
DATABASE

CSE3103 : Database

Nazmus Sakib
Assistant Professor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Boyce Codd Normal Form (BCNF)

- A relation is in BCNF if it is satisfied the following conditions.
  - **It should be in the Third Normal Form.**
  - **For any dependency A → B, Determinant A should be a candidate key (a candidate key is always a super key).**
  - **Candidate key may be in the form of composite and overlapping.**
  - **Primary key is generated from the set of a candidate keys.**

Boyce Codd Normal Form (BCNF)

Enrollment

student_id	subject	Professor
12101	Java	Prof. S
12101	C++	Prof. A
12102	Java	Prof. W
12103	C#	Prof. T
12104	Java	Prof. S

- One student can enroll for multiple subjects.
- For each subject, a professor is assigned to the student.
- Multiple professors teaching one subject like we have for Java.

Who will be the **Primary Key** in this table ?

- **Single or composite ?**
- {student_id, subject} together form the primary key. subject is a candidate key.
- Because using student_id and subject, we can find all the columns of the table.

Boyce Codd Normal Form (BCNF)

Enrollment

student_id	subject	Professor
12101	Java	Prof. S
12101	C++	Prof. A
12102	Java	Prof. W
12103	C#	Prof. T
12104	Java	Prof. S

•Check Normality

- This table satisfies the **1st Normal form** because all the values are atomic, column names are unique and all the values stored in a particular column are of same domain.
- This table also satisfies the **2nd Normal Form** as their is no **Partial Dependency**.
- There is no **Transitive Dependency**, hence the table also satisfies the **3rd Normal Form**.
- But this table is not in **Boyce-Codd Normal Form**.

- There is a dependency between subject and professor here, where subject depends on the professor name.
- Dependency professor → subject
- student_id, subject form primary key, subject is also a candidate key.

Boyce Codd Normal Form (BCNF)

To make this relation(table) satisfy BCNF, we will decompose this table into two tables, **student** table and **professor** table. Below we have the structure for both the tables.

Student

student_id	p_id
12101	1
12101	2
12102	3
12103	4
12104	1

Professor

p_id	professor	subject
1	Prof. S	Java
2	Prof. A	C++
3	Prof. W	Java
4	Prof. T	C#

Boyce Codd Normal Form (BCNF)

Another Example for you on the different scenario.

Client Interview

client_id	Interview_date	Interview_time	Staff_no	Room_no
C76	11-9-19	10:30	S7	G101
C54	11-9-19	11:20	S7	G101
C74	11-9-19	12:30	S9	G102
c54	11-9-19	12:30	S7	G102

Primary Key: client_id , interview_date

Candidate key : {staff_no, interview_date, interview_time} , {room_no, interview_date, interview_time}

Boyce Codd Normal Form (BCNF)

Another Example for you on the different scenario.

Functional Dependency:

1. Client_id, interview date → interview_time, staff_no, room_no
2. staff_no, interview_date, interview_time → Client_id
3. room_no, interview_date, interview_time → staff_no, client_id
4. staff_no, interview_date → room_no
5. staff_no, room_no → interview_time
6. So on ….

Primary Key: client_id , interview_date
Candidate key : {staff_no, interview_date, interview_time} , {room_no, interview_date, interview_time}

Boyce Codd Normal Form (BCNF)

Solution on different scenario.

Client Interview date

Interview_date	Staff_no	Room_no
11-9-19	S7	G101
11-9-19	S7	G101
11-9-19	S9	G102
11-9-19	S7	G102

Primary Key: interview_date

Boyce Codd Normal Form (BCNF)

Solution on the different scenario.

Client Interview detail

client_id	Interview_date	Interview_time	Staff_no
C76	11-9-19	10:30	S7
C54	11-9-19	11:20	S7
C74	11-9-19	12:30	S9
C54	11-9-19	10:30	S7

Primary Key: client_id , interview_date, interview_time


DBMS
data
database
distributed
parallel
shared
system
systems
architecture
DBMSs
Distributed
network
processing
performance
processors
access
centralized
multiple
memory
DDBMS
nothing
sites
single
processor
Parallel
provide
local
may
typically
fragments
require
physically
site
users
applications
computers
technology
computer
disk
stored
number
sometimes
one
large
decentralized
logically
bottleneck
interconnection
Figure
application
management
transparent
organizational
Shared
nodes
definition
development
logical
distinction
control
scalability
approach
improve
Database
like
within
machines
across
major
fact
eliminates
reliability
distribution
communication
also
many
replicated
make
discuss
often
Management
makes
per
user
dis
administered
mode
consists
way
global
associated
partitioned
used
least
must
System
capable
databases
computing
split
order
scalable
architectures
referred
operational
directly
com
disks
islands
resources
remote
defined
controlled
based
Date
structure
incompatible
collection
provides
independently
share