**DBMS ASSIGNMENT-3**

(Amitabh Saini, TY-E 4)

* **Rules for lossless decomposition**:

There are two important properties associated with decomposition.

* **Lossless join**: This property ensures that any instance of the original relation can be identified from corresponding instances in the smaller relations.
* **Dependency preservation**: This property ensures that a constraint on the original relation can be maintained by simply enforcing some constraint on each of the smaller relations.

1. **Lossless Join Decomposition**:

* Let R be a relation schema, and let F be a set of functional dependencies on R. Let R1 and R2 form a decomposition of R if at least one of the functional dependencies is in F(plus):

R1 intersection R2 🡪 R1

R1 intersection R2 🡪 R2

* In other words, if R1 intersection R2 forms a super key of either R1 or R2, the decomposition of R is a lossless-join decomposition.
* To determine lossless decomposition it should match following conditions:

1. R1 union R2=R

2. R1 intersection R2 ≠ Φ {null}

3. R1 intersection R2 should be super key of either R1 or R2 or Both

I. R1 intersection R2🡪R1

II. R1 intersection R2🡪R2

1. **Dependency Preservation**:

Given a relation schema R and set of functional dependencies associated with it is F. R is decomposed into the relation schemas R1, R2,…,Rn with functional dependencies F1, F2,…..,Fn. Then the decomposition is dependency preserving if the closure of F’ (where F’ = F1 F2 ….. Fn) is identical to F (plus) i.e.

F’ (plus) = F (plus)

* In Banking System, consider the following lending relation:

Lending\_schema = (Branch\_name, Branch\_city, Assets, Customer\_name, Loan\_No, Amount)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Branch\_Name | Branch\_city | Assets | Customer\_name | Loan\_No | Amount |
| Pune | Shivajinagar | 90000 | ABC | L-10 | 10000 |
| Pune | Swargate | 90000 | XYZ | L-12 | 200000 |
| Navi Mumbai | Nerul | 50000 | PQR | L-14 | 500000 |
| Navi Mumbai | Vashi | 20000 | MNO | L-13 | 450000 |
| Mumbai | Chembur | 60000 | DEF | L-17 | 359000 |

The decomposition of lending relation given below is lossless-join decomposition.

It is described below:

* We now decompose the passenger relation into two relations:

1. Branch\_schema = (Branch\_name, Branch\_city, Assets)
2. Loan\_info\_schema = (Branch\_name, Customer\_name, Loan\_No, Amount)

* Branch\_schema intersection Loan\_info\_schema is Branch\_name which is common attribute and

Branch\_name 🡪 Branch\_city, Assets (By augmentation rule)

Branch\_name 🡪 Customer\_name, Loan\_No,Amount

which is nothing but a Branch\_schema relation. Thus the first clause is satisfied. Hence, the decomposition of Passenger relation into:

Branch\_schema and Loan\_info\_schema is lossless-join decomposition.

* Next, we decompose the Loan\_info\_schema into:

Loan\_schema = (Loan\_No, Branch\_name, Amount)

Borrower\_schema = (Customer\_name, Loan\_No)

* This decomposition is also a lossless-join decomposition since Address is a common attribute and Loan\_No 🡪 Amount, Branch\_name, Customer\_name .

And if we decompose the Passenger relation into two relations:

1. Branch\_schema = (Branch\_name, Branch\_city, Assets)

2. Loan\_info\_schema = (Customer\_name, Loan\_no, Amount)

This decomposition is a lossy decomposition since there is no common attribute and it is violating the following rules:

1. R1 union R2=R
2. R1 intersection R2 ≠ Φ {null}
3. R1 intersection R2 should be super key of either R1 or R2 or Both

I. R1 intersection R2🡪R1

II. R1 intersection R2🡪R2

* Now, consider the following Account relation:

Account\_info = (Account\_No, Account Type, First\_name, Last\_name, Address, Balance, Mobile\_No)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Account\_No | Account\_type | First\_name | Last\_name | Address | Balance | Mobile\_No |
| 27376 | Savings | ABC | XYZ | Nerul | 378284 | 7634871536 |
| 25329 | Savings | DEF | PQR | Vashi | 572371 | 2714687545 |
| 13124 | Current | RTG | UIT | Nerul | 612487 | 8234685757 |
| 27578 | Current | ABC | PQR | Vashi | 273582 | 2858362163 |

The decomposition of Account relation given below is lossless-join decomposition.

It is described below:

* We now decompose the passenger relation into two relations:

1. Account\_customer\_info = (Account\_No, First\_name, Last\_name, Address, Balance)

2. Account\_bal\_info = (Account\_No, Account\_type, Balance)

* Account\_customer\_info intersection Account\_bal\_info is Account\_No which is common attribute and

Account\_no 🡪 First\_name, Last\_name, Address,Balance (By augmentation rule)

Account\_no 🡪 Account\_type, Balance

which is nothing but a Account\_info relation. Thus the first clause is satisfied. Hence, the decomposition of Flight relation into:

Account\_customer\_info and Account\_bal\_info is lossless-join decomposition.

And if we decompose the Account relation into two relations:

1. Account\_customer\_info = (Account\_No, First\_name, Last\_name, Address, Balance)

2. Account\_bal\_info = (Account\_type, Balance)

This decomposition is a lossy decomposition since there is no common attribute and it is violating the following rules:

1. R1 union R2=R
2. R1 intersection R2 ≠ Φ {null}
3. R1 intersection R2 should be super key of either R1 or R2 or Both

I. R1 intersection R2🡪R1

II. R1 intersection R2🡪R2