Normalization

Normalization is a database design technique used to organize and structure relational databases efficiently. It involves breaking down a single large table into multiple smaller tables, reducing redundancy and improving data integrity. The process ensures that each piece of data is stored in the database only once, reducing the possibility of data inconsistencies and anomalies.There are several levels of normalization, each designed to eliminate specific types of data anomalies. The most commonly used normalization forms are:

1. First Normal Form (1NF)

Each table cell contains a single value (atomic).Eliminates repeating groups by putting each value in its own row.
Example:

```
Original Table (Not in 1NF):
-------------------------------------
| Employee ID |  Name      | Skills   |
-------------------------------------
| 101         | John Doe   | Java     |
|             |            | Python   |
-------------------------------------

Converted to 1NF:
-----------------------------
| Employee ID |  Name      |
-----------------------------
| 101         | John Doe   |
-----------------------------


-----------------------------
| Employee ID | Skills     |
-----------------------------
| 101         | Java       |
-----------------------------
| 101         | Python     |
-----------------------------
```

2. Second Normal Form (2NF)

Meets 1NF requirements.
Has a primary key to uniquely identify each row in a table.
Eliminates partial dependencies by creating separate tables for independent sets of data.
Example:

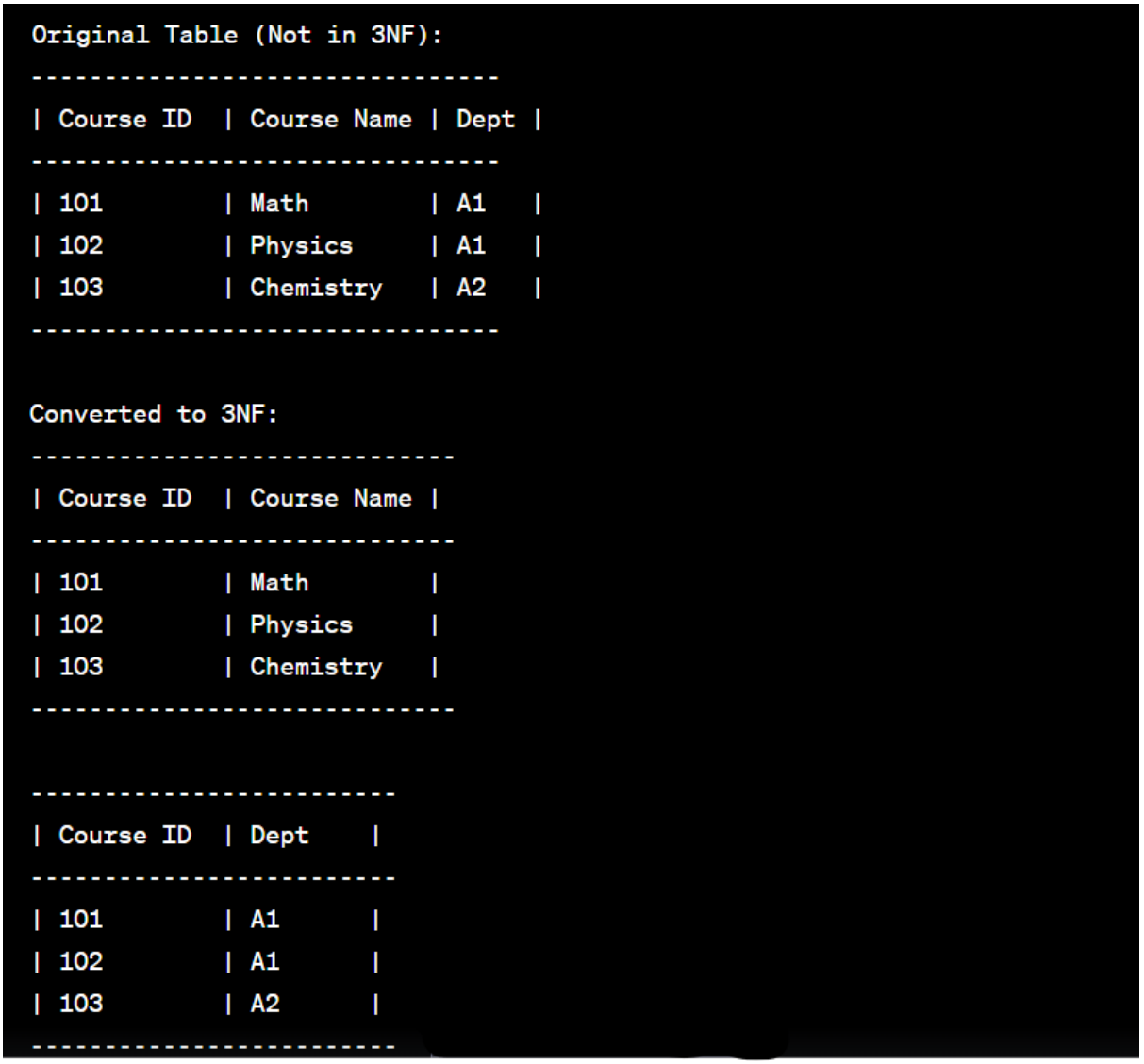
```
Original Table (Not in 2NF):
-----------------------------
| Order ID   | Product    |
-----------------------------
| 1          | Laptop     |
| 1          | Mouse      |
| 2          | Keyboard   |
-----------------------------

Converted to 2NF:
-----------------------------
| Order ID   |
-----------------------------
| 1          |
| 2          |
-----------------------------


-----------------------------
| Order ID   | Product    |
-----------------------------
| 1          | Laptop     |
| 1          | Mouse      |
| 2          | Keyboard   |
-----------------------------
```

3. Third Normal Form (3NF)

Meets 2NF requirements.
Eliminates transitive dependencies by moving non-key attributes to separate tables.
Example:



4. <u>Boyce-Codd Normal Form (BCNF)</u>

BCNF is a higher level of normalization than Third Normal Form (3NF) and is used to eliminate all non-trivial functional dependencies within a table. A non-trivial functional dependency occurs when a non-prime attribute (attribute not part of the primary key) depends on part of the primary key. BCNF ensures that there are no partial dependencies, meaning each non-prime attribute is fully dependent on the entire primary key.

Example:

Consider the following table with the functional dependencies: StudentID → CourseCode and CourseCode → CourseName.

```
Table: StudentCourses
--------------------------------------
| StudentID  | CourseCode  | CourseName |
--------------------------------------
| 101        | CS101       | C  uter Science I   |
| 101        | MATH101     | Calculus I          |
| 102        | MATH101     | Calculus I          |
| 103        | CS101       | Computer Science I  |
--------------------------------------
```

In this table, both StudentID and CourseCode are part of the primary key. However, the CourseName attribute depends only on CourseCode and not on the entire primary key (StudentID + CourseCode). This table is not in BCNF because there is a non-trivial functional dependency (CourseCode → CourseName) that violates the BCNF requirement.To bring this table into BCNF, we split it into two separate tables:

```
| StudentID  | Name  |
----------------------
| 101        | John  |
| 102        | Jane  |
| 103        | Jack  |
----------------------

Table: Courses
--------------------------------------
| CourseCode  | CourseName          |
--------------------------------------
| CS101       | Computer Science I  |
| MATH101     | Calculus I          |
--------------------------------------

Table: StudentCourses
----------------------
| StudentID  | CourseCode  |
----------------------
| 101        | CS101       |
| 101        | MATH101     |
| 102        | MATH101     |
```

5. Fourth Normal Form (4NF)

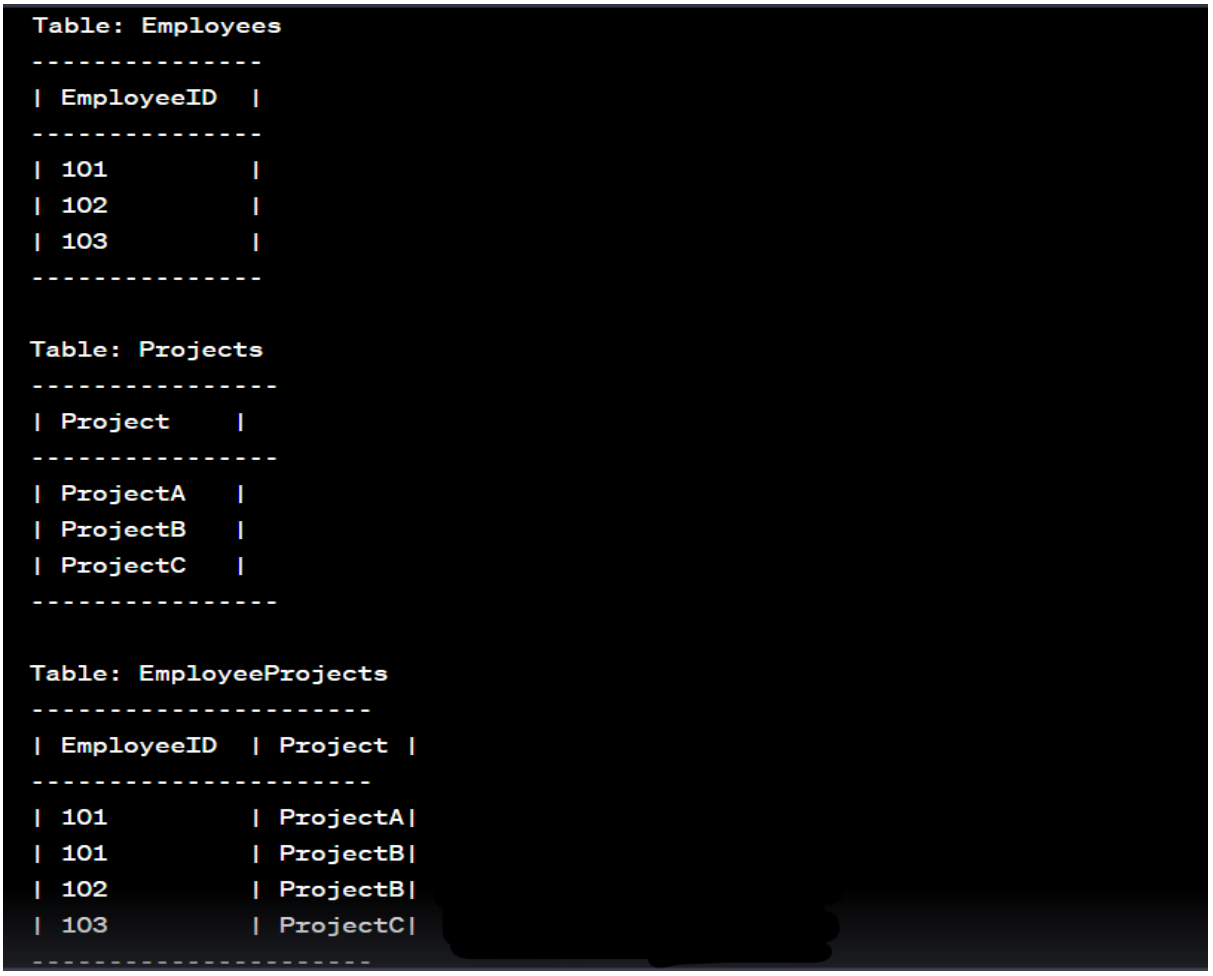
Fourth Normal Form (4NF) is an extension of BCNF and deals with multi-valued dependencies. It ensures that a table does not contain any non-trivial multi-valued dependencies. A non-trivial multi-valued dependency occurs when an attribute depends on another attribute but is independent of the primary key.

Example:
Consider the following table with multi-valued dependencies: EmployeeID → Project and Project → Employee.

```
Table: EmployeeProjects
-------------------------
| EmployeeID  | Project |
---------------------- .
| 101         | ProjectA|
| 101         | ProjectB|
| 102         | ProjectB|
| 103         | ProjectC|
-------------------------
```

In this table, EmployeeID is part of the primary key, and there is a multi-valued dependency between EmployeeID and Project. Each employee can be assigned to multiple projects, and each project can have multiple employees. This table is not in 4NF because it contains non-trivial multi-valued dependencies.

To bring this table into 4NF, we split it into two separate tables:



Now, we have separate tables for Employees, Projects, and EmployeeProjects, and there are no multi-valued dependencies. The EmployeeProjects table is in 4NF.