Boyce-Codd Normal Form (BCNF)

Last Updated : 25 Jul, 2025

Boyce-Codd Normal Form (BCNF) is an advanced version of 3NF used to reduce redundancy in databases. It ensures that for every functional dependency, the left side must be a superkey. This helps create cleaner and more consistent database designs, especially when there are multiple candidate keys.

Rules for BCNF

- **Rule 1:** The table should be in the 3rd Normal Form.
- **Rule 2:** X should be a super-key for every functional dependency (FD) X−>Y in a given relation.

> ***Note:*** *To test whether a relation is in BCNF, we identify all the determinants and make sure that they are candidate keys.*

Key Notes:

1. To verify BCNF, identify all determinants (left side of FDs) and check whether each is a candidate key.

2. If a relation is in BCNF, it is automatically in 3NF, 2NF, and 1NF as well.

The normal forms become stricter as we move from 1NF → 2NF → 3NF → BCNF:

- 1NF: Each field must hold atomic (indivisible) values.
- 2NF: No partial dependency on a primary key.
- 3NF: No transitive dependency on a primary key.
- BCNF: Every determinant must be a candidate key.

This progression ensures better structure and removes redundancy at each level.

Why Do We Need BCNF?

- 2NF and 3NF may allow anomalies if a functional dependency exists where the determinant is not a superkey.
- BCNF handles edge cases where 3NF fails to remove all redundancy, especially in tables with multiple candidate keys.

- Prevents update, insert, and delete anomalies by ensuring every determinant is a superkey.
- Makes database design more robust and easier to maintain over time.
- Improves data consistency and clarity by removing hidden or indirect dependencies.

We are going to discuss some basic examples which let you understand the properties of BCNF. We will discuss multiple examples here.

Example 1

Consider a relation R with attributes (student, teacher, subject).

R

Student	Teacher	Subject
Shayan	R.Das	Database
Shayan	K.Raman	C
Tahira	R.Das	Database
Tahira	N.Gupta	C

FD: { (student, Teacher) -> subject, (student, subject) -> Teacher, (Teacher) -> subject}

- Candidate keys are (student, teacher) and (student, subject).
- The above relation is in 3NF (since there is no transitive dependency). A relation R is in BCNF if for every non-trivial FD X->Y, X must be a key.
- The above relation is not in BCNF, because in the FD (teacher->subject), teacher is not a key. This relation suffers with anomalies −
- For example, if we delete the student Tahira , we will also lose the information that N.Gupta teaches C. This issue occurs because the teacher is a determinant but not a candidate key.

R1

Teacher	Subject
R.Das	Database
K.Raman	C
N.Gupta	C

R2

Student	Teacher
Shayan	R.Das
Shayan	K.Raman
Tahira	R.Das
Tahira	N.Gupta

R is divided into two relations R1(Teacher, Subject) and R2(Student, Teacher).

For more, refer to BCNF in DBMS.

How to Satisfy BCNF?

For satisfying this table in BCNF, we have to decompose it into further tables. Here is the full procedure through which we transform this table into BCNF. Let us first divide this main table into two tables **Stu_Branch** and **Stu_Course** Table.

Stu_Branch Table

Stu_ID	Stu_Branch
101	Computer Science & Engineering
102	Electronics & Communication Engineering

Candidate Key for this table: **Stu_ID**.

Stu_Course Table

Stu_Course	Branch_Number	Stu_Course_No
DBMS	B_001	201
Computer Networks	B_001	202

Stu_Course	Branch_Number	Stu_Course_No
VLSI Technology	B_003	401
Mobile Communication	B_003	402

Candidate Key for this table: **Stu_Course**.

Stu_Enroll Table

Stu_ID	Stu_Course_No
101	201
101	202
102	401
102	402

Candidate Key for this table: **{Stu_ID, Stu_Course_No}.**

After decomposing into further tables, now it is in BCNF, as it is passing the condition of Super Key, that in functional dependency X−>Y, X is a Super Key.

Example 3

Find the highest normal form of a relation R(A, B, C, D, E) with FD set as:

{ BC->D, AC->BE, B->E }

Explanation:

- **Step-1:** As we can see, (AC)+ ={A, C, B, E, D} but none of its subsets can determine all attributes of the relation, So AC will be the candidate key. A or C can’t be derived from any other attribute of the relation, so there will be only 1 candidate key {AC}.

- **Step-2:** Prime attributes are those attributes that are part of candidate key {A, C} in this example and others will be non-prime {B, D, E} in this example.
- **Step-3:** The relation R is in 1st normal form as a relational DBMS does not allow multi-valued or composite attributes.

The relation is in 2nd normal form because BC->D is in 2nd normal form (BC is not a proper subset of candidate key AC) and AC->BE is in 2nd normal form (AC is candidate key) and B->E is in 2nd normal form (B is not a proper subset of candidate key AC).

The relation is **not** in 3rd normal form because in BC->D (neither BC is a super key nor D is a prime attribute) and in B->E (neither B is a super key nor E is a prime attribute) but to satisfy 3rd normal for, either LHS of an FD should be super key or RHS should be a prime attribute. So the highest normal form of relation will be the 2nd Normal form.

Note: A prime attribute cannot be transitively dependent on a key in BCNF relation.

Consider these functional dependencies of some relation R

AB ->C
C ->B
AB ->B

From the given functional dependencies, the candidate keys of relation R are AB and AC. On close observation, we see that B depends transitively on AB through C, making it a transitive dependency.

- The first and third dependencies are in BCNF as their left sides are candidate keys.
- The second dependency is not in BCNF, but it's in 3NF since the right side is a prime attribute.

So, the highest normal form of relation R is 3NF.

Example 3

For example consider relation R(A, B, C)

A -> BC,
B -> A

A and B both are super keys so the above relation is in BCNF.

Note: BCNF decomposition may always not be possible with dependency preserving, however, it always satisfies the lossless join condition. For example, relation R (V, W, X, Y, Z), with functional dependencies:

V, W -> X
Y, Z -> X
W -> Y

It would not satisfy dependency preserving BCNF decomposition.

Note: Redundancies are sometimes still present in a BCNF relation as it is not always possible to eliminate them completely.

There are also some higher-order normal forms, like the 4th Normal Form and the 5th Normal Form.