Boyce–Codd normal form

Boyce - Codd normal form (or **BCNF** or **3.5NF**) is a normal form used in database normalization. It is a slightly stronger version of the third normal form (3NF). BCNF was developed in 1974 by Raymond F. Boyce and Edgar F. Codd to address certain types of anomalies not dealt with by 3NF as originally defined.[1]

If a relational schema is in BCNF then all redundancy based on functional dependency has been removed,[2] although other types of redundancy may still exist. A relational schema R is in Boyce–Codd normal form if and only if for every one of its dependencies $X \rightarrow Y$, at least one of the following conditions hold:[3]

- $X \rightarrow Y$ is a trivial functional dependency ($Y \subseteq X$),
- X is a superkey for schema R.[3]

Note that if a relational schema is in BCNF, then it is in 3NF.

3NF table always meeting BCNF (Boyce–Codd normal form)

Only in rare cases does a 3NF table not meet the requirements of BCNF. A 3NF table that does not have multiple overlapping candidate keys is guaranteed to be in BCNF.[4] Depending on what its functional dependencies are, a 3NF table with two or more overlapping candidate keys may or may not be in BCNF.

An example of a 3NF table that does not meet BCNF is:

Today's court bookings

Court	Start time	End time	Rate type
1	09:30	10:30	SAVER
1	11:00	12:00	SAVER
1	14:00	15:30	STANDARD
2	10:00	11:30	PREMIUM-B
2	11:30	13:30	PREMIUM-B
2	15:00	16:30	PREMIUM-A

- Each row in the table represents a court booking at a tennis club. That club has one hard court (Court 1) and one grass court (Court 2)
- A booking is defined by its Court and the period for which the Court is reserved
- Additionally, each booking has a Rate Type associated with it. There are four distinct rate types:
  - SAVER, for Court 1 bookings made by members
  - STANDARD, for Court 1 bookings made by non-members

- PREMIUM-A, for Court 2 bookings made by members
- PREMIUM-B, for Court 2 bookings made by non-members

The table's superkeys are:

- S_1 = {Court, Start time}
- S_2 = {Court, End time}
- S_3 = {Rate type, Start time}
- S_4 = {Rate type, End time}
- S_5 = {Court, Start time, End time}
- S_6 = {Rate type, Start time, End time}
- S_7 = {Court, Rate type, Start time}
- S_8 = {Court, Rate type, End time}
- S_T = {Court, Rate type, Start time, End time}, the trivial superkey

Note that even though in the above table *Start time* and *End time* attributes have no duplicate values for each of them, we still have to admit that in some other days two different bookings on court 1 and court 2 could *start at the same time* or *end at the same time*. This is the reason why {Start time} and {End time} cannot be considered as the table's superkeys.

However, only S_1, S_2, S_3 and S_4 are candidate keys (that is, minimal superkeys for that relation) because e.g. $S_1 \subset S_5$, so S_5 cannot be a candidate key.

Recall that 2NF prohibits partial functional dependencies of non-prime attributes (i.e., an attribute that does not occur in *any* candidate key. See "candidate keys") and that 3NF prohibits transitive functional dependencies of non-prime attributes on candidate keys.

In **Today's court bookings** table, there are no non-prime attributes: that is, all attributes belong to some candidate key. Therefore the table adheres to both 2NF and 3NF.

The table does not adhere to BCNF. This is because of the dependency Rate type → Court in which the determining attribute Rate type – on which Court depends – (1) is neither a candidate key nor a superset of a candidate key and (2) Court is no subset of Rate type.

Dependency Rate type → Court is respected, since a Rate type should only ever apply to a single Court.

The design can be amended so that it meets BCNF:

Rate types

Rate type	Court	Member flag
SAVER	1	Yes
STANDARD	1	No
PREMIUM-A	2	Yes
PREMIUM-B	2	No

Today's bookings

Court	Start time	End time	Member flag
1	09:30	10:30	Yes
1	11:00	12:00	Yes
1	14:00	15:30	No
2	10:00	11:30	No
2	11:30	13:30	No
2	15:00	16:30	Yes

The candidate keys for the Rate types table are {Rate type} and {Court, Member flag}; the candidate keys for the Today's bookings table are {Court, Start time} and {Court, End time}. Both tables are in BCNF. When {Rate type} is a key in the Rate types table, having one Rate type associated with two different Courts is impossible, so by using {Rate type} as a key in the Rate types table, the anomaly affecting the original table has been eliminated.

Achievability of BCNF

In some cases, a non-BCNF table cannot be decomposed into tables that satisfy BCNF and preserve the dependencies that held in the original table. Beeri and Bernstein showed in 1979 that, for example, a set of functional dependencies {AB → C, C → B} cannot be represented by a BCNF schema.[5]

Consider the following non-BCNF table whose functional dependencies follow the {AB → C, C → B} pattern:

Nearest shops

Person	Shop type	Nearest shop
Davidson	Optician	Eagle Eye
Davidson	Hairdresser	Snippets
Wright	Bookshop	Merlin Books
Fuller	Bakery	Doughy's
Fuller	Hairdresser	Sweeney Todd's
Fuller	Optician	Eagle Eye

For each Person / Shop type combination, the table tells us which shop of this type is geographically nearest to the person's home. We assume for simplicity that a single shop cannot be of more than one type.

The candidate keys of the table are:

- {Person, Shop type},
- {Person, Nearest shop}.

Because all three attributes are prime attributes (i.e. belong to candidate keys), the table is in 3NF. The table is not in BCNF, however, as the Shop type attribute is functionally dependent on a non-superkey: Nearest shop.

The violation of BCNF means that the table is subject to anomalies. For example, Eagle Eye might have its Shop type changed to "Optometrist" on its "Fuller" record while retaining the Shop type "Optician" on its "Davidson" record. This would imply contradictory answers to the question: "What is Eagle Eye's Shop Type?" Holding each shop's Shop type only once would seem preferable, as doing so would prevent such anomalies from occurring:

Shop near person

Person	Shop
Davidson	Eagle Eye
Davidson	Snippets
Wright	Merlin Books
Fuller	Doughy's
Fuller	Sweeney Todd's
Fuller	Eagle Eye

Shop

Shop	Shop type
Eagle Eye	Optician
Snippets	Hairdresser
Merlin Books	Bookshop
Doughy's	Bakery
Sweeney Todd's	Hairdresser

In this revised design, the "Shop near person" table has a candidate key of {Person, Shop}, and the "Shop" table has a candidate key of {Shop}. Unfortunately, although this design adheres to BCNF, it is unacceptable on different grounds: it allows us to record multiple shops of the same type against the same person. In other words, its candidate keys do not guarantee that the functional dependency {Person, Shop type} → {Shop} will be respected.

A design that eliminates all of these anomalies (but does not conform to BCNF) is possible. This design introduces a new normal form, known as Elementary Key Normal Form.[6] This design consists of the original "Nearest shops" table supplemented by the "Shop" table described above. The table structure generated by Bernstein's schema generation algorithm[7] is actually EKNF, although that enhancement to 3NF had not been recognized at the time the algorithm was designed:

Nearest shops

Person	Shop type	Nearest shop
Davidson	Optician	Eagle Eye
Davidson	Hairdresser	Snippets
Wright	Bookshop	Merlin Books
Fuller	Bakery	Doughy's
Fuller	Hairdresser	Sweeney Todd's
Fuller	Optician	Eagle Eye

Shop

Shop	Shop type
Eagle Eye	Optician
Snippets	Hairdresser
Merlin Books	Bookshop
Doughy's	Bakery
Sweeney Todd's	Hairdresser

If a referential integrity constraint is defined to the effect that {Shop type, Nearest shop} from the first table must refer to a {Shop type, Shop} from the second table, then the data anomalies described previously are prevented.

Intractability

It is NP-complete, given a database schema in third normal form, to determine whether it violates Boyce–Codd normal form.[8]

Decomposition into BCNF

If a relation R is not in BCNF due to a functional dependency $X \rightarrow Y$, decompose R into BCNF by replacing that relation with two sub-relations:

1. One with the attributes X^+,
2. and another with the attributes $R\text{-}X^+ + X$. Note that R represents all the attributes in the original relation.

Check whether both sub-relations are in BCNF and repeat the process recursively with any sub-relation which is not in BCNF.[9]

History

Chris Date has pointed out that a definition of what we now know as BCNF appeared in a paper by Ian Heath in 1971.[10] Date writes:[11]

> Since that definition predated Boyce and Codd's own definition by some three years, it seems to me that BCNF ought by rights to be called *Heath* normal form. But it isn't.

Edgar F. Codd released his original article "A Relational Model of Data for Large Shared Databanks" in June 1970. This was the first time the notion of a relational database was published. All work after this, including the Boyce–Codd normal form method was based on this relational model.

References

1. Codd, E. F. "Recent Investigations into Relational Data Base" in *Proc. 1974 Congress* (Stockholm, Sweden, 1974). New York, N.Y.: North-Holland (1974).
2. Köhler, Henning; Link, Sebastian (2018-07-01). "SQL schema design: foundations, normal forms, and normalization" (https://linkinghub.elsevier.com/retrieve/pii/S0306437917305069). *Information Systems*. **76**: 88–113. doi:10.1016/j.is.2018.04.001 (https://doi.org/10.1016%2Fj.is.2018.04.001).
3. Silberschatz, Abraham (2006). *Database System Concepts* (https://archive.org/details/databasesystemco00silb_973) (6th ed.). McGraw-Hill. pp. 333 (https://archive.org/details/databasesystemco00silb_973/page/n359). ISBN 978-0-07-352332-3.
4. Vincent, M. W. and B. Srinivasan. "A Note on Relational Schemes Which Are in 3NF But Not in BCNF". *Information Processing Letters* 48(6), 1993, pp. 281–283.

5. Beeri, Catriel and Bernstein, Philip A. "Computational problems related to the design of normal form relational schemas". *ACM Transactions on Database Systems* 4(1), March 1979, p. 50.
6. Zaniolo, Carlo. "A New Normal Form for the Design of Relational Database Schemata". *ACM Transactions on Database Systems* 7(3), September 1982, p. 493.
7. Bernstein, P. A. "Synthesizing Third Normal Form relations from functional dependencies". *ACM Transactions on Database Systems* 1(4), December 1976 pp. 277–298.
8. Beeri, Catriel; Bernstein, Philip A. (1979). "Computational Problems Related to the Design of Normal Form Relational Schemas". *ACM Transactions on Database Systems*. **4**: 30–59. doi:10.1145/320064.320066 (https://doi.org/10.1145%2F320064.320066). S2CID 11409132 (https://api.semanticscholar.org/CorpusID:11409132), Corollary 3.
9. *BCNF Decomposition* (https://www.youtube.com/watch?v=WKJH3V7UAgg) , retrieved 2023-03-15
10. Heath, I. "Unacceptable File Operations in a Relational Database". *Proc. 1971 ACM SIGFIDET Workshop on Data Description, Access, and Control*, San Diego, Calif. (November 11th–12th, 1971).
11. Date, C. J. *Database in Depth: Relational Theory for Practitioners*. O'Reilly (2005), p. 142.

Bibliography

- Date, C. J. (1999). *An Introduction to Database Systems* (8th ed.). Addison-Wesley Longman. ISBN 0-321-19784-4.

External links

- Rules Of Data Normalization (https://web.archive.org/web/20080805014412/http://www.datamodel.org/NormalizationRules.html)
- Advanced Normalization (https://web.archive.org/web/20080423014733/http://www.utexas.edu/its/archive/windows/database/datamodeling/rm/rm8.html) by ITS, University of Texas.