$

FROM participated in accident

~~WHERE p.report\_num = a.report\_num~~

AND accident\_date LIKE "2008-10%", (6)

- Delete the tuple where damage amount is below the avg damage amount

→ DELETE FROM participated WHERE ...

WHERE damage amount >= 33 000;

Chlorophyll a fluorescence - Photosynthesis = TEP

• The Second Part - Annotations by T.S. Eliot

000001 1999-01-01 00000000000000000000000000000000

00002 1998-0000015 200

6.  $\lim_{x \rightarrow 0} \frac{\sin x}{x}$  (Ans: 1)

Digitized by srujanika@gmail.com

Digitized by srujanika@gmail.com

2000 1000 500 250 125 62.5 31.25

• E. L. (LAWRENCE LOWELL) MERRILL

## LAB-4

Date   /  /    
Page

Customer Name	Branch Name	Branch City	Assets
Accno	Branch name	balance	customer's name
1000	Branch 1	10000	Customer 1
2000	Branch 2	20000	Customer 2
3000	Branch 3	30000	Customer 3
4000	Branch 4	40000	Customer 4
5000	Branch 5	50000	Customer 5
6000	Branch 6	60000	Customer 6
7000	Branch 7	70000	Customer 7
8000	Branch 8	80000	Customer 8
9000	Branch 9	90000	Customer 9
10000	Branch 10	100000	Customer 10

(Q) Display the branch names and assets from all branches in banks of super and rename the assets column to "assets in banks".

I/P SELECT branch\_name, address as 'address in branch'  
FROM branch;

O/P | branch\_name ~~branch\_name~~ exists in fetch

SBI - janitorial 20000

SBI - Parliament Road 10000

SBJ. ResidencyRoad 10000

SRJ Sivagiri Road 20000

~~SBJ Chamarsaj post~~ 50000

(Q) Find all customers who have at least two account at the same branch (ex: SBI - Beliendy Road)

TIP | SELECT all no, branch name

FROM branch account

Group By branch name

HAVING count(branch\_name) > 1;

OLP | acc no branch name

SBT - Residential Roads

3 SBI - SWARI Road

9

SBT - Parliament road

5 SBS - Säntarmanta.

- (Q) Create a view which gives each branch the sum of amount of all loans at the branch.

I/P CREATE view insert sum

AS SELECT branch\_name, account

FROM Joan

GROUP BY branch name;

~~SELECT \* FROM loan sum;~~

O/P | branch\_name | account

SBJ - charansing Peet 1900

SBS - Residential Road 2000

SB5 - Savari Road Extension 3000 ft No 189 (6)

SBS - Parliament Road 4000 m² site no.

SRS - Januszmaria 5000

Kennerly, 1997, owner, small, T9170-912

~~2000-01-08~~ (S.M. Thorne-Preston) 297H

Digitized by srujanika@gmail.com

1 Page 90 : 910

*Monica and Thomas are also awaiting adoption.*

the Laramie, 20 miles south of Cheyenne, the last big town.

• Wetlands, floodplains & coastal areas also feed

19. *Leucosia* *leucostoma* *Leucosia leucostoma*

## Answers to the Document Model

LAB-5

Date \_\_\_\_\_  
Page \_\_\_\_\_

15/04/2024

Branch name	Branch city	Address
Bank Account 1	L	E
Acc no	Branch name	Balance
Bank Account 2	P	
Customer name	Customer name	Acc no
Customer name	Customer street	City
Customer name	Branch name	Amount
Loan number	Branch name	Amount
Customer name	Customer name	Loan number
Customer name	Branch name	Borrower?
Customer name	Branch name	Branch name

- (Q) Find all the customers who have an account at all the branches located in a specific city (e.g. Delhi).

IIP SELECT customer\_name FROM bankcustomer  
WHERE customer\_city = 'Delhi';

Off | Niko )

Ravi

- (④) find all customers who have a loan at the bank but do not have an account.

EMP    SELECT D. Customer names

~~FROM borrowed B~~

~~WHERE R.Customer-name NOT IN (~~

Q1. SELECT Customer\_name FROM  
depositors WHERE  
Customer\_name = 'John'

O/P Customer\_name  
John  
Moham  
Alka  
Anita

(Q) Find the names of all branches that have greater assets than all branches located in Bangalore.

O/P Branch\_name  
Bengaluru

I/P SELECT branch\_name FROM branch WHERE  
assets > (SELECT assets FROM branch

WHERE branch\_city = 'Bangalore')

Demo: Customer with max assets in Bangalore

O/P branch\_name  
SRIS-achanajipet-001

SRIS-Santacruz-002

SRI-Sivrajg road-003

Demo: Customer with min assets in Bangalore

(Q) Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Mumbai)

DEMO: Customer with min assets in Bangalore

I/P DELETE FROM bankcustomer

WHERE customer\_city = 'Bangalore';

(Q) Update the balance of all accounts by 5%

I/P UPDATE bankaccount

SET balance = balance \* 1.05;

SELECT \* FROM bankaccount;

Output:

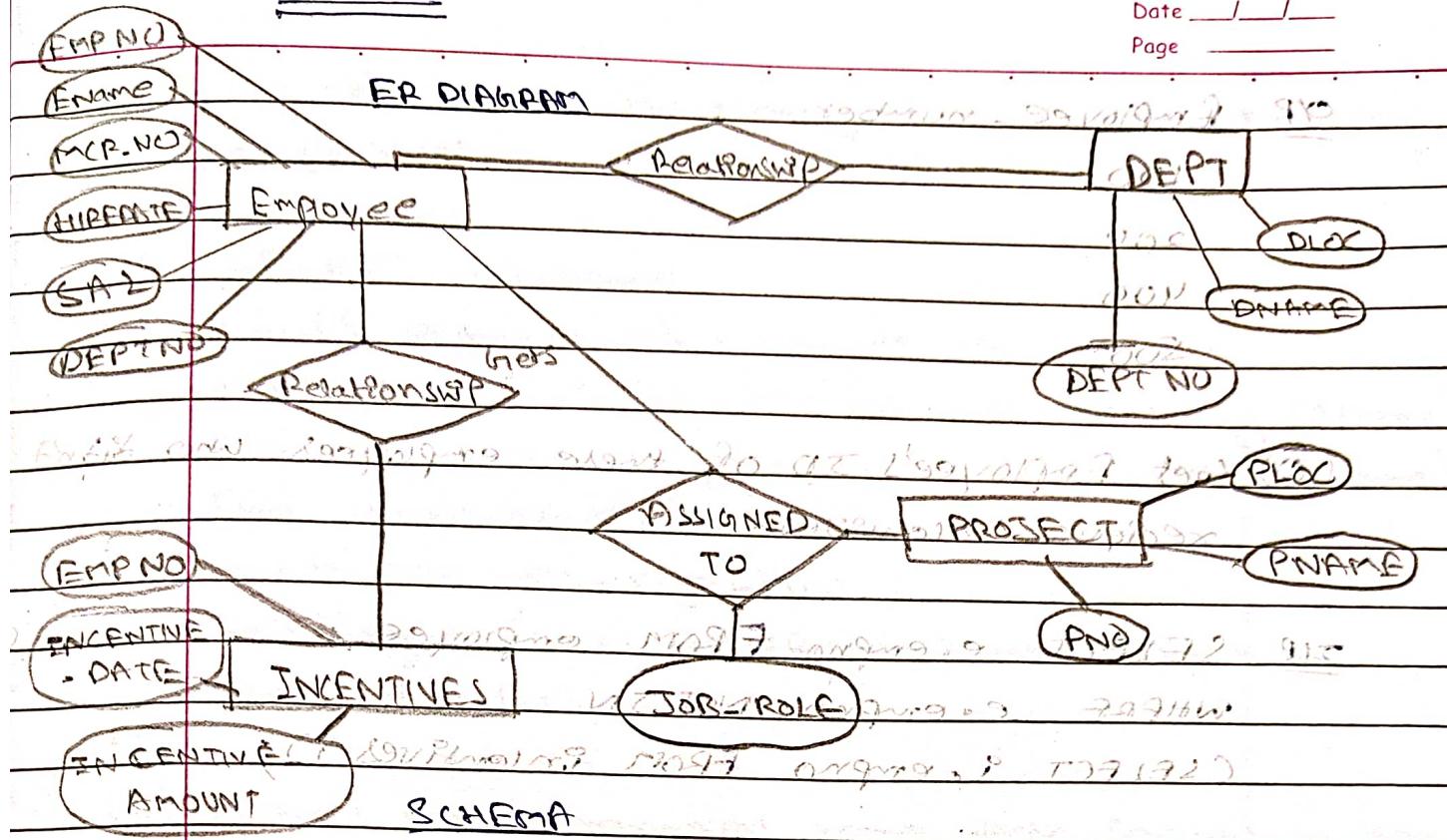
acc_no	branch_name	balnace
1	SBI - chamarajpet	22050
2	SBI - Residency road	5512.5
3	SBI - shivaji road	6615
4	SBI - Parliament Road	99220.5
5	SBI - Santarmanta	88201
6	SBI - signal road	4410
8	SBI - Residency road	4410 (0)
9	SBI - Parliament Road	33070.5
10	SBI - Residency Road	5512.5

(Q) find all customers who have both an account and a loan at the Bangalore branch.

SQL  
~~SELECT distinct D.customername FROM Depositor D WHERE D.customername IN (~~  
~~SELECT D.customername from branch b~~  
~~Depositor D, bankaccount ba WHERE~~  
~~ba.branch\_city = 'Bangalore' and~~  
~~b.branch\_name = ba.branch\_name and~~  
~~ba.acno = D.acno and customername IN (~~  
~~SELECT customername from Borrower)~~  
~~);~~

O/P SBI - Mantisnagar for account no 10001 (0)

## ER DIAGRAM



## INCENTIVES

EMPNO
INCENTIVE-DATE
INCENTIVE-AMOUNT

OR Ques 8.10

Q8 DEPT

DEPTNO
DNAME
LOC

## PROJECT

## EMPLOYEE

PNO
PLLOC
PNAME

EMPNO
Ename
MUR-NO
HIREDATE
SAL
DEPTNO (PK)

## ASSIGNED-TO

EMPNO ON JOBSITE OR - Ques 8.10

PNO, PLLOC, DNAME, LOC, DEPTNO

PROJ. NO. ON SITE, PLLOC, DNAME, LOC, DEPTNO

(Q) Retrieved the employee numbers of all employees

who work on project located in Bengaluru,

Hyderabad or Mysore. (Ans 8.10)

SQL SELECT proj.PNO, Employee.EMPNO FROM Project P,  
Employee E WHERE Employee.PNO = P.PNO AND P.PLLOC IN ('Mysore',

('Hyderabad', 'Bengaluru', 'Mysore'));

O/P Employee-number

100

200

400

500

(Q) Get Employee's ID of those employees who didn't receive incentives.

I/P SELECT e.empno FROM employee e

WHERE e.empno NOT IN

(SELECT i.empno FROM Incentives i);

O/P empno,

700

300

(Q) Write a SQL query to find the employee name, number, dept, job-role, department, location and project location who are working for a project location same as his/her department location.

I/P SELECT e.name Emp-name, e.empno Emp-number, d.dname Dept, a.job-role Job-role, d.loc Department Location, p.place Project Location FROM Employee e, Project p, Dept d, Employee e, AssignedTo at WHERE e.empno = a.empno and p.no = a.pno and e.deptno = d.deptno and p.place = d.loc;

Emp-Name	Emp-Number	Dept	Job-Role	Department	Project Location
Ramya P	100	Sales	Project Manager	Bengaluru	Bengaluru

(Q) List the name of managers with the maximum employees.

Answer: Manager with maximum employees

IP

```
SELECT e1.name (Manager with max employee)
FROM employee e1, employee e2
WHERE e1.empno = e2.mgr_no + group by e1.name
HAVING count(e1.mgr_no) = (SELECT count(e1.name)
FROM employee e1, employee e2 WHERE
e1.empno = e2.mgr_no + group by e1.name ORDER BY count(e1.name)
desc LIMIT 1);
```

(Q) Display those managers name whose salary is more than average salary of all employees.

IP

```
SELECT m.name FROM employee m
WHERE m.empno IN (SELECT m.mgr_no
SELECT mgr_no FROM employee)
and m.sal > (SELECT avg(m.sal) FROM employee m
WHERE m.mgr_no = m.empno);
```

(Q) Find the name of the second top level managers of each department.

IP

```
SELECT name FROM employee WHERE empno IN
(SELECT distinct mgr_no)
FROM employee WHERE empno IN
(SELECT distinct mgr_no FROM employee WHERE
empno IN (SELECT distinct mgr_no FROM employee)));
```

(Q) Find the employee details who got second maximum incentive in January 2019.

Q18

SELECT \* FROM employee WHERE empno = 101

(SELECT i.empno FROM incentives i WHERE

i.incentive\_amount = (SELECT max

c.incentive\_amount) FROM incentives c WHERE

c.incentive\_amount < (SELECT max (i.incentive

amount) FROM incentives i) AND i.incentive

incentive\_data between '2019-01-01' AND

'2018-12-31') AND incentive\_data BETWEEN

('2019-01-01' AND '2019-12-31'))

(Q) Pay those employees who are working in the  
same department where his manager is working  
and the year salary increment starts 101928 (P)

Q18 SELECT gender, name FROM employee

FROM employee e1, employee e2

WHERE e1.empno = e2.empno AND e1.deptno = e2.deptno

e1.deptno = e2.deptno AND name IN ('JADEH')

IT IS VARIOUS THINGS (DEPARTMENT NUMBER 101928)

DEPARTMENT NUMBER 101928 IT IS VARIOUS THINGS

IT IS VARIOUS THINGS IT IS VARIOUS THINGS

(a) A SUPPLIER  $P_{SUPP}$  is assigned to a number of parts. (Q)

Sid	Sname	City	Line	Pid	Pname	Color
1	ABC	Delhi	1	1	Red	Red
2	DEF	Mumbai	2	2	Blue	Blue
3	GHI	Bangalore	3	3	Green	Green
4	JKL	Kolkata	4	4	Yellow	Yellow
5	MNO	Chennai	5	5	Black	Black
6	PQR	Hyderabad	6	6	White	White
7	STU	Jaipur	7	7	Orange	Orange
8	VWX	Surat	8	8	Pink	Pink
9	YZK	Delhi	9	9	Grey	Grey
10	LMN	Mumbai	10	10	Red	Red
11	OPQ	Bangalore	11	11	Blue	Blue
12	RST	Kolkata	12	12	Green	Green
13	UVW	Chennai	13	13	Yellow	Yellow
14	JKL	Hyderabad	14	14	White	White
15	STU	Jaipur	15	15	Orange	Orange
16	VWX	Surat	16	16	Pink	Pink
17	YZK	Delhi	17	17	Grey	Grey
18	LMN	Mumbai	18	18	Red	Red
19	OPQ	Bangalore	19	19	Blue	Blue
20	RST	Kolkata	20	20	Green	Green
21	UVW	Chennai	21	21	Yellow	Yellow
22	JKL	Hyderabad	22	22	White	White
23	STU	Jaipur	23	23	Orange	Orange
24	VWX	Surat	24	24	Pink	Pink
25	YZK	Delhi	25	25	Grey	Grey
26	LMN	Mumbai	26	26	Red	Red
27	OPQ	Bangalore	27	27	Blue	Blue
28	RST	Kolkata	28	28	Green	Green
29	UVW	Chennai	29	29	Yellow	Yellow
30	JKL	Hyderabad	30	30	White	White
31	STU	Jaipur	31	31	Orange	Orange
32	VWX	Surat	32	32	Pink	Pink
33	YZK	Delhi	33	33	Grey	Grey
34	LMN	Mumbai	34	34	Red	Red
35	OPQ	Bangalore	35	35	Blue	Blue
36	RST	Kolkata	36	36	Green	Green
37	UVW	Chennai	37	37	Yellow	Yellow
38	JKL	Hyderabad	38	38	White	White
39	STU	Jaipur	39	39	Orange	Orange
40	VWX	Surat	40	40	Pink	Pink
41	YZK	Delhi	41	41	Grey	Grey
42	LMN	Mumbai	42	42	Red	Red
43	OPQ	Bangalore	43	43	Blue	Blue
44	RST	Kolkata	44	44	Green	Green
45	UVW	Chennai	45	45	Yellow	Yellow
46	JKL	Hyderabad	46	46	White	White
47	STU	Jaipur	47	47	Orange	Orange
48	VWX	Surat	48	48	Pink	Pink
49	YZK	Delhi	49	49	Grey	Grey
50	LMN	Mumbai	50	50	Red	Red
51	OPQ	Bangalore	51	51	Blue	Blue
52	RST	Kolkata	52	52	Green	Green
53	UVW	Chennai	53	53	Yellow	Yellow
54	JKL	Hyderabad	54	54	White	White
55	STU	Jaipur	55	55	Orange	Orange
56	VWX	Surat	56	56	Pink	Pink
57	YZK	Delhi	57	57	Grey	Grey
58	LMN	Mumbai	58	58	Red	Red
59	OPQ	Bangalore	59	59	Blue	Blue
60	RST	Kolkata	60	60	Green	Green
61	UVW	Chennai	61	61	Yellow	Yellow
62	JKL	Hyderabad	62	62	White	White
63	STU	Jaipur	63	63	Orange	Orange
64	VWX	Surat	64	64	Pink	Pink
65	YZK	Delhi	65	65	Grey	Grey
66	LMN	Mumbai	66	66	Red	Red
67	OPQ	Bangalore	67	67	Blue	Blue
68	RST	Kolkata	68	68	Green	Green
69	UVW	Chennai	69	69	Yellow	Yellow
70	JKL	Hyderabad	70	70	White	White
71	STU	Jaipur	71	71	Orange	Orange
72	VWX	Surat	72	72	Pink	Pink
73	YZK	Delhi	73	73	Grey	Grey
74	LMN	Mumbai	74	74	Red	Red
75	OPQ	Bangalore	75	75	Blue	Blue
76	RST	Kolkata	76	76	Green	Green
77	UVW	Chennai	77	77	Yellow	Yellow
78	JKL	Hyderabad	78	78	White	White
79	STU	Jaipur	79	79	Orange	Orange
80	VWX	Surat	80	80	Pink	Pink
81	YZK	Delhi	81	81	Grey	Grey
82	LMN	Mumbai	82	82	Red	Red
83	OPQ	Bangalore	83	83	Blue	Blue
84	RST	Kolkata	84	84	Green	Green
85	UVW	Chennai	85	85	Yellow	Yellow
86	JKL	Hyderabad	86	86	White	White
87	STU	Jaipur	87	87	Orange	Orange
88	VWX	Surat	88	88	Pink	Pink
89	YZK	Delhi	89	89	Grey	Grey
90	LMN	Mumbai	90	90	Red	Red
91	OPQ	Bangalore	91	91	Blue	Blue
92	RST	Kolkata	92	92	Green	Green
93	UVW	Chennai	93	93	Yellow	Yellow
94	JKL	Hyderabad	94	94	White	White
95	STU	Jaipur	95	95	Orange	Orange
96	VWX	Surat	96	96	Pink	Pink
97	YZK	Delhi	97	97	Grey	Grey
98	LMN	Mumbai	98	98	Red	Red
99	OPQ	Bangalore	99	99	Blue	Blue
100	RST	Kolkata	100	100	Green	Green
101	UVW	Chennai	101	101	Yellow	Yellow
102	JKL	Hyderabad	102	102	White	White
103	STU	Jaipur	103	103	Orange	Orange
104	VWX	Surat	104	104	Pink	Pink
105	YZK	Delhi	105	105	Grey	Grey
106	LMN	Mumbai	106	106	Red	Red
107	OPQ	Bangalore	107	107	Blue	Blue
108	RST	Kolkata	108	108	Green	Green
109	UVW	Chennai	109	109	Yellow	Yellow
110	JKL	Hyderabad	110	110	White	White
111	STU	Jaipur	111	111	Orange	Orange
112	VWX	Surat	112	112	Pink	Pink
113	YZK	Delhi	113	113	Grey	Grey
114	LMN	Mumbai	114	114	Red	Red
115	OPQ	Bangalore	115	115	Blue	Blue
116	RST	Kolkata	116	116	Green	Green
117	UVW	Chennai	117	117	Yellow	Yellow
118	JKL	Hyderabad	118	118	White	White
119	STU	Jaipur	119	119	Orange	Orange
120	VWX	Surat	120	120	Pink	Pink
121	YZK	Delhi	121	121	Grey	Grey
122	LMN	Mumbai	122	122	Red	Red
123	OPQ	Bangalore	123	123	Blue	Blue
124	RST	Kolkata	124	124	Green	Green
125	UVW	Chennai	125	125	Yellow	Yellow
126	JKL	Hyderabad	126	126	White	White
127	STU	Jaipur	127	127	Orange	Orange
128	VWX	Surat	128	128	Pink	Pink
129	YZK	Delhi	129	129	Grey	Grey
130	LMN	Mumbai	130	130	Red	Red
131	OPQ	Bangalore	131	131	Blue	Blue
132	RST	Kolkata	132	132	Green	Green
133	UVW	Chennai	133	133	Yellow	Yellow
134	JKL	Hyderabad	134	134	White	White
135	STU	Jaipur	135	135	Orange	Orange
136	VWX	Surat	136	136	Pink	Pink
137	YZK	Delhi	137	137	Grey	Grey
138	LMN	Mumbai	138	138	Red	Red
139	OPQ	Bangalore	139	139	Blue	Blue
140	RST	Kolkata	140	140	Green	Green
141	UVW	Chennai	141	141	Yellow	Yellow
142	JKL	Hyderabad	142	142	White	White
143	STU	Jaipur	143	143	Orange	Orange
144	VWX	Surat	144	144	Pink	Pink
145	YZK	Delhi	145	145	Grey	Grey
146	LMN	Mumbai	146	146	Red	Red
147	OPQ	Bangalore	147	147	Blue	Blue
148	RST	Kolkata	148	148	Green	Green
149	UVW	Chennai	149	149	Yellow	Yellow
150	JKL	Hyderabad	150	150	White	White
151	STU	Jaipur	151	151	Orange	Orange
152	VWX	Surat	152	152	Pink	Pink
153	YZK	Delhi	153	153	Grey	Grey
154	LMN	Mumbai	154	154	Red	Red
155	OPQ	Bangalore	155	155	Blue	Blue
156	RST	Kolkata	156	156	Green	Green
157	UVW	Chennai	157	157	Yellow	Yellow
158	JKL	Hyderabad	158	158	White	White
159	STU	Jaipur	159	159	Orange	Orange
160	VWX	Surat	160	160	Pink	Pink
161	YZK	Delhi	161	161	Grey	Grey
162	LMN	Mumbai	162	162	Red	Red
163	OPQ	Bangalore	163	163	Blue	Blue
164	RST	Kolkata	164	164	Green	Green
165	UVW	Chennai	165	165	Yellow	Yellow
166	JKL	Hyderabad	166	166	White	White
167	STU	Jaipur	167	167	Orange	Orange
168	VWX	Surat	168	168	Pink	Pink
169	YZK	Delhi	169	169	Grey	Grey
170	LMN	Mumbai	170	170	Red	Red
171	OPQ	Bangalore	171	171	Blue	Blue
172	RST	Kolkata	172	172	Green	Green
173	UVW	Chennai	173	173	Yellow	Yellow
174	JKL	Hyderabad	174	174	White	White
175	STU	Jaipur	175	175	Orange	Orange
176	VWX	Surat	176	176	Pink	Pink
177	YZK	Delhi	177	177	Grey	Grey
178	LMN	Mumbai	178	178	Red	Red
179	OPQ	Bangalore	179	179	Blue	Blue
180	RST	Kolkata	180	180	Green	Green
181	UVW	Chennai	181	181	Yellow	Yellow
182	JKL	Hyderabad	182	182	White	White
183	STU	Jaipur	183	183	Orange	Orange
184	VWX	Surat	184	184	Pink	Pink
185	YZK	Delhi	185	185	Grey	Grey
186	LMN	Mumbai	186	186	Red	Red
187	OPQ	Bangalore	187	187	Blue	Blue
188	RST	Kolkata	188	188	Green	Green
189	UVW	Chennai	189	189	Yellow	Yellow
190	JKL	Hyderabad	190	190	White	White
191	STU	Jaipur	191	191	Orange	Orange
192	VWX	Surat	192	192	Pink	Pink
193	YZK	Delhi	193	193	Grey	Grey
194	LMN	Mumbai	194	194	Red	Red
195	OPQ	Bangalore	195	195	Blue	Blue
196	RST	Kolkata	196	196	Green	Green
197	UVW	Chennai	197	197	Yellow	Yellow
198	JKL	Hyderabad	198	198	White	White
199	STU	Jaipur	199	199	Orange	Orange
200	VWX	Surat	200	200	Pink	Pink
201	YZK	Delhi	201	201	Grey	Grey
202	LMN	Mumbai	202	202	Red	Red
203	OPQ	Bangalore	203	203	Blue	Blue
204	RST	Kolkata	204	204	Green	Green
205	UVW	Chennai	205	205	Yellow	Yellow
206	JKL	Hyderabad	206	206	White	White
207	STU	Jaipur	207			

(Q) find the names of parts supplied by Acme widget suppliers and by no one else.

IP  $\text{SELECT } p.pname}$   
 $\text{FROM parts } p, \text{catalog } c, \text{Supplier } s$   
 $\text{WHERE } p.pid = c.pid \text{ and } c.sid = s.sid$   
 $\text{and } s.sname = "Acme widget" \text{ and NOT}$   
 $\text{Exists } (\text{SELECT } * \text{ FROM catalog } c1, \text{Supplier } s1$   
 ~~$\text{WHERE } c1.pid = c.pid \text{ and } c1.sid \neq s.sid \text{ and }$~~   
 $s1.sname = "Acme widget")$

(Q) find the sid of suppliers who charge more  
 for some part than the average cost of that  
 part.

IP  $\text{SELECT distinct sid } c.sid$  ~~FROM catalog } (Q)~~  
 $\text{WHERE } c.cost > (\text{SELECT avg(c2.cost)}$   
 $\text{FROM catalog } c1 \text{ WHERE } c1.pid = c.pid);$

(Q) for each part, find the sname of supplier who  
 charges the most for that part - this line

IP  $\text{SELECT } p.pid, s.sname}$   
 $\text{FROM parts } p, \text{Supplier } s, \text{catalog } c$   
 $\text{WHERE } c.pid = p.pid \text{ and } c.sid = s.sid \text{ and }$   
 $c.cost = (\text{SELECT max(c1.cost)})$  ~~FROM catalog } (Q)~~  
 $\text{FROM catalog } c1$   
 $\text{WHERE } c1.pid = p.pid;$

Perform MongoDB Operations using mongoDB.

1) Create a database "student" with the following attributes : rollno, Age, contact no,

Email id and gender (M/F).

(1) Create a database "student" with the following attributes : rollno, Age, contact no, Email id and gender (M/F).

(2) Insert appropriate values.

(3) Write & query new update command of a student with roll no 10, Age 18, Name "Rajesh" & gender M.

(4) Replace the student name "from" "ABC" (To "FEM" ob roll no. 33 "Name" cannot be 28)

(5) Export the generated tables into local file system

(6) Drop the table & schema program

(7) Import a given CSV data set from local file system into mongoDB collection.

I/P 1) db.createCollection("student");

2) db.student.insert({rollno: 1, Age: 21, cont:

email: "antara.deo@gmail.com"});

db.student.insert({rollno: 2, Age: 22,

cont: 9876, email: "antara9@gmail.com"}, {

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100});

3) db.student.insert({rollno: 3, Age: 23, cont: 5545},

email: "anubhav.deo@gmail.com"});

db. Student.insert({ RollNo: 4, Age: 20, Contact: 9977889900, Email: "paul.de9@gmail.com" })

db. Student.insert({ RollNo: 10, Age: 23, Contact: 8877665544, Email: "rephag-deg@gmail.com" })

db. Student.insert({ "name": "tom", "data": 1 })

3) db. Student.update({ RollNo: 10 }, { \$set: { Email: "Abhinav@gmail.com" } })

4) db. Student.update({ RollNo: 11, name: "ABC" }, { \$set: { name: "FEM" } })

5) Execute at command prompt:

mongoclient mongo://127.0.0.1:27017/test  
student -> db.getCollection("student").find().pretty()

6) FROM: ("Mongo Single 10.0.11.0") ORACLE DB (10.2)

mongoclient student.insert({ "name": "tom", "data": 1 })

(7) At command prompt:

mongorestore --db=student --port=27017  
--collection=new\_student\_type  
--dbpath=c:\users\bmscse\Downloads\output.json  
--drop

Q-1) Create a collection by name `customers` with the following attributes = `cust_id`, `acc_bal`, `acc_type`.

1) `db.createCollection("customers")`

Q-2) Insert at least 15 values in the collection.

`customers.insert([{"_id": "1", "acc_type": "2", "acc_bal": 1000}, {"_id": "2", "acc_type": "2", "acc_bal": 1500}, {"_id": "3", "acc_type": "2", "acc_bal": 1000}, {"_id": "4", "acc_type": "2", "acc_bal": 1300}, {"_id": "5", "acc_type": "2", "acc_bal": 800}, {"_id": "6", "acc_type": "2", "acc_bal": 2000}, {"_id": "7", "acc_type": "2", "acc_bal": 1000}, {"_id": "8", "acc_type": "2", "acc_bal": 1500}, {"_id": "9", "acc_type": "2", "acc_bal": 1000}, {"_id": "10", "acc_type": "2", "acc_bal": 1300}, {"_id": "11", "acc_type": "2", "acc_bal": 800}, {"_id": "12", "acc_type": "2", "acc_bal": 2000}, {"_id": "13", "acc_type": "2", "acc_bal": 1000}, {"_id": "14", "acc_type": "2", "acc_bal": 1500}, {"_id": "15", "acc_type": "2", "acc_bal": 1000}])`

Q-3) Write a query to display those records whose total account balance is greater than 1200 & account type '2' for each `customer_id`.

Q-4) Determine minimum and maximum account balance for each customer\_id.

`db.customers.aggregate([{"$group": {"_id": "$cust_id", "min": {"$min": "$acc_bal"}, "max": {"$max": "$acc_bal"}}}])`

Q-5) Report the created collection into local file system

Q-6) Drop the table.

Q-7) Import a given CSV dataset from local file system into mongoDB collection.

Q-8) 1) `db.createCollection("customers")`

`db.customers.insert([{"_id": "1", "acc_type": "2", "acc_bal": 1000}, {"_id": "2", "acc_type": "2", "acc_bal": 1500}, {"_id": "3", "acc_type": "2", "acc_bal": 1000}, {"_id": "4", "acc_type": "2", "acc_bal": 1300}, {"_id": "5", "acc_type": "2", "acc_bal": 800}, {"_id": "6", "acc_type": "2", "acc_bal": 2000}, {"_id": "7", "acc_type": "2", "acc_bal": 1000}, {"_id": "8", "acc_type": "2", "acc_bal": 1500}, {"_id": "9", "acc_type": "2", "acc_bal": 1000}, {"_id": "10", "acc_type": "2", "acc_bal": 1300}, {"_id": "11", "acc_type": "2", "acc_bal": 800}, {"_id": "12", "acc_type": "2", "acc_bal": 2000}, {"_id": "13", "acc_type": "2", "acc_bal": 1000}, {"_id": "14", "acc_type": "2", "acc_bal": 1500}, {"_id": "15", "acc_type": "2", "acc_bal": 1000}])`

2) `db.customers.aggregate([{"$group": {"_id": "$cust_id", "avg": {"$avg": "$acc_bal"}, "total": {"$sum": "$acc_bal"}}}])`

`{ "cust_id": "1", "avg": 1000, "total": 1000 }, { "cust_id": "2", "avg": 1500, "total": 1500 }, { "cust_id": "3", "avg": 1000, "total": 1000 }, { "cust_id": "4", "avg": 1300, "total": 1300 }, { "cust_id": "5", "avg": 800, "total": 800 }, { "cust_id": "6", "avg": 2000, "total": 2000 }, { "cust_id": "7", "avg": 1000, "total": 1000 }, { "cust_id": "8", "avg": 1500, "total": 1500 }, { "cust_id": "9", "avg": 1000, "total": 1000 }, { "cust_id": "10", "avg": 1300, "total": 1300 }, { "cust_id": "11", "avg": 800, "total": 800 }, { "cust_id": "12", "avg": 2000, "total": 2000 }, { "cust_id": "13", "avg": 1000, "total": 1000 }, { "cust_id": "14", "avg": 1500, "total": 1500 }, { "cust_id": "15", "avg": 1000, "total": 1000 }`

3) `db.customers.aggregate([`

`{ $match: { acc_type: "2" } },`

`{ $group: { _id: "$cust_id", total_balance: {`

$\sum \$10m$  : " \$ Acc = Bal" } } } }  
 $\sum \$ \text{match}$  :  $\sum \text{total balance} :$   $\sum \$97.1200$  }

4) `db.customers.aggregate([`

- `{ $group: { _id: "$customer_id", minBalance: {`
- `$min: " $acc-bal" } , maxBalance:`
- `$max: " $acc-bal" } ] );`

(5) At command prompt:

~~mongoexport -MongoDB & Srvs // antarctica ("M1")~~  
@ clusters 10. mlnkeys . mongo db . net / oxy db -  
collections = customers - out c. / users / BMSCECSF

Down loads front-pumpision software and targets (2-8)

6) Form mongo shell: . ~~water art shell (2.6)~~  
~~do, (uniformly draped),~~

7) At command prompt:

mongoimport from mongoDB server / antarctic Test  
1234 @ cluster0.monkeys.mongodb.net/  
my DR-collection = NewP (Western) type (from  
file c:\users\rafael\Downloads\output

~~8:30 AM - 10:00 AM 2008 - 1A 10:00 AM 2008 - 1A 10:00 AM 2008 - 1A~~

• What is the best approach to evaluate a hypothesis?

$\frac{d^2f''}{dx^2} \leq M$  for all  $x \in [a, b]$

~~752nd Inf Regt Bn 116 22nd Rgt 69-2 1st Sqdron 1~~