DBMS IAT 2 Question paper solution key

1A) Demonstrate the working of Natural join & Theta join for the given relations: Teacher (Branch-id, Tname, Exp) & Student (Sname, Branch-id).?

A1)Natural join is an SQL join operation that creates a join on the base of the common columns in the tables. To perform natural join there must be one common attribute(Column) between two tables. Natural join will retrieve from multiple relations.
Select * from Table1 natural join table2;
A theta join is a join that links tables based on a relationship other than equality between two columns. A theta join could use any operator other than the "equal" operator.

1b)Explain trigger with suitable example.
For the following relation identify possible primary key, candidate key & super key.

Subject_Number	Subject_Name	Subject_Instructor
10	DBMS	Korth
20	Algorithms	Cormen
30	Algorithms	Leiserson

A **Trigger** in Structured Query Language is a set of procedural statements which are executed automatically when there is any response to certain events on the particular table in the database. Triggers are used to protect the data integrity in the database.

Syntax of Trigger in SQL

CREATE TRIGGER Trigger_Name
[BEFORE | **AFTER**] [**Insert** | **Update** | **Delete**]
ON [Table_Name]
[**FOR** EACH ROW | **FOR** EACH **COLUMN**]
AS
Set of SQL Statement

In the trigger syntax, firstly, we have to define the name of the trigger after the CREATE TRIGGER keyword. After that, we have to define the BEFORE or AFTER keyword with anyone event.

Then, we define the name of that table on which trigger is to occur.

After the table name, we have to define the row-level or statement-level trigger.

And, at last, we have to write the SQL statements which perform actions on the occurring of event.

Example of Trigger in SQL

To understand the concept of trigger in SQL, first, we have to create the table on which trigger is to be executed.

The following query creates the **Student_Trigger** table in the SQL database:

```
CREATE TABLE Student_Trigger
(
Student_RollNo INT NOT NULL PRIMARY KEY,
Student_FirstName Varchar (100),
Student_EnglishMarks INT,
Student_PhysicsMarks INT,
Student_ChemistryMarks INT,
Student_MathsMarks INT,
Student_TotalMarks INT,
Student_Percentage  );
```

The following query shows the structure of the**Student_Trigger** table:

```
DESC Student_Trigger;
CREATE TRIGGER Student_Table_Marks
BEFORE INSERT
ON
Student_Trigger
FOR EACH ROW
SET new.Student_TotalMarks = new.Student_EnglishMarks + new.Student_Physics
Marks + new.Student_ChemistryMarks + new.Student_MathsMarks,
new.Student_Percentage = ( new.Student_TotalMarks / 400) * 100;
```

The following query inserts the record into Student_Trigger table:

```
INSERT INTO Student_Trigger (Student_RollNo, Student_FirstName, Student_Eng
lishMarks, Student_PhysicsMarks, Student_ChemistryMarks, Student_MathsMarks,
Student_TotalMarks, Student_Percentage) VALUES ( 201, Sorya, 88, 75, 69, 92, 0, 0
);
```

To check the output of the above INSERT statement, you have to type the following SELECT statement:

```
SELECT * FROM Student_Trigger;
```

2A) Consider functional dependency X→ Y. Explain all the inferences rules for the given functional dependency with proof?

1. Reflexive Rule (IR_1)

In the reflexive rule, if Y is a subset of X, then X determines Y.

1. If $X \supseteq Y$ then $X \rightarrow Y$

Example:

1. X = {a, b, c, d, e}
2. Y = {a, b, c}

2. Augmentation Rule (IR_2)

The augmentation is also called as a partial dependency. In augmentation, if X determines Y, then XZ determines YZ for any Z.

1. If $X \rightarrow Y$ then $XZ \rightarrow YZ$

Example:

1. For R(ABCD), **if** $A \rightarrow B$ then $AC \rightarrow BC$

3.Transitive Rule (IR_3)

In the transitive rule, if X determines Y and Y determine Z, then X must also determine Z.

1. If $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$

4. Union Rule (IR_4)

Union rule says, if X determines Y and X determines Z, then X must also determine Y and Z.

If $X \rightarrow Y$ and $X \rightarrow Z$ then $X \rightarrow YZ$

Proof:

1. $X \rightarrow Y$ (given)
2. $X \rightarrow Z$ (given)
3. $X \rightarrow XY$ (using IR_2 on 1 by augmentation with X. Where XX = X)
4. $XY \rightarrow YZ$ (using IR_2 on 2 by augmentation with Y)
5. $X \rightarrow YZ$ (using IR_3 on 3 and 4)

5. Decomposition Rule (IR_5)

Decomposition rule is also known as project rule. It is the reverse of union rule.

If X → YZ then X → Y and X → Z

Proof:

1. X → YZ (given)
2. YZ → Y (using IR_1 Rule)
3. X → Y (using IR_3 on 1 and 2)

6. Pseudo transitive Rule (IR_6)

In Pseudo transitive Rule, if X determines Y and YZ determines W, then XZ determines W.

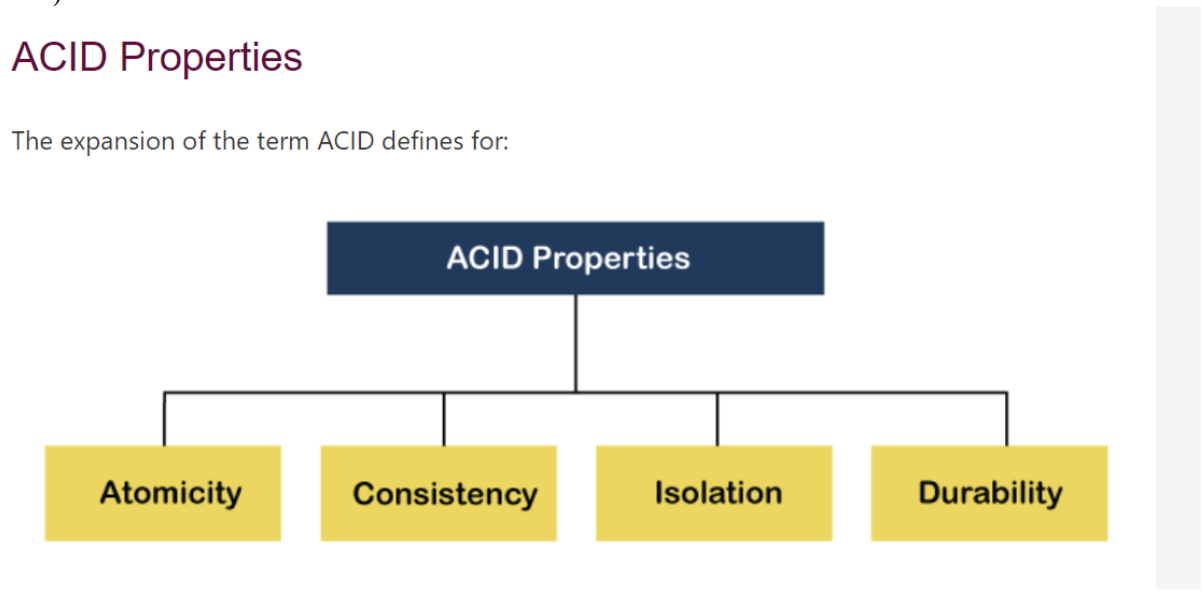
If X → Y and YZ → W then XZ → W

Proof:

X → Y (given)
WY → Z (given)
WX → WY (using IR_2 on 1 by augmenting with W)
. WX → Z (using IR_3 on 3 and 2)

2b)Explain the ACID properties of a database transaction. Consider two bank accounts X & Y. Before transaction assume balance of **X=100, Y=150**. The transaction T1 & T2 executes in sequence. **T1:** read (X), X=X-50, write(X). **T2:** read (Y), Y=Y+50, write(Y). Identify which ACID property ensures that before and after the transactions total amount of X & Y remains same. Justify your answer with suitable demonstration.

A2b)



1) Atomicity

The term atomicity defines that the data remains atomic. It means if any operation is performed on the data, either it should be performed or executed completely or should not be executed at all. It further means that the operation should not break in between or execute partially. In the case of executing operations on the transaction, the operation should be completely executed and not partially.

2) Consistency

The word **consistency** means that the value should remain preserved always. In DBMS, the integrity of the data should be maintained, which means if a change in the database is made, it should remain preserved always. In the case of transactions, the integrity of the data is very essential so that the database remains consistent before and after the transaction. The data should always be correct.

3) Isolation

The term 'isolation' means separation. In DBMS, Isolation is the property of a database where no data should affect the other one and may occur concurrently. In short, the operation on one database should begin when the operation on the first database gets complete. It means if two operations are being performed on two different databases, they may not affect the value of one another. In the case of transactions, when two or more transactions occur simultaneously, the consistency should remain maintained. Any changes that occur in any particular transaction will not be seen by other transactions until the change is not committed in the memory.

4) Durability

Durability ensures the permanency of something. In DBMS, the term durability ensures that the data after the successful execution of the operation becomes permanent in the database. The durability of the data should be so perfect that even if the system fails or leads to a crash, the database still survives. However, if gets lost, it becomes the responsibility of the recovery manager for ensuring the durability of the database. For committing the values, the COMMIT command must be used every time we make changes.Therefore, the ACID property of DBMS plays a vital role in maintaining the consistency and availability of data in the database.

Consider two bank accounts X & Y. Before transaction assume balance of **X=100, Y=150**. The transaction T1 & T2 executes in sequence. **T1:** read (X), X=X-50, write(X). **T2:** read (Y), Y=Y+50, write(Y). Identify which ACID property ensures that before and after the transactions total amount of X & Y remains same. Justify your answer with suitable demonstration.

Before:100+150=250

T1	T2	
Read(X)	Read(Y)	
X=100-50=50	Y=Y+50	150+50=200
Write(X)	Write(Y)	

After Execution:50+200=250

Transaction is in consistency state.

Q3) With suitable example, explain the following Relational Algebra operators: Select, Project, Division, Set Difference, Cartesian Product.

A3) **1)Select operation**

- The select operation selects tuples that satisfy a given predicate.
- It is denoted by sigma (σ).

Notation: σ p(r)

Where:

σ is used for selection prediction
r is used for relation
p is used as a propositional logic formula which may use connectors like: AND OR and NOT. These relational can use as relational operators like =, ≠, ≥, <, >, ≤.

2)Project Operation:

- This operation shows the list of those attributes that we wish to appear in the result. Rest of the attributes are eliminated from the table.
- It is denoted by ∏.

1. Notation: ∏ A1, A2, An (r)

Where

A1, **A2**, **A3** is used as an attribute name of relation **r**.

3.)Division operation

The division operator is used for queries which involve the 'all'.

R1 ÷ R2 = tuples of R1 associated with all tuples of R2.

4)Set Difference:

- Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in R but not in S.
- It is denoted by intersection minus (-).

Notation: R - S

5)Cartesian Product:

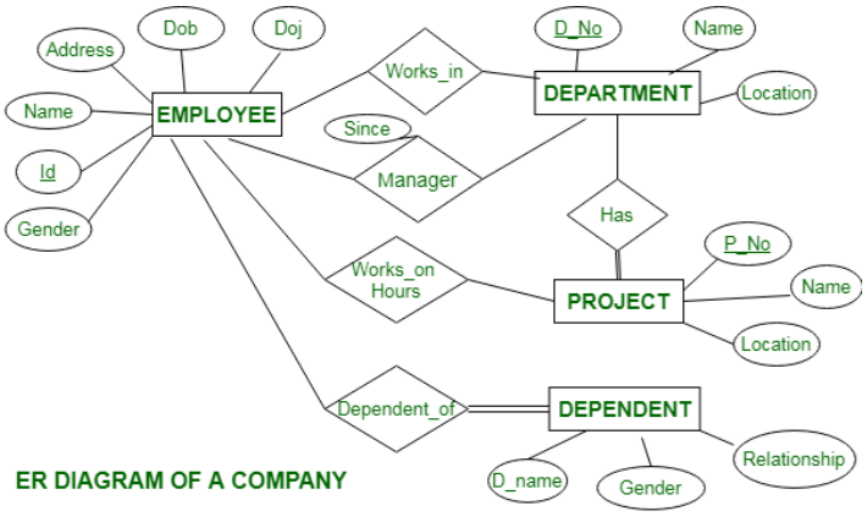
On applying CARTESIAN PRODUCT on two relations that is on two sets of tuples, it will take every tuple one by one from the left set(relation) and will pair it up with all the tuples in the right set(relation). So, the CROSS PRODUCT of two relation A(R1, R2, R3, …, Rp) with degree p, and B(S1, S2, S3, …, Sn) with degree n, is a relation C(R1, R2, R3, …, Rp, S1, S2, S3, …, Sn) with degree p + n attributes. CROSS PRODUCT is a binary set operation means, at a time we can apply the operation on two relations. But the two relations on which we are performing the operations do not have the same type of tuples, which means Union compatibility (or Type compatibility) of the two relations is not necessary. **Notation:**

$A \times S$

where A and S are the relations, the symbol '×' is used to denote the CROSS PRODUCT operator

4)Consider a Company database having entities: Employee, Department, Project, Dependent with suitable attributes and relationships between each of the entities. Draw the possible ER diagram and explain the ER to Relational mapping algorithm with step by step example.

ER Diagram of Company :



ER DIAGRAM OF A COMPANY

This Company ER diagram illustrates key information about Company, including entities such as employee, department, project and dependent. It allows to understand the relationships between entities. **Entities** and their **Attributes** are

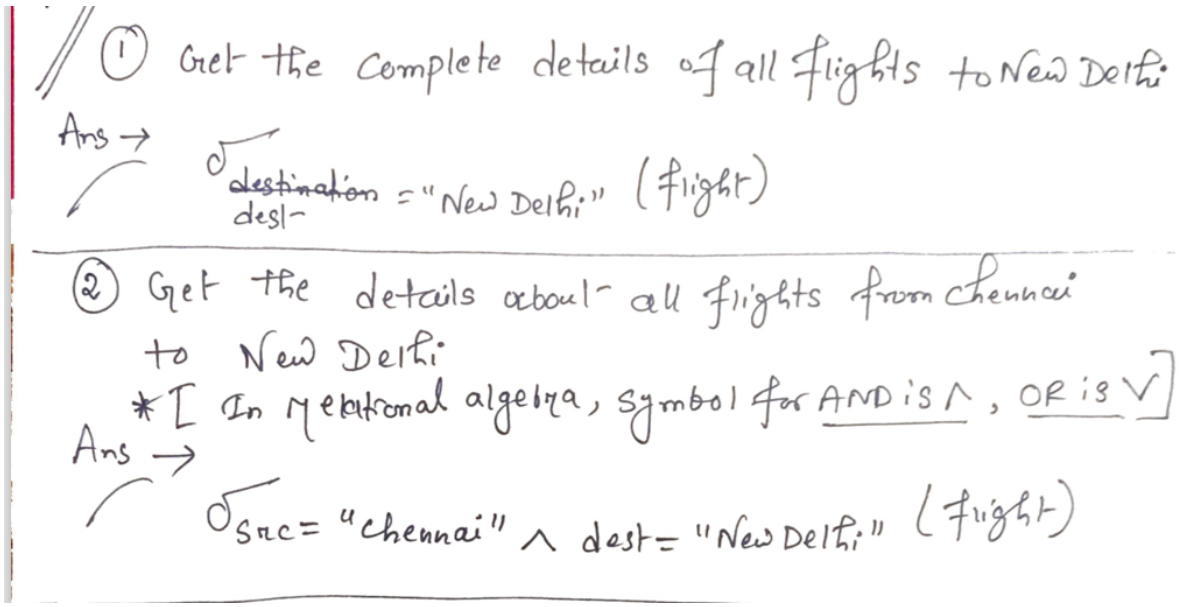
- **Employee Entity :** Attributes of Employee Entity are Name, Id, Address, Gender, Dob and Doj. Id is Primary Key for Employee Entity.
- **Department Entity :** Attributes of Department Entity are D_no, Name and Location. D_no is Primary Key for Department Entity.
- **Project Entity :** Attributes of Project Entity are P_No, Name and Location. P_No is Primary Key for Project Entity.
- **Dependent Entity :** Attributes of Dependent Entity are D_no, Gender and relationship.

Relationships are :

- **Employees works in Departments –** Many employee works in one Department but one employee can not work in many departments.

- **Manager controls a Department –** employee works under the manager of the Department and the manager records the date of joining of employee in the department.
- **Department has many Projects –** One department has many projects but one project can not come under many departments.
- **Employee works on project –** One employee works on several projects and the number of hours worked by the employee on a single project is recorded.
- **Employee has dependents –** Each Employee has dependents. Each dependent is dependent of only one employee.
- **ER-to-Relational Mapping Algorithm**
- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relation Types
- Step 4: Mapping of Binary 1:N Relationship Types.
- Step 5: Mapping of Binary M:N Relationship Types.
- Step 6: Mapping of Multivalued attributes.
- Step 7: Mapping of N-ary Relationship Types.

Q5) Consider the following relational schema:
Passenger (pid, pname, pgender, pcity)
Agency (aid, aname, acity)
Flight (fid, fdate, time, src, dest)
Booking (pid, aid, fid, date)
Given the relational algebra expression for the following:
1)Get the complete details of all flights to Mumbai
2)Get the details about all flights from Kolkata to New Delhi.
3)Find only the flight numbers for passengers with pid 456 for flights to Chennai before 10/01/2023.
4)Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.
5)Find the details of all female passengers who are associated with Indigo agency.



③ Find only the flight-numbers for passenger with pid 123 for flights to Chennai before 06/11/2020.

Ans → $\Pi_{fid}(\sigma_{pid=123}(Booking) \bowtie \sigma_{dest="chennai" \wedge fdate < 06/11/2020}(flight))$

④ Find the Passenger names for passengers who have bookings on at least one flight.

Ans → $\Pi_{name}(Passenger \bowtie booking)$

Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours.

s →

$(\sigma_{fdate=01/12/2020 \wedge time=16:00}(flight)) \cap (\sigma_{fdate=02/12/2020 \wedge time=16:00}(flight))$

) Find the details of all male passengers who are associated with jet agency.

ns → $\Pi_{passengers.pid, pname, pcity}(\sigma_{pgender="Male"}(Passengers \bowtie booking \bowtie agency))$

Please note: pgender=male write pgender='female'

Q6 a) Explain view with syntax. Consider the relation: Student (usn, name, branch, marks). Create a view to retrieve all those students details who belong to CSE branch and scored above 20 marks.

A6a)Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.

We can create View using **CREATE VIEW** statement. A View can be created from a single table or multiple tables. **Syntax**:

```
CREATE VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE condition;
```

view_name: Name for the View
table_name: Name of the table
condition: Condition to select rows

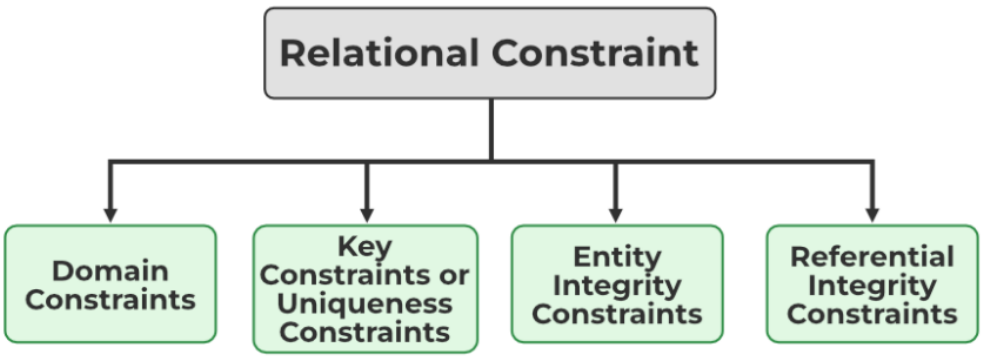
Create view Student as (Select usn,name,branch,marks from Student1 where branch='cse' and marks>=20);

Select * from Student;

Q6b)List various constraints of a relational model. Consider the relations: Employee (SSN, name, gender, salary, Dno) & Department (Dno, Dname). Explain how insertion and deletion operation will deal with all types of constraint violation for the given relations.

Mainly Constraints on the relational database are of 4 types

- Domain constraints
- Key constraints or Uniqueness Constraints
- Entity Integrity constraints
- Referential integrity constraints



Consider the relations: Employee (SSN, name, gender, salary, Dno) & Department (Dno, Dname).

First and forest SSN is primary key in Employee table

Dno is primary key in Department table and foreign key wrt Employee table(Dno) ,So if value exist in Parent table then value can be there in child table

Deletion:When Dno attribute is created with foreign key apply a on delete cascade then it will allow delete from subquent table.