DBMS WORKBOOK

Workbook for V Sem ISE

Prepared by Mr. Bineet Kumar Jha & Dr. Anand R

OVERVIEW & PURPOSE

The workbook is designed to meet the requirements to understand the DBMS concepts through Questionnaire/Quiz.

OBJECTIVES

After solving Workbook Students will be able to-

(i) Understand important terms used in database management domain.
(2) Understand how to model data as entity and attributes
(3) Take a real life scenario and model to solve problems
(4) Differentiate between selection and projection or union or inner join or outer join.

Database Concepts:

Description: *This Section questionnaire the understanding of fundamentals of DBMS Concepts.*

1. A table is a collection of interrelated data stored in form of rows and columns
2. A SQL is a command based, fourth generation language to create, store and manipulate data in a database.
3. Which of the following is not a type of DBMS-
   a. DML
   b. DDL
   c. HDL
   d. None of these

2. Explain the following terms briefly:

a. Attribute: In RDBMS, a table organizes data in rows and columns. The columns are known as attributes whereas the rows are known as records.

b. Domain: A domain is a unique set of values permitted for an attribute in a table.

c. Entity: an entity is a table or attribute of a table in database

d. Relationship: A relationship, in the context of databases, is a situation that exists between two relational database tables when one table has a foreign key that references the primary key of the other table. Relationships allow relational databases to split and store data in different tables, while linking disparate data items.

e. Entity set: An entity set is a group of similar entities and these entities can have attributes.

f. Relationship set: An entity set is a group of similar entities and these entities can have attributes.

g. One-to-many relationship: When a single instance of an entity is associated with more than one instance of another entity then it is called one to many relationship. For example – a customer can place many orders but an order cannot be placed by many customers.

h. Many-to-many relationship: When more than one instance of an entity is associated with more than one instance of another entity then it is called many to many relationship. For example, a student can be assigned to many projects and a project can be assigned to many students.

i. Participation constraint: Participation constraint defines the least number of relationship instances in which an entity must participate.

j. Overlap constraint: An overlap constraint determines whether or not two subclasses can contain the same entity

k. Covering constraint: A covering constraint determines where the entities in the subclasses collectively include all entities in the superclass.

l. Weak entity set: An entity that cannot be identified uniquely without considering some primary key attributes of another identifying owner entity.

m. Aggregation: Aggregation is a process in which a single entity alone is not able to make sense in a relationship so the relationship of two entities acts as one.

n. Role indicator: If an entity set plays more than one role, role indicators describe the different purpose in the relationship.

3. The people who are working in Bank or Railway stations and frequently doing fixed kinds of transactions based on canned queries. Which kind of database user he or she is-
   a) Sophisticated Users
   b) Naïve Users
   c) Parametric users
   d) Casual Users
   e) None of these

4. DBA stands for Database Administrator

5. Why SQL is used to handle structured data. What do you mean by structure data here?

Structured data is the data which conforms to a data model, has a well defined structure, follows a consistent order and can be easily accessed and used by a person or a computer program.

Structured data is usually stored in well-defined schemas such as Databases. It is generally tabular with columns and rows that clearly define its attributes.

SQL (Structured Query language) is often used to manage structured data stored in databases.

Database Design:

Description: *This part refers to the conceptual modeling of the database. As a database designer the first step is to analyze the requirements for mini world (e.g. College or any organization), data collection and modeling followed afterwards. ER diagram and Schema Diagram are the tools to represent the database in the form of entities and relationships among them. There are various notions to represent our design that would have been covered in theoretical courses.*

1. University database contains information about professors (identified by social security number, or SSN) and courses (identified by courseid). Professors teach courses; each of the following situations concerns the Teaches relationship set. For each situation, draw an ER diagram that describes it (assuming no further constraints hold).

a. Professors can teach the same course in several semesters, and each offering must be recorded.

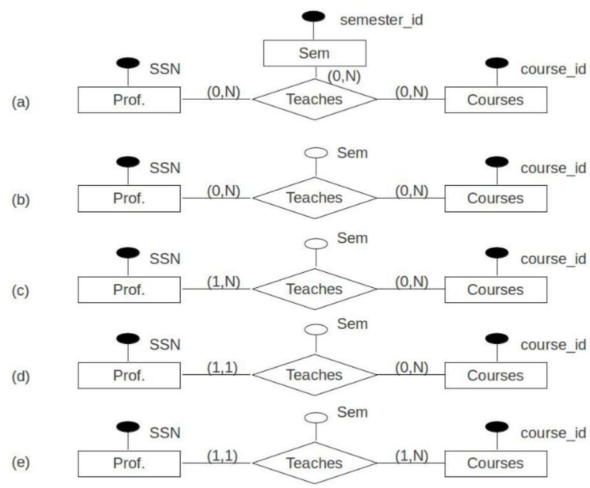
b. Professors can teach the same course in several semesters, and only the most recent such offering needs to be recorded. (Assume this condition applies in all subsequent questions.)

c. Every professor must teach some course.

d. Every professor teaches exactly one course (no more, no less).

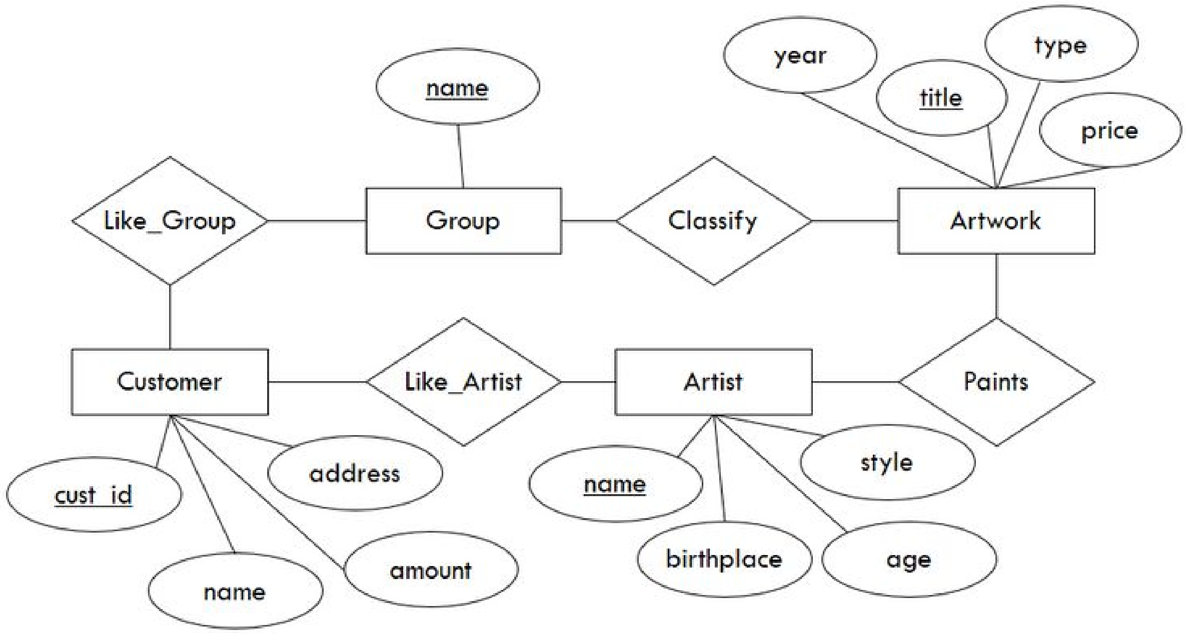
e. Every professor teaches exactly one course (no more, no less), and every course must be taught by some professor.

Ans:



2. Although you always wanted to be an artist, you ended up being an expert on databases because you love to cook data and you somehow confused databases with *data baste*. Your old love is still there, however, so you set up a database company, ArtBase that builds a product for art galleries. The core of this product is a database with a schema that captures all the information that galleries need to maintain. Galleries keep information about artists, their names (which are unique), birthplaces, age, and style of art. For each piece of artwork, the artist, the year it was made, it's unique title, its type of art (e.g., painting, lithograph, sculpture, photograph), and its price must be stored. Pieces of artwork are also classified into groups of various kinds, for example, portraits, still lifes, works by Picasso, or works of the 19th century; a given piece may belong to more than one group. Each group is identified by a name (like those just given) that describes the group. Finally, galleries keep information about customers. For each customer, galleries keep that person's unique name, address, total amount of dollars spent in the gallery (very important!), and the artists and groups of art that the customer tends to like. Draw the ER diagram for the database.

Ans:



Relational Algebra, Calculus and SQL:

Description: *The relational algebra is very important for several reasons: 1. it provides a formal foundation for relational model operations. 2. and perhaps more important, it is used as a basis for implementing and optimizing queries in the query processing and optimization modules that are integral parts of relational database management systems (RDBMSs 3. Some of its concepts are incorporated into the SQL standard query language for RDBMSs.*

1. Consider the following tables to match the table -

Table: CUSTOMER

Customer_Id Customer_Name Customer_City

----------- ------------- -------------

C10100 Steve Agra

C10111	Raghu	Agra
C10115	Chaitanya	Noida
C10117	Ajeet	Delhi
C10118	Carl	Delhi

Table 1: COURSE

Course_Id	Student_Name	Student_Id
C101	Aditya	S901
C104	Aditya	S901
C106	Steve	S911
C109	Paul	S921
C115	Lucy	S931

Table 2: STUDENT

Student_Id	Student_Name	Student_Age
S901	Aditya	19
S911	Steve	18
S921	Paul	19
S931	Lucy	17
S941	Carl	16
S951	Rick	18

1	Display all customers who are from “Agra”	d		a)	∏ Customer_Name, Customer_City (CUSTOMER)
2	Display Name and City of all Customers	a		b)	∏ Student_Name (COURSE) ∩ ∏ Student_Name (STUDENT)
3	Display all the student who are there in both Student and Course table	f		c)	∏ Student_Name (STUDENT) - ∏ Student_Name (COURSE)
4	Display the common Students in both Course and Student tables	b		d)	σ Customer_City="Agra" (CUSTOMER)
5	Display students who are not there in one table but not in other	c		e)	ρ(CUST_NAMES, ∏(Customer_Name)(CUSTOMER))
6	Fetch all the customer names under a new column.	e		f)	∏ Student_Name (COURSE) ∪ ∏ Student_Name (STUDENT)

2. Consider the following tables to answer the given questions-

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

1	Results of SELECT and PROJECT operations. (a) $\sigma_{(Dno=4\ \textbf{AND}\ Salary>25000)\ OR\ (Dno=5\ \textbf{AND}\ Salary>30000)}$(EMPLOYEE). (b) $\pi_{Lname,\ Fname,\ Salary}$(EMPLOYEE). (c) $\pi_{Sex,\ Salary}$(EMPLOYEE).
a)	

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000	888665555	5
Jennifer		Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
Ramesh		Narayan	666884444	1962-09-15	975 FireOak,Humble,TX	M	38000	333445555	5

b)

LNAME	FNAME	SALARY
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

c)

SEX	SALARY
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

2	Write a query in form of relational algebra to count the number of male and female employees working on the projects at “Huston”
	πSex(σ(address=”Hutson”))

3	Write a query in form of relational algebra to Display the employees who work for "Research" Department
	SELECT Fname, Minit, Address FROM EMPLOYEE, DEPARTMENT WHERE Dname='Research' AND Dnumber='Dno'

3. Match the relational algebra operation with corresponding notations-

1	SELECT	c		a)	$\pi_{\text{<attribute list>}}(R)$
2	PROJECT	a		b)	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{(\text{<join attributes 1>}),(\text{<join attributes 2>})} R_2$
3	THETA JOIN	b		c)	$\sigma_{\text{<selection condition>}}(R)$
4	EQUI JOIN	j		d)	$R_1 \cup R_2$
5	NATURAL JOIN	i		e)	$R_1 - R_2$
6	UNION	d		f)	$R_1 \times R_2$
7	INTERSECTION	h		g)	$R_1(Z) \div R_2(Y)$
8	DIFFERENCE	e		h)	$R_1 \cap R_2$
9	CARTESIAN PRODUCT	f		i)	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{(\text{<join attributes 1>}),(\text{<join attributes 2>})} R_2$ OR $R_1 * R_2$
10	DIVISION	g		j)	$R_1 \bowtie_{\text{<join condition>}} R_2$

4. Consider the following relational schema-

Reader (RDNR, Surname, Firstname, City, Birthdate)
Book (ISBN, Title, Author, NoPages, PubYear, PublisherName)
Publisher (PublisherName, PublisherCity)
Category (CategoryName, BelongsTo)
Copy (ISBN, CopyNumber, Shelf, Position)
Loan (ReaderNr, ISBN, Copy, ReturnDate)
BookCategory (ISBN, CategoryName)

Formulate the following queries in relational algebra:

i)	Which are the last names of the readers in Zurich?
	∏ Surname(σ City='Zurich' (Reader))
ii)	Which books (Author, Title) are from publishers in Zurich, Bern, or New York?
	∏ Author,Title (σ PubCity='Zurich' ∨ PubCity='Bern' ∨ PubCity='NewYork' (Book ⋈ Publisher))
iii)	Which books in the category 'Alps' do not belong to the category 'Switzerland'? Do not take into account subcategories!
	(∏ ISBN (σ CategoryName='Alps' (Book ⋈ BookCategory)))−(∏ ISBN (σ CategoryName='Switzerland' (Book ⋈ BookCategory)))
iv)	Which readers (Surname, Firstname) have borrowed books that were published in their home town?
	∏ Firstname,Surname (σ City=PubCity (Publisher ⋈ Book ⋈ Loan ⋈ Reader))
v)	Which readers (Surname, Firstname) have borrowed at least a book that has been borrowed also by the reader Lemmi Schmöker (the reader Lemmi Schmöker should not be included in the results)?
	∏ R1.Firstname, R1.Surname (ρ R1(Reader) ⋈ ρ L1(Loan)) ⋈ R1.ReaderNr<>R2.ReaderNr, L1.ISBN=L2.ISBN (ρ R2 (σ Surname='Schmoker' ∧ Surname='Lemmi' Reader)⋈ ρ L2(Loan)))

5. Consider the **Train Connections** database and given schema-

Cities (<u>Name</u>, State)

Stations (<u>Name</u>, NoPlatforms, CityName, State)

Itinerary (<u>ItNr</u>, Length, StartStation, DestinationStation)

Connections (<u>FromStation</u>, <u>ToStation</u>, <u>ItNr</u>, Departure, Arrival)

Suppose that the relation Connections already contains the transitive closure for each given train, e.g., if there is a direct train from Zurich to Geneva with a stop in Bern, then there exists a relation tuple for Zurich → Bern, Bern → Geneva, and Zurich → Geneva.

Formulate the following queries in relational algebra:

a)	Find all the direct connections from Zurich (any station) to Geneva (any station)
	(ρ FromName ← Name (∏ Name (σ CityName = Zurich (Stations)))) ⋈FromName=FromStation Connections ⋈ToName=ToStation (ρ ToName ← Name (∏ Name (σ CityName = Geneva(Stations))))
b)	Find all the single-transfer connections from Zurich to Locarno. The transfer station can be any of the stations but the connecting trains should run on the same day. (You can use a function DAY() on the attributes Departure and Arrival in order to determine the day.)
	(ρFromName←Name(∏Name(σCityName=Zurich(Stations)))) ⋈FromName=c1.FromStation ρc1(Connections) ⋈c1.ToStation=c2.FromStation ∧ c1.Arrival<c2.Departure ∧ DAY(c1.Arrival)=DAY(c2.Departure) ∧ c1.ItNr<>c2.ItNrρc2(Connections) ⋈ToName=c2.ToStation(ρToName←Name(∏Name(σCityName=Locarno(Stations))))
c)	Is it possible to find all possible connections between two stations independent on the number of transfers?
	Relational algebra does not offer infinite recursion. Therefore, we can only find connections up to a specific maximum search depth.

6. Match related algebra expression-.

Which of the following relational algebra expressions represents ① a left outer join (⟕), ② a right outer join (⟖), and ③ a full outer join (⟗).

Put the related query order in associated boxes-

Seq. No.	Relational Algebra Expression
2	$\Pi_{R \cup S}(S - \Pi_S(R \bowtie S)) \cup (R \bowtie S)$
1	$\Pi_{R \cup S}(R - \Pi_R(R \bowtie S)) \cup (R \bowtie S)$
3	$(R \bowtie S) \cup (\Pi_{R \cup S}(R - \Pi_R(R \bowtie S))) \cup (\Pi_{R \cup S}(S - \Pi_S(R \bowtie S)))$

7. Consider the following two relations:

P =

A	B
1	1
2	2
1	3
2	4
3	1
1	2

Q =

B	C	D
1	4	0
2	5	2
1	7	2
3	2	2

For each of the following expressions circle or Tick the possible outcomes(all the tuples that are not in its result set (the tuples contain all four columns: [A, B, C, D]).

A. $P \bowtie Q$:				**B.** $P ⟕ Q$:
(a) [1, 1, 7, 2]				(a) [3, 1, 7, 2]
(b) [1, 2, 5, 2]				(b) [4, 1, 4, 0] not contained
(c) [3, 2, 5, 0] not contained				(c) [2, 4, - , -] not contained
(d) [3, 1, 4, 0]				(d) [3, 1, 4, 0]
(e) [2, 4, 2, 2] not contained				(e) [1, 3, 2, 2]

8. **Olympic Game Database:**

Consider the following relational schema:

Runner (<u>Name</u>, Birthday, Country)

Run (<u>Name</u>, <u>Distance</u>, Time)

A runner can run in several runs over different race distances. Thanks to high-speed cameras, two runners cannot have the exact same time in the same run.

For every description find a matching relational algebra query. For some descriptions there is no matching query.

1	All 100m race distance runs in which only runners from Switzerland (CH) participated.	IV		I	$\Pi_{\texttt{Name,Distance,Time}}((\text{Runner} - \sigma_{\texttt{Country != 'CH'}}(\text{Runner})) \bowtie (\text{Run} \cup \sigma_{\texttt{Distance} < 100}(\text{Run})))$
2	All runs with a distance greater than 100m in which only runners from Switzerland participated.	II		II	$\Pi_{\texttt{Name,Distance,Time}}((\sigma_{\texttt{Country = 'CH'}}(\text{Runner}) - \sigma_{\texttt{Country != 'CH'}}(\text{Runner})) \bowtie \sigma_{\texttt{Distance != 100}}(\text{Run} - \sigma_{\texttt{Distance} < 100}(\text{Run})))$

3	All runs in which only runners from Switzerland participated.	I		III	$\Pi_{Name,Distance,Time}((\sigma_{Country\ !=\ 'CH'}(Runner) - \sigma_{Country\ =\ 'CH'}(Runner)) \bowtie \sigma_{Distance=100}(Run - \sigma_{Distance\ >\ 100}(Run)))$
4	All 100m race distance runs in which the runners were not from Switzerland.	III		IV	$\Pi_{Name,Distance,Time}((\sigma_{Country\ =\ 'CH'}(Runner) - \sigma_{Country\ !=\ 'CH'}(Runner)) \bowtie \sigma_{Distance=100}(Run - \sigma_{Distance\ <\ 100}(Run)))$
5	All runs in which the runners were not from Switzerland.	V		V	$\Pi_{Name,Distance,Time}(\sigma_{Country\ !=\ 'CH'}(Runner)\ ) \bowtie \sigma_{Distance>100}(Run\ - \sigma_{Distance\ <\ 100}(Run)))$

9. Consider the following relational schema-

User

Id	Name	Age	Gender	OccupationId	CityId
1	John	25	Male	1	3
2	Sara	20	Female	3	4
3	Victor	31	Male	2	5
4	Jane	27	Female	1	3

Occupation

OccupationId	OccupationName
1	Software Engineer
2	Accountant
3	Pharmacist
4	Library Assistant

City

CityId	CityName
1	Halifax
2	Calgary
3	Boston
4	New York
5	Toronto

i)	Solve the following relational expressions for above relations. (Output will be Set of rows and columns) **a.** $P_{Name}(R_{Age>25}(User))$ **b.** $R_{Id>2vAge!=31}(User)$ **c.** $R_{User.OccupationId=Occupation.OccupationId}(User\ X\ Occupation)$ **d.** $User \bowtie Occupation \bowtie City$ **e.** $P_{Name,Gender}(R_{CityName="Boston"}(User \bowtie City))$
a.	

Name
Victor
Jane

b.

id	Name	Age	Gender	OccupationId	CityId
1	John	25	Male	1	3
2	Sara	20	Female	3	4
3	Victor	31	Male	2	5
4	Jane	27	Female	1	3

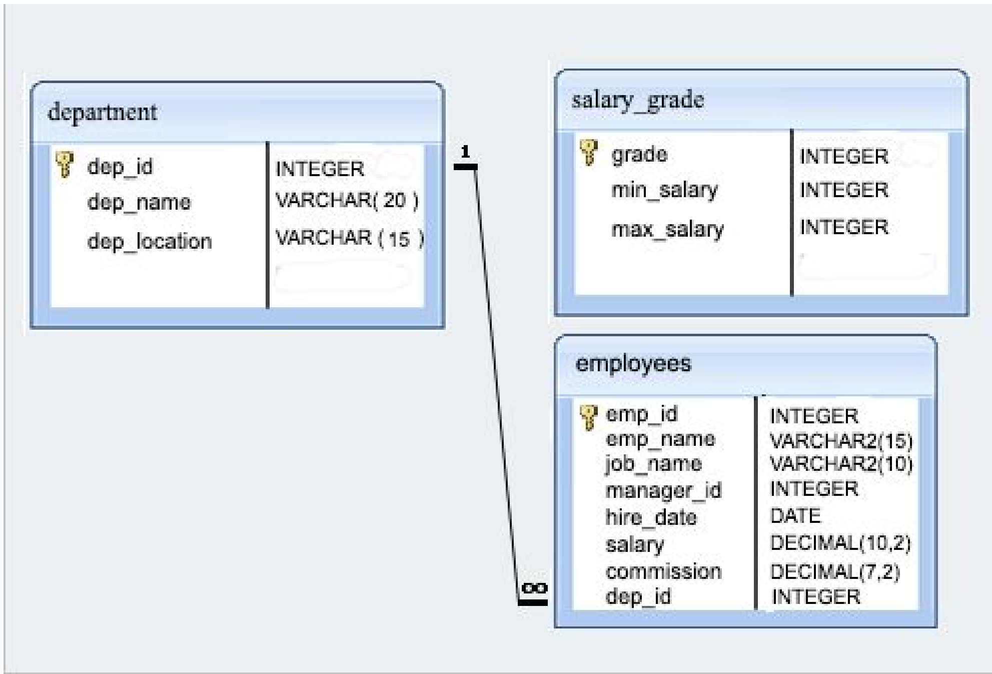
c

id	Name	Age	Gender	OccupationId	CityId	OccupationName
1	John	25	Male	1	3	Software Engineer
2	Sara	20	Female	3	4	Pharmacist
3	Victor	31	Male	2	5	Accountant
4	Jane	27	Female	1	3	Software Engineer

<table>
<tr><td></td><td></td></tr>
<tr><td>d</td><td>
<table>
<tr><td>id</td><td>Name</td><td>Age</td><td>Gender</td><td>OccupationId</td><td>CityId</td><td>OccupationName</td><td>CityName</td></tr>
<tr><td>1</td><td>John</td><td>25</td><td>Male</td><td>1</td><td>3</td><td>Software Engineer</td><td>Boston</td></tr>
<tr><td>2</td><td>Sara</td><td>20</td><td>Female</td><td>3</td><td>4</td><td>Pharmacist</td><td>New York</td></tr>
<tr><td>3</td><td>Victor</td><td>31</td><td>Male</td><td>2</td><td>5</td><td>Accountant</td><td>Toronto</td></tr>
<tr><td>4</td><td>Jane</td><td>27</td><td>Female</td><td>1</td><td>3</td><td>Software Engineer</td><td>Boston</td></tr>
</table>
</td></tr>
<tr><td>e</td><td>
<table>
<tr><td>Name</td><td>Gender</td></tr>
<tr><td>John</td><td>Male</td></tr>
<tr><td>Jane</td><td>Female</td></tr>
</table>
</td></tr>
<tr><td>ii)</td><td>Write SQL statements for relational expressions in question i).</td></tr>
<tr><td>a)</td><td>SELECT Name FROM User WHERE Age>25;</td></tr>
</table>

b)	SELECT * FROM User WHERE id>2 AND Age!=31;
c)	SELECT * FROM User NATURAL JOIN Occupation;
d)	SELECT * FROM User NATURAL JOIN Occupation NATURAL JOIN City;
e)	SELECT Name, Gender FROM User NATURAL JOIN City WHERE CityName='Boston';

10. Consider the following schema to write queries-



i)	Write a query in SQL to display all the details of managers
	SELECT * FROM employees WHERE emp_id IN (SELECT manager_id FROM employees);
ii)	Write a query in SQL to display the employee ID, name, job name, hire date, and experience of all the managers
	SELECT emp_id, emp_name, job_name, hire_date, age(CURRENT_DATE, hire_date) "Experience" FROM employees WHERE emp_id IN (SELECT manager_id FROM employees);
iii)	Write a query in SQL to list the employee ID, name, salary, department name of all

	the 'MANAGERS' and 'ANALYST' working in SYDNEY, PERTH with an exp more than 5 years without receiving the commission and display the list in ascending order of location.
	<pre>SELECT e.emp_id, e.emp_name, e.salary, d.dep_name FROM employees e, department d WHERE d.dep_location IN ('SYDNEY', 'PERTH') AND e.dep_id = d.dep_id AND e.emp_id IN (SELECT e.emp_id FROM employees e WHERE e.job_name IN ('MANAGER', 'ANALYST') AND (DATE_PART('year', CURRENT_DATE)-DATE_PART('year', hire_date))> 5 AND e.commission IS NULL) ORDER BY d.dep_location ASC;</pre>
iv)	Write a query in SQL to display the employee ID, name, salary, department name, location, department ID, job name of all the employees working at SYDNEY or working in the FINANCE department with an annual salary above 28000, but the monthly salary should not be 3000 or 2800 and who does not works as a MANAGER and whose ID containing a digit of '3' or '7' in 3rd position. List the result in ascending order of department ID and descending order of job name.
	<pre>SELECT E.emp_id, E.emp_name, E.salary, D.dep_name, D.dep_location, E.dep_id, E.job_name FROM employees E, department D WHERE (D.dep_location = 'SYDNEY' OR D.dep_name = 'FINANCE') AND E.dep_id=D.dep_id AND E.emp_id IN (SELECT emp_id FROM employees E WHERE (12*E.salary) > 28000</pre>

	`` AND E.salary NOT IN (3000, 2800) AND E.job_name !='MANAGER' AND (trim(to_char(emp_id,'99999')) LIKE '__3%' OR trim(to_char(emp_id,'99999')) LIKE '__7%')) ORDER BY E.dep_id ASC, E.job_name DESC; ```
v)	Write a query in SQL to list all the employees of grade 2 and 3.
	`` SELECT * FROM employees e, salary_grade s WHERE e.salary BETWEEN s.min_sal AND s.max_sal AND s.grade IN (2, 3); ```
vi)	Write a query in SQL to display all the employees of grade 4 and 5 who are working as ANALYST or MANAGER.
	`` SELECT * FROM employees e, salary_grade s WHERE e.salary BETWEEN s.min_sal AND s.max_sal AND s.grade IN (4, 5) AND e.emp_id IN (SELECT e.emp_id FROM employees e WHERE e.job_name IN ('MANAGER', 'ANALYST')); ```
v)	Write a query in SQL to list the details of the employees whose salary is more than the salary of JONAS.
	`` SELECT * FROM employees WHERE salary > (SELECT salary FROM employees WHERE emp_name = 'JONAS'); ```
vi)	Write a query in SQL to list the employees of department ID 2001 who works in the designation same as department ID 1001

	``SELECT * FROM employees e, department d WHERE d.dep_id = 2001 AND e.dep_id = d.dep_id AND e.job_name IN (SELECT e.job_name FROM employees e, department d WHERE e.dep_id = d.dep_id AND d.dep_id =1001);```
vii)	Write a query in SQL to list the employees whose salary is same as the salary of FRANK or SANDRINE. List the result in descending order of salary.
	``SELECT * FROM employees WHERE salary IN (SELECT salary FROM employees e WHERE (emp_name = 'FRANK' OR emp_name = 'BLAZE') AND employees.emp_id <> e.emp_id) ORDER BY salary DESC;```
viii)	Write a query in SQL to list the employees whose salary is more than the total remuneration of the SALESMAN.
	``SELECT * FROM employees WHERE salary > (SELECT max(salary+commission) FROM employees WHERE job_name = 'SALESMAN');```

11. Consider the following relations containing airline flight information:
Flights(flno:integer,from:string,to:string,distance:integer,departs:time,arrives:time)

Aircraft (aid:integer, aname:string, cruisingrange:integer)

Certified (eid:integer, aid:integer)

Employees (eid:integer, ename:string, salary:integer)

Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified for some aircraft (otherwise, he or she would not qualify as a pilot), and only pilots are certified to fly. Write the following queries in relational algebra(RA), tuple relational calculus(TRC), domain relational calculus(DRC) and Structured Query Language(SQL). Note that some of these queries may not be expressible in relational algebra (and, therefore, also not expressible in tuple and domain relational calculus). For such queries, informally explain why they cannot be expressed. (You can refer Standard airline schema).

1	Find the eids of pilots certified for some Boeing aircraft
RA	$\pi_{eid}(\sigma_{aname='Boeing'}(Aircraft \bowtie Certified))$
TRC	$\{C.eid \mid C \in Certified \wedge \exists A \in Aircraft(A.aid = C.aid \wedge A.aname = 'Boeing')\}$

DRC	$\{\langle Ceid\rangle \mid \langle Ceid, Caid\rangle \in Certified \wedge \exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft \wedge Aid = Caid \wedge AN = \text{'}Boeing'\rangle\}$
SQL	SELECT C.eid FROM Aircraft A, Certified C WHERE A.aid = C.aid AND A.aname = 'Boeing'
2	Find the names of pilots certified for some Boeing aircraft.
RA	$\pi_{ename}(\sigma_{aname=\text{'}Boeing'}(Aircraft \bowtie Certified \bowtie Employees))$

TRC	$\{E.ename \mid E \in Employees \wedge \exists C \in Certified$ $(\exists A \in Aircraft(A.aid = C.aid \wedge A.aname = \text{'}Boeing' \wedge E.eid = C.eid))\}$
DRC	$\{\langle EN\rangle \mid \langle Eid, EN, ES\rangle \in Employess\wedge$ $\exists Ceid, Caid(\langle Ceid, Caid\rangle \in Certified\wedge$ $\exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft\wedge$ $Aid = Caid \wedge AN = \text{'}Boeing' \wedge Eid = Ceid)\}$
SQL	SELECT E.ename FROM Aircraft A, Certified C, Employees E WHERE A.aid = C.aid AND A.aname = 'Boeing' AND E.eid = C.eid
3	Find the aids of all aircraft that can be used on non-stop flights from Bonn to Madras.
RA	$\rho(LAtoNY, \sigma_{from=\text{'}L.A.' \wedge to=\text{'}N.Y.'}(Flights))$ $\pi_{aid}(\sigma_{cruisingrange>distance}(Aircraft \times LAtoNY))$

TRC	$\{A.aid \mid A \in Aircraft \wedge \exists F \in Flights$ $(F.from = \text{'L.A.'} \wedge F.to = \text{'N.Y.'} \wedge A.cruisingrange > F.distance)\}$
DRC	$\{Aid \mid \langle Aid, AN, AR\rangle \in Aircraft \wedge$ $(\exists FN, FF, FT, FDi, FDe, FA(\langle FN, FF, FT, FDi, FDe, FA\rangle \in Flights \wedge$ $FF = \text{'L.A.'} \wedge FT = \text{'N.Y.'} \wedge FDi < AR))\}$
SQL	`SELECT` A.aid `FROM` Aircraft A, Flights F `WHERE` F.from = 'L.A.' `AND` F.to = 'N.Y.' `AND` A.cruisingrange > F.distance
4	Identify the flights that can be piloted by every pilot whose salary is more than$100,000.
RA	$\pi_{flno}(\sigma_{distance<cruisingrange \wedge salary>100,000}(Flights \bowtie Aircraft \bowtie$ $Certified \bowtie Employees)))$

TRC	$\exists E \in Employees(A.cruisingrange > F.distance \wedge E.salary > 100,000 \wedge$ $A.aid = C.aid \wedge E.eid = C.eid)\}$
DRC	$\{FN \mid \langle FN, FF, FT, FDi, FDe, FA\rangle \in Flights \wedge$ $\exists Ceid, Caid(\langle Ceid, Caid\rangle \in Certified \wedge$ $\exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft \wedge$ $\exists Eid, EN, ES(\langle Eid, EN, ES\rangle \in Employees$ $(AR > FDi \wedge ES > 100,000 \wedge Aid = Caid \wedge Eid = Ceid)\}$
SQL	`SELECT` E.ename `FROM` Aircraft A, Certified C, Employees E, Flights F `WHERE` A.aid = C.aid `AND` E.eid = C.eid `AND` distance < cruisingrange `AND` salary > 100,000
5	Find the names of pilots who can operate planes with a range greater than 3,000miles but are not certified on any Boeing aircraft.
RA	RA $\rho(R1, \pi_{eid}(\sigma_{cruisingrange>3000}(Aircraft \bowtie Certified)))$ $\pi_{ename}(Employees \bowtie (R1 - \pi_{eid}(\sigma_{aname='Boeing'}(Aircraft \bowtie Certified))))$

TRC	$\{E.ename \mid E \in Employees \wedge \exists C \in Certified(\exists A \in Aircraft$ $(A.aid = C.aid \wedge E.eid = C.eid \wedge A.cruisingrange > 3000))\wedge$ $\neg(\exists C2 \in Certified(\exists A2 \in Aircraft(A2.aname = \text{'}Boeing' \wedge C2.aid = A2.aid \wedge C2.eid = E.eid)))\}$
DRC	$\{\langle EN\rangle \mid \langle Eid, EN, ES\rangle \in Employess\wedge$ $\exists Ceid, Caid(\langle Ceid, Caid\rangle \in Certified\wedge$ $\exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft\wedge$ $Aid = Caid \wedge Eid = Ceid \wedge AR > 3000))\wedge$ $\neg(\exists Aid2, AN2, AR2(\langle Aid2, AN2, AR2\rangle \in Aircraft\wedge$ $\exists Ceid2, Caid2(\langle Ceid2, Caid2\rangle \in Certified$ $\wedge Aid2 = Caid2 \wedge Eid = Ceid2 \wedge AN2 = \text{'}Boeing'))))\}$
SQL	SELECT E.ename FROM Certified C, Employees E, Aircraft A WHERE A.aid = C.aid AND E.eid = C.eid AND A.cruisingrange > 3000 AND E.eid NOT IN (SELECT C2.eid FROM Certified C2, Aircraft A2 WHERE C2.aid = A2.aid AND A2.aname = 'Boeing')
6	Find the eids of employees who make the second highest salary.

RA	$\rho(E2, Employees)$ $\rho(E3, \pi_{E2.eid}(E1 \bowtie_{E1.salary > E2.salary} E2)$ $\rho(E4, E2 \bowtie E3)$ $\rho(E5, E2 \bowtie E3)$ $\rho(E6, \pi_{E5.eid}(E4 \bowtie_{E1.salary > E5.salary} E5)$ $(\pi_{eid} E3) - E6$
TRC	$\{E1.eid \mid E1 \in Employees \wedge \exists E2 \in Employees(E2.salary > E1.salary$ $\wedge \neg(\exists E3 \in Employees(E3.salary > E2.salary)))\}$
DRC	$\{\langle Eid1 \rangle \mid \langle Eid1, EN1, ES1 \rangle \in Employess \wedge$ $\exists Eid2, EN2, ES2(\langle Eid2, EN2, ES2 \rangle \in Employess(ES2 > ES1)$ $\wedge \neg(\exists Eid3, EN3, ES3(\langle Eid3, EN3, ES3 \rangle \in Employess(ES3 > ES2))))\}$
SQL	`SELECT E.eid` `FROM Employees E` `WHERE E.salary = (SELECT MAX (E2.salary)` `FROM Employees E2` `WHERE E2.salary ≠ (SELECT MAX (E3.salary)` `FROM Employees E3 ))`
7	Find the eids of employees who are certified for the largest number of aircraft

RA	This cannot be expressed in relational algebra (or calculus) because there is no operator to count, and this query requires the ability to count upto a number that depends on the data.
TRC	This cannot be expressed in relational algebra (or calculus) because there is no operator to count, and this query requires the ability to count upto a number that depends on the data.
DRC	This cannot be expressed in relational algebra (or calculus) because there is no operator to count, and this query requires the ability to count upto a number that depends on the data.
SQL	`SELECT Temp.eid FROM ( SELECT C.eid AS eid, COUNT(C.aid) AS cnt FROM Certified C GROUP BY C.eid ) AS Temp WHERE Temp.cnt = (SELECT MAX(Temp.cnt) FROM Temp);`

12.

1) Given below is the table **department_manager_employee**

department_manager_employee		
employee_id	department	manager
1101	Finance	Sam
1102	R&D	Rohit
1101	HR	Sam
1103	Retail	Rohit

Find out the functional dependencies that **do not** hold on the above table:

a) $employee\_id \rightarrow manager$

b) $employee\_id \rightarrow department$

c) $manager \rightarrow department$

d) $department \rightarrow manager$

A		B		C		D	

2) Consider the emp_project table.

emp_project			
pname	pnumber	plocation	dnum
ProductA	1	New York	5
ProductB	2	Tempe	5
ProductC	3	Wilmington	5
ERP	10	Delaware	4
Reorg	20	Wilmington	1
Newbee	30	Delaware	4

Identify the correct query for the following output.

emp_project
pname
ERP
Newbee

a) `SELECT pname FROM TABLE emp_project WHERE dnum = 4;`

b) `SELECT pname FROM emp_project FOR dnum = 4;`

c) `SELECT pname FROM TABLE emp_project FOR dnum = 4;`

d) `SELECT pname FROM emp_project WHERE dnum = 4;`

A		B		C		D	

3) Consider the `customer` table.

`customer(customer_id, dept_name, credits).`

Create a new customer `'Stacey'`, with 10 credits for department `'Loan'`.

Identify the appropriate SQL.

a)
```
INSERT INTO TABLE customer
VALUES ('Stacey','Loan', 10)
```

b)
```
INSERT INTO customer
('Stacey', 'Loan', 10)
```

c)
```
INSERT INTO customer
VALUES ('Stacey', 'Loan', 10)
```

d)
```
INSERT INTO TABLE customer
('Stacey','Loan', 10)
```

A		B		C		D	

4) Find the names of the branches whose average loan amount is less than 126678.

Consider the relation
loan(loan_number, branch_name, amount)

a)
```
SELECT AVG(amount), branch_name
FROM loan
GROUP BY branch_name AVG(amount) < 126678;
```

b)
```
SELECT branch_name
FROM loan
GROUP BY branch_name
HAVING AVG(amount) < 126678;
```

c)
```
SELECT branch_name
FROM loan
ORDER BY branch_name
HAVING AVG(amount) < 126678;
```

d)
```
SELECT branch_name
FROM loan
GROUP BY branch_name
HAVING amount < 126678;
```

A		B	GROUP BY, HAVING AVG(amount)	C		D	

5) Write a query to set the default amount for fees to 1000 for table student.

a)
```
UPDATE  student
MODIFY fees DEFAULT 1000;
```

b)
```
UPDATE student
SET fees TO DEFAULT 1000;
```

c)
```
ALTER TABLE student
SET fees TO DEFAULT 1000;
```

d)
```
ALTER TABLE student
MODIFY fees DEFAULT 1000;
```

A		B		C		D	ALTER, MODIFY

6)

Consider the table below:

employee								
fsname	lname	ssl	bdate	address	sex	salary	s_ssl	dno
Rita	Chatterjee	123	1965-01-09	731 Fondren	F	30000	333	5
Rajeev	Sanyal	333	1955-12-08	638 Voss	M	40000	888	5
Sanjay	Agrawal	999	1978-01-19	3321 Castle	M	25000	987	4
Jennifer	Agacy	987	1980-12-08	291 Voss	F	45000	888	4
Narayan	Chaudhuri	666	1980-10-04	561 Castle	M	47000	333	5
Aishwarya	Kapoor	453	1972-09-08	291 Oak	F	25000	333	5
Mahesh	Srivastava	988	1982-11-07	567 Oak	M	35000	987	4
James	Stanley	888	1962-10-01	467 Oak	M	56000	⊥	1

The data types of the columns of the table employee are given below:

- fsname, lname, bdate, address, sex: VARCHAR
- ssl, salary, s_ssl, dno: INT

How many rows of employee table will be selected for the query given below?

```
SELECT E.salary
FROM employee AS  E, employee AS S
WHERE E.s_ssl = S.ssl;
```

a) 4
b) 2
c) 7
d) 8

A		B		C	7	D	

7)
Consider the two relations below

course

course_id	title	dept_name	credits
BIO-301	Genetics	Biology	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3

prereq

course_id	prereq_id
BIO-301	BIO-101
CS-190	CS-101
CS-347	CS-101

An operation on these two relations produce the following output.

course_id	title	dept_name	credits	prereq_id
BIO-301	Genetics	Biology	4	BIO-101
CS-190	Game Design	Comp. Sci.	4	CS-101
CS-347	null	null	null	CS-101

Identify the operation.

a) `course EQUI JOIN prereq`

b) `course NATURAL RIGHT OUTER JOIN prereq`

c) `course NATURAL LEFT OUTER JOIN prereq`

d) `course NATURAL FULL OUTER JOIN prereq`

A		B	RIGHT OUTER JOIN	C		D	

8)

Consider the table below:

employee								
fsname	lname	ssl	bdate	address	sex	salary	s_ssl	dno
Rita	Chatterjee	123	1965-01-09	731 Fondren	F	30000	333	5
Rajeev	Sanyal	333	1955-12-08	638 Voss	M	40000	888	5
Sanjay	Agrawal	999	1978-01-19	3321 Castle	M	25000	987	4
Jennifer	Agacy	987	1980-12-08	291 Voss	F	45000	888	4
Narayan	Chaudhuri	666	1980-10-04	561 Castle	M	47000	333	5
Aishwarya	Kapoor	453	1972-09-08	291 Oak	F	25000	333	5
Mahesh	Srivastava	988	1982-11-07	567 Oak	M	35000	987	4
James	Stanley	888	1962-10-01	467 Oak	M	56000	⊥	1

The data types of the columns of the table `employee` are given below:

- `fsname, lname, bdate, address, sex:  VARCHAR`
- `ssl, salary, s_ssl, dno:  INT`

What will be the output of the given query?

What will be the output of the given query?

```
SELECT lname
FROM employee
WHERE fsname = 'Jennifer' OR  s_ssl = 987 AND address like '7%';
```

a) Agacy

b) Chatterjee

c) Srivastava

d) Agrawal

A		B	Chatterjee	C		D	