Trigger – for managing denormalized data

empno	ename	sal	deptno	dname	
1	Rajan	345677	10	Gaming	
2	Revati	567899	20	UX design	
3	Atharva	777777	10	Gaming	

100

dept

deptno	dname	location
10	Gaming	Pune
20	UX design	pune

When any user changes the name of the department in dept table then also change the corresponding dname in emp, because the data is denormalized.

update dept

set dname='ux design'

where deptno= 20;

update emp

set dname='ux design'

where deptno= 20;

----- the following trigger will update dname in emp table as soon as dename changes in dept table

```
create trigger updateemp before update on dept
for each row
        update emp
          set dname=new.dname
          where dname= old.dname;
create trigger updatedept before update on emp
for each row
        update dept
          set dname=new.dname
          where dname= old.dname;
```

window function

row_number()	assigns unique value to each row
rank()	rank assigns same rank if the value in the column is same, but if same rank is given to multiple rows then it will skip numbers in between
dense_rank()	dense_rank assigns same rank if the value in the column is same, but if same rank is given to multiple rows then it will not skip numbers in between
first_value()	it will display the first value in the given column
last_value()	it will display the last value in the given column
lead(col,num)	it will give the nth next value in the given column
lag(col,value)	it will give the nth previous value in the given column

1. to find row_numer , rank and dense_rank departmentwise

```
select *
from (select empno,ename,sal,deptno,row_number()
   over(partition by deptno order by sal desc ) rn,
rank() over (partition by deptno  order by sal desc) rk,
 dense_rank() over (partition by deptno order by sal desc)
 drk from emp) e
where e.drk=1;
```

1. to find row_numer , rank and dense_rank for the table

```
select empno,ename,sal,deptno,row_number()
   over(order by sal desc ) rn,
rank() over (order by sal desc) rk,
 dense_rank() over (order by sal desc)
 drk from emp;
```

2. to find 2 nd previous value, next value and the first value of the sal column.

```
select empno,ename,sal,lag(sal,2) over (order by sal desc) lagsal,
lead(sal,1) over(order by sal desc) leadsal,
first_value(sal) over(order by sal desc) fv ,
last_value(sal) over(order by sal desc) lv
from emp;
```

Normalization

acno	custid	cname	mobile	balance	type	date
100	1000	Kishori	55555	33333	saving	23 march 20
101	1000	Kishori	2222	44444	current	23 march 20
102	1000	Kishori	2222	565656	demat	23 march 20
103	1001	Rajan	4444	5788	saving	23 march 21
null		Revati	66666	3456		

insertion anamoly --- In the above table a customer details cannot be inserted, without opening the account, because acno is a primary key and it cannot be empty.

deletion anamoly---- in the above table if rajn closes the account, then we will loose customer details also along with account details.

updation anamoly---- in the above table if the customer has multiple accounts and if mobile is changed at one place, and will not get reflected in other account

acno	custid	balance	type	date
100	1000	33333	saving	23 march 20
101	1000	44444	current	23 march 20
102	1000	565656	demat	23 march 20
103	1001	5788	saving	23 march 21

custid	cname	mobile
1000	Kishori	55555
1001	Rajan	4444
1002	Revati	66666

Normalization

If the data is in 3NF or BCNF form then there is no insertion , deletion and updation anomaly will be there.

1NF

1. According to the E.F. Codd, a relation will be in 1NF, if each cell of a relation contains only an atomic value. This normal form states that an attribute of a relation cannot hold multiple values. It should hold only single-valued attributes. Values stored in an attribute should be of the same domain.
2. **Example:**
3. The following **student relation** is not in 1NF because the Subject attribute contains multiple values.

Student_id	Name	Subject
101	Akash	Computer Network, JAVA
102	Vikrant	Database Management System
103	Amrita	Software Engineering, Compiler Design

101	Computer Network
101	JAVA
102	Database Management System
103	Software Engineering
103	Compiler Design

Student_id	Name
101	Akash
102	Vikrant
103	Amrita

3. 2NF

According to the E.F. Codd, a relation is in **2NF,** if it satisfies the following conditions:

- A relation must be in 1NF.
- And the candidate key in a relation should determine all non-prime attributes or no partial dependency should exist in the relation.

Prime attributes: The attributes which are used to form a candidate key are called prime attributes. any attributes in the table which should be unique and not null
Non-Prime attributes: The attributes which do not form a candidate key are called non-prime attributes.
Partial Dependency: If a non-prime attribute can be determined by the part of the candidate key in a relation, it is known as a partial dependency. Or we can say that, if L.H.S is the proper subset of a candidate key and R.H.S is the non-prime attribute, then it shows a partial dependency.

stuid	sname	cid	cname	marks
1	Ramesh	100	JAVA	88
1	Ramesh	101	c++	85
1	Ramesh	103	C#	85
2	Atharva	100	JAVA	88
2	Atharva	101	c++	98

candidate key for the table -→stuid+cid,fid

prime attribute stuid,cid

non prime attributes ---- sname,cname,marks

stuid-→sname

cid--→cname

stuid+cid---→marks

stuid	sname
1	Ramesh
2	Atharva

cid	cname
100	JAVA
101	c++
103	C#

stuid	cid	marks
1	100	88
1	101	85
1	103	85
2	100	88
2	101	98

3NF

According to the E.F. Codd, a relation is in **third normal form (3NF)** if it satisfies the following conditions:

- A relation must be in second normal form (2NF).
- And there should be no transitive functional dependency exists for non-prime attributes in a relation.

Third Normal Form is used to achieve data integrity and reduce the duplication of data.

A relation is in 3NF if and only if any one of the following conditions will satisfy for each non-trivial functional dependency **X→ Y:**

1. X is a super key or candidate key
2. And, Y is a prime attribute, i.e., Y is a part of candidate key.

Transitive Dependency: If **X → Y and Y→ Z** are two functional dependencies, **X → Z** is called as a transitive functional dependency.

prime attribute--→ non prime attribute -----→ nonprime attribute

acno	custid	cname	mobile	balance	type	date
100	1000	Kishori	2222	33333	saving	23 march 20
101	1000	Kishori	2222	44444	current	23 march 20
102	1000	Kishori	2222	565656	demat	23 march 20
103	1001	Rajan	4444	5788	saving	23 march 21
null		Revati	66666	3456		

the table is in 1NF

the table is 2NF

acid-→custid-→ cname , acid-→custid-→mobile

both the following tables are in 3 NF

acno	custid	balance	type	date
100	1000	33333	saving	23 march 20
101	1000	44444	current	23 march 20
102	1000	565656	demat	23 march 20
103	1001	5788	saving	23 march 21
null		3456		

custid	cname	mobile
1000	Kishori	2222
1001	Rajan	4444
1002	Revati	66666

BCNF (boyce codd normal form)

Boyce-Codd Normal Form (BCNF) is the advance version of the third normal form (3NF) that's why it is also known as a **3.5NF**
According to the E.F. Codd, a relation is in **Boyce-Codd normal form (3NF)** if it satisfies the following conditions:

- A relation is in 3NF.
- And, for every functional dependency, **X → Y,** L.H.S of the functional dependency (X) be the super key of the table.

Example 3:

In this example, we have a relation R with three columns: Id, Subject, and Professor. We have to find the highest normalization form, and also, if it is not in BCNF, we have to decompose it to satisfy the conditions of BCNF.

Id	Subject	Professor
101	Java	Mayank
101	C++	Kartik
102	Java	Sarthak
103	C#	Lakshay
104	Java	Mayank

Interpreting the table:

- One student can enroll in more than one subject.
    - Example: student with **Id 101** has enrolled in **Java and C++**.
- Professor is assigned to the student for a specified subject, and there is always a possibility that there can be multiple professors teaching a particular subject.

Finding the solution:

- Using Id and Subject together, we can find all unique records and also the other columns of the table. Hence, the Id and Subject together form the primary key.
- The table is in 1NF because all the values inside a column are atomic and of the same domain.
- We can't uniquely identify a record solely with the help of either the Id or the Subject name. As there is no partial dependency, the table is also in 2NF.
- There is no transitive dependency because the non-prime attribute i.e., Professor, is not deriving any other non-prime attribute column in the table. Hence, the table is also in 3NF.
- There is a point to be noted that the table is not in **BCNF (Boyce-Codd Normal Form)**.

Why is the table not in BCNF?

As we know that each professor teaches only one subject, but one subject may be taught by multiple professors. This shows that there is a dependency between the subject & the professor, and the subject is always dependent on the professor **(professor -> subject).** As we know that the professor column is a non-prime attribute, while the subject is a prime attribute. This is not allowed in BCNF in DBMS. **For BCNF, the deriving attribute (professor here) must be a prime attribute.**

How to satisfy BCNF?

In Example 3, we will decompose the table into two tables: the Student table and the Professor table to satisfy the conditions of BCNF.

Student Table

P_Id	S_Id	Professor
1	101	Mayank
2	101	Kartik
3	102	Sarthak
4	103	Lakshay
5	104	Mayank

Professor Table

Professor	Subject
Mayank	Java
Kartik	C++
Sarthak	Java
Lakshay	C#

Professor is now the primary key and the prime attribute column, deriving the subject column. **Hence, it is in BCNF.**

Proj Code	Proj Type	Proj Desc	Empno	Ename	Grade	Sal scale	Proj Join Date	Alloc Time
001	APP	LNG	46	JONES	A1	5	12/1/1998	24
001	APP	LNG	92	SMITH	A2	4	2/1/1999	24
001	APP	LNG	96	BLACK	B1	9	2/1/1999	18
004	MAI	SHO	72	JACK	A2	4	2/4/1999	6
004	MAI	SHO	92	SMITH	A2	4	5/5/1999	6

- Orderno
- Orderdate
- Itemno
- Qty
- Price
- Cname
- Custno
- Email
- Orderamt
- Salespersonno
- Salespersonname
- Locationid ----------location from where item dispatched
- Location name

One customer can place many order
One order contains many items
One order will be managed by one salesperson
One order belong to one customer
One order can be dispatched from different location