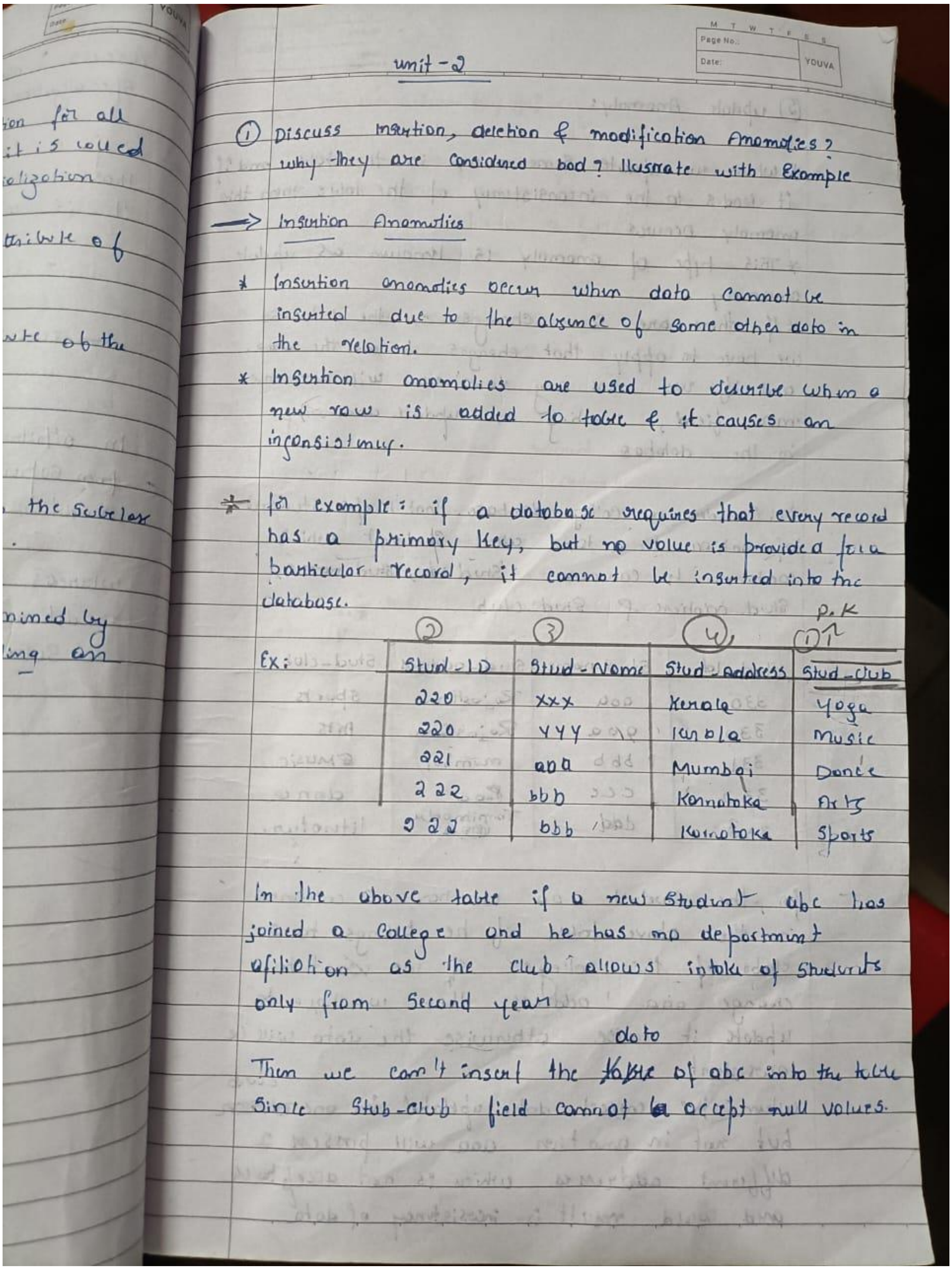
1. **Discuss insertion, deletion, and modification anomalies. Why are they considered bad? Illustrate with examples.**



ion for all
it is called
ialization

tribute of

ute of the

the subclass

ined by
ing on

unit - 2

① Discuss insertion, deletion & modification Anomalies? why they are considered bad? Illustrate with Example

→ Insertion Anomalies

* Insertion anomalies occur when data cannot be inserted due to the absence of some other data in the relation.

* Insertion anomalies are used to describe when a new row is added to table & it causes an inconsistency.

* for example: if a database requires that every record has a primary key, but no value is provided for a particular record, it cannot be inserted into the database.

Ex:

②	③	④	① P.K
Stud-ID	Stud-Name	Stud-Address	Stud-Club
220	xxx	Kerala	Yoga
220	yyy	kerala	Music
221	aaa	Mumbai	Dance
222	bbb	Karnataka	Arts
222	bbb	Karnataka	Sports

In the above table if a new student abc has joined a College and he has no department afiliation as the club allows intake of students only from Second year

Then we can't insert the ~~table~~ data of abc into the table since Stub-club field cannot ~~be~~ accept null values.

⑤ update Anomaly:

* when we update some rows in the table and if it leads to the inconsistency of the table then this anomaly occurs

* This type of anomoly is known as update anomaly

* If there are some changes in the database we have to apply that changes in all the rows. and if we miss any row we will have one more field, creating an update anomaly in the database

Ex: Consider a College database that keeps student information in a table called Student which contains 4 columns : Stud Id, stud_name, Stud_address & Stud_club

P.K

Stud-Id	Stud-name	Stud-address	Stud-club
330	aaa	Rajasthan	sports
330	aaa	Rajasthan	Arts
331	bbb	mumbai	Music
332	ccc	Goa	dance
333	ddd	Tamilnadu	literature

for the student aaa we have 2 columns in the above table if he belongs to 2 clubs of the college if we want to change aaa address then we have to update it twice otherwise the data will be inconsistent

when the correct data updated in one club but not in another aaa will possess 2 different addresses which is not acceptable and could result in inconsistency of data

③ Deletion Anomoly :

* The term deletion Anomoly in the database is used when we delete some rows in the table & any necessary additional information is also lost from the Database.

* A deletion anomoly occurs when we delete a record that may contain attributes that shouldn't be deleted

*. Ex: if a student named Stud_name add wants to leave the college then its respective P. K literature also looses the table hence if someone wants to enroll for literature he cannot enroll since literature not present this shows that if one value removed the other values are also removed with it.

3.What is a functional dependency? What are the possible sources of the information that defines the functional dependencies that hold among the attributes of a relation schema?

A functional dependency is a relationship between sets of attributes in a relation, where the value of one set of attributes determines the value of another set of attributes. In other words, a functional dependency describes the dependencies between the columns (attributes) of a relation.

In a relation schema, a functional dependency is denoted as X -> Y, where X and Y are sets of attributes. This means that for any two tuples (rows) in the relation that have the same values for the attributes in X, they must also have the same values for the attributes in Y.

Possible sources of information that define the functional dependencies among the attributes of a relation schema include:

1. Business rules and requirements: Functional dependencies can be derived from the knowledge of the business domain, business rules, and requirements. By analyzing the semantics and behavior of the data, one can identify the dependencies that exist between the attributes.

2. Expert knowledge: Experts in the domain or subject matter experts can provide insights and define the functional dependencies based on their expertise and understanding of the data.

3. Data analysis: Analyzing the existing data can reveal patterns and dependencies among the attributes. By examining the values and relationships between the columns, functional dependencies can be inferred.

4. Data dependencies: Certain dependencies may be inherent in the data itself. For example, if a relation has a primary key, the attributes that make up the primary key will be functionally dependent on the primary key itself.

5. System documentation: Documentation such as data dictionaries, entity-relationship diagrams (ERDs), and schema diagrams can provide information about the intended dependencies between attributes.

6. User input: Users of the system or stakeholders involved in the design process may provide input regarding the dependencies that should be enforced in the database.

It's important to note that defining accurate and meaningful functional dependencies is a crucial step in database design and normalization. They help ensure data integrity, eliminate redundancy, and support efficient querying and data manipulation operations.

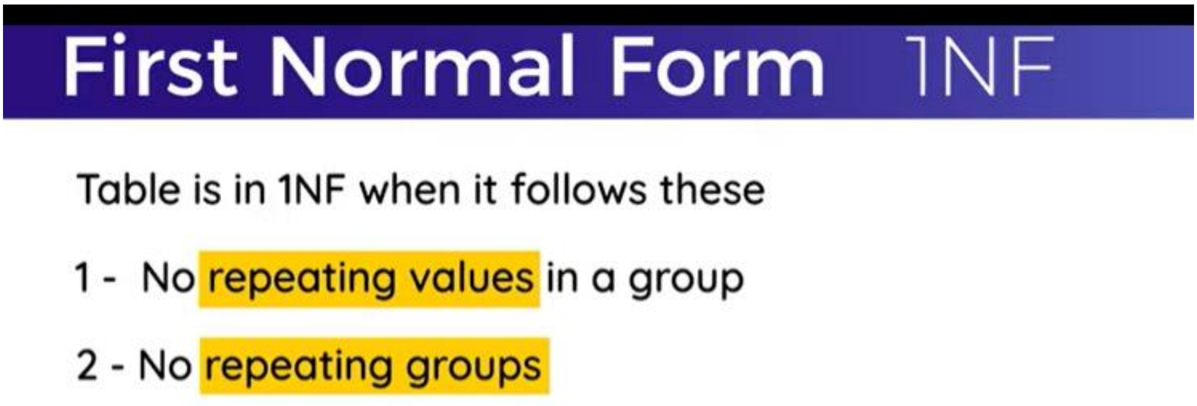
4.Define first, second, and third normal forms when only primary keys are considered. How do the general definitions of 2NF and 3NF, which consider all keys of a relation, differ from those that consider only primary keys?

In database management systems (DBMS), the first, second, and third normal forms are used to eliminate redundancy and ensure data integrity in relational databases. These normal forms are defined based on functional dependencies within a relation.

When considering only primary keys, the definitions of the second and third normal forms are the same as when considering all keys of a relation. However, the first normal form may have a slight difference. Let's look at each normal form:

1. First Normal Form (1NF):

In 1NF, the relation must have a primary key, and each attribute (column) in the relation must be atomic, meaning it cannot contain multiple values. In other words, each column should hold only a single value from its respective domain. This requirement remains the same regardless of whether all keys or only primary keys are considered.

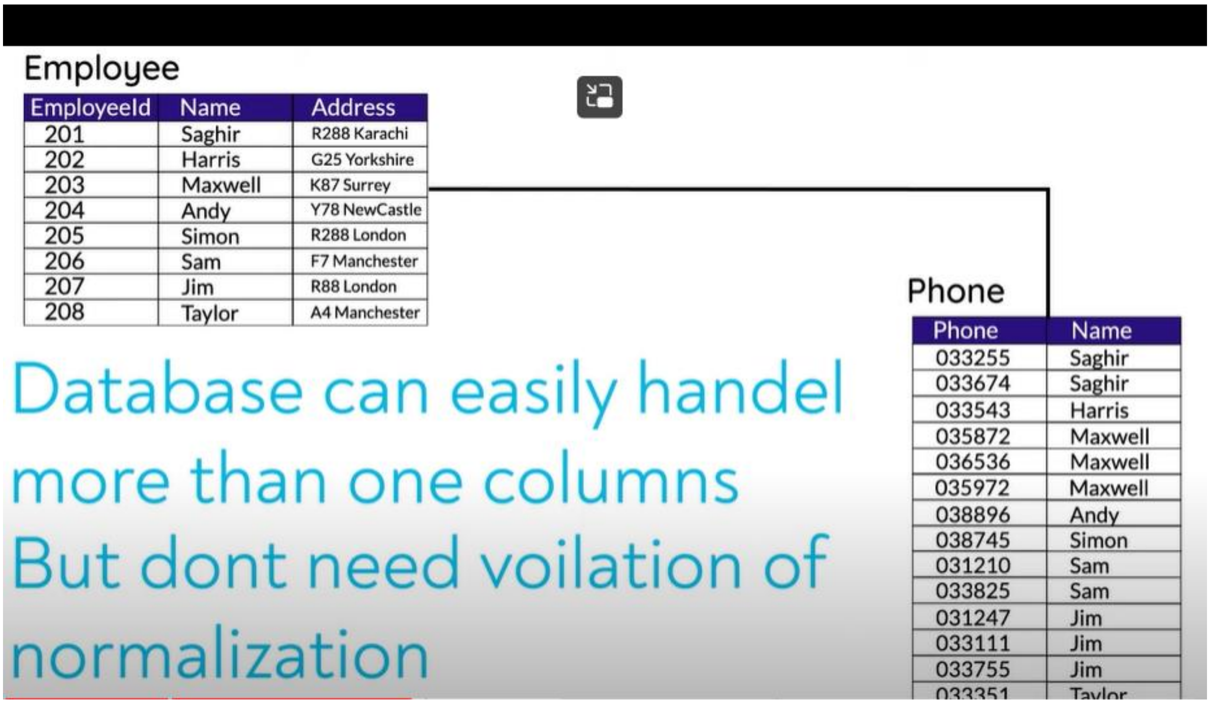


First Normal Form 1NF

Table is in 1NF when it follows these

1 - No repeating values in a group

2 - No repeating groups



Employee

EmployeeId	Name	Address
201	Saghir	R288 Karachi
202	Harris	G25 Yorkshire
203	Maxwell	K87 Surrey
204	Andy	Y78 NewCastle
205	Simon	R288 London
206	Sam	F7 Manchester
207	Jim	R88 London
208	Taylor	A4 Manchester

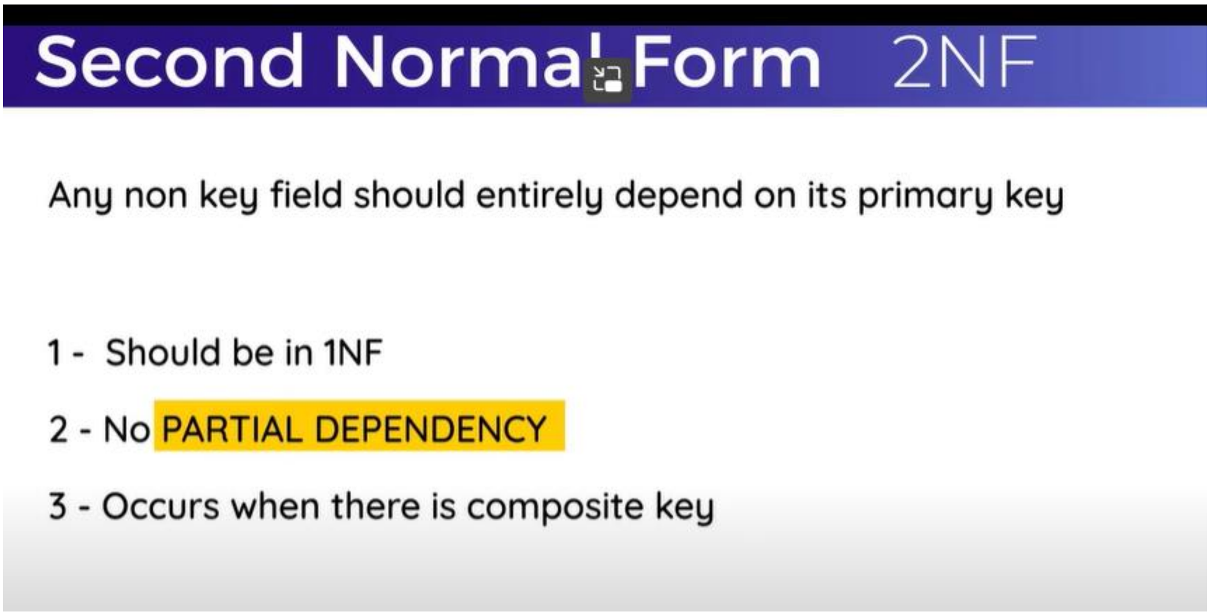
Phone

Phone	Name
033255	Saghir
033674	Saghir
033543	Harris
035872	Maxwell
036536	Maxwell
035972	Maxwell
038896	Andy
038745	Simon
031210	Sam
033825	Sam
031247	Jim
033111	Jim
033755	Jim
033351	Taylor

Database can easily handel more than one columns But dont need voilation of normalization

2. Second Normal Form (2NF):

In 2NF, the relation must satisfy the requirements of 1NF. Additionally, all non-key attributes (columns) must be functionally dependent on the entire primary key. This means that if a relation has a composite primary key consisting of multiple attributes, each non-key attribute should depend on the entire composite key, not just a part of it. The definition of 2NF remains the same, regardless of whether all keys or only primary keys are considered.



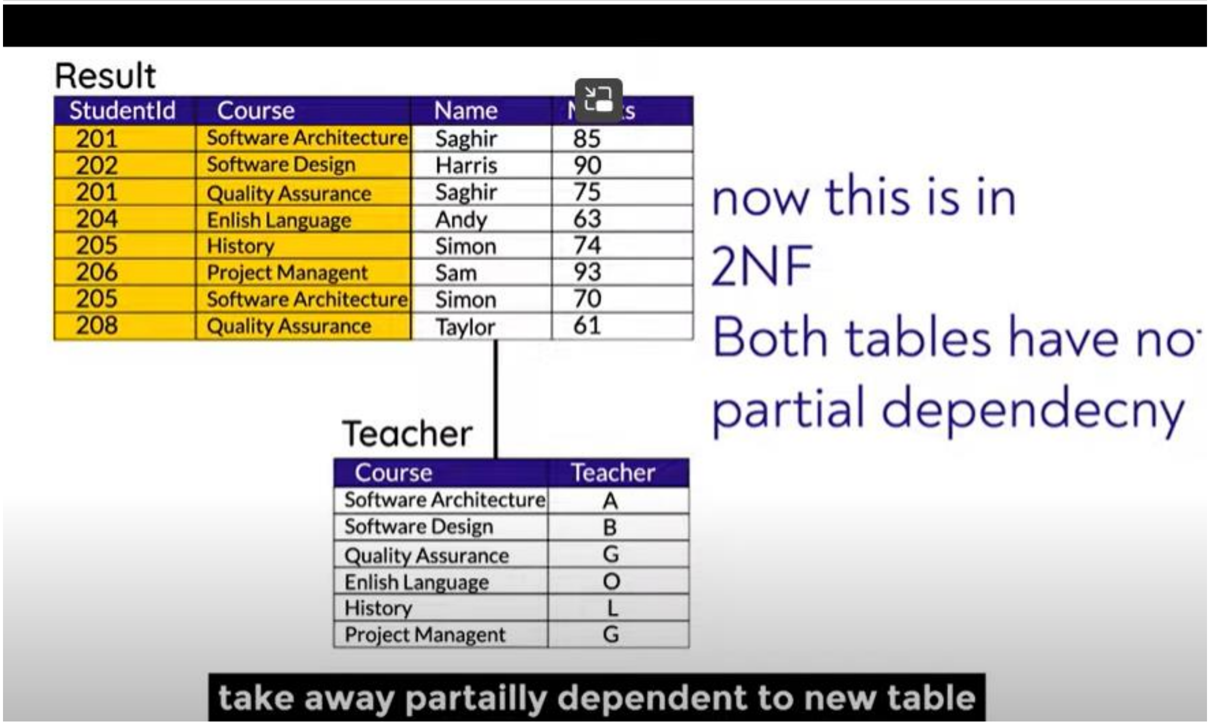
Second Normal Form 2NF

Any non key field should entirely depend on its primary key

1 - Should be in 1NF

2 - No PARTIAL DEPENDENCY

3 - Occurs when there is composite key



Result

StudentId	Course	Name	Marks
201	Software Architecture	Saghir	85
202	Software Design	Harris	90
201	Quality Assurance	Saghir	75
204	Enlish Language	Andy	63
205	History	Simon	74
206	Project Managent	Sam	93
205	Software Architecture	Simon	70
208	Quality Assurance	Taylor	61

Teacher

Course	Teacher
Software Architecture	A
Software Design	B
Quality Assurance	G
Enlish Language	O
History	L
Project Managent	G

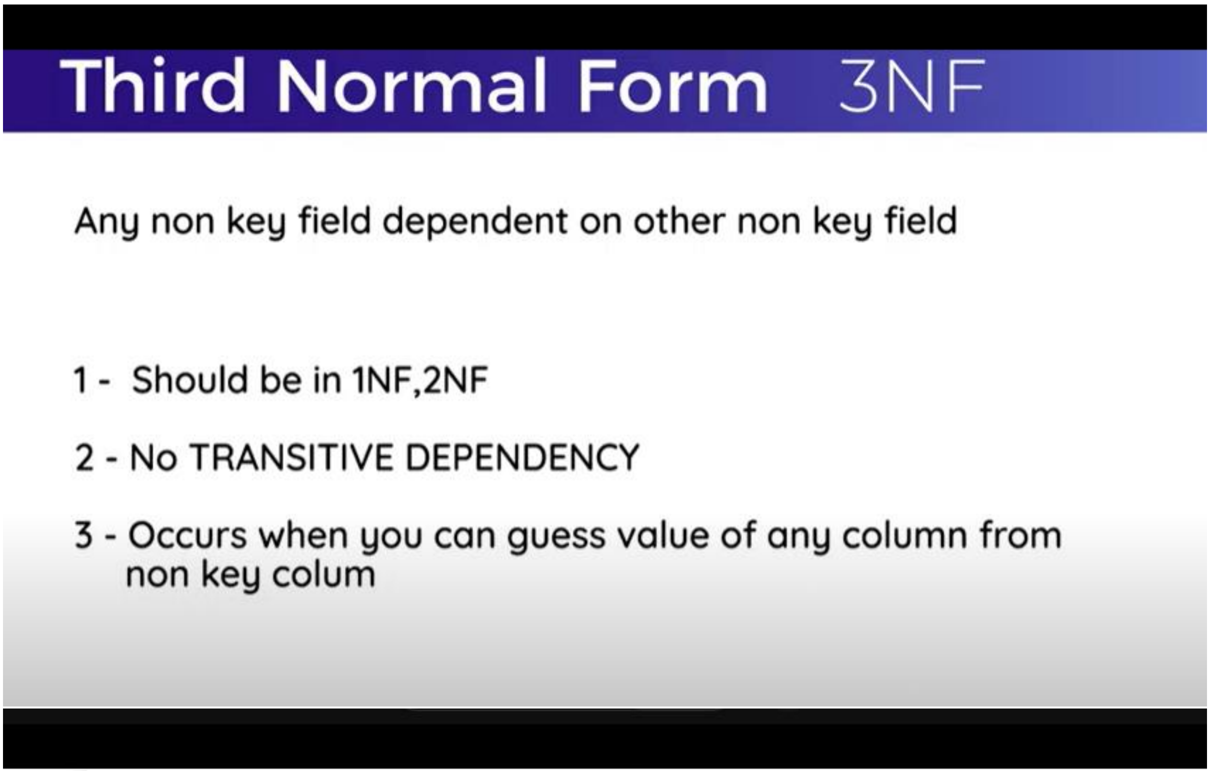
now this is in 2NF

Both tables have no partial dependecny

take away partailly dependent to new table

3. Third Normal Form (3NF):

In 3NF, the relation must satisfy the requirements of 2NF. Additionally, no non-key attribute should depend on another non-key attribute. In other words, there should be no transitive dependencies between non-key attributes. This definition also remains the same, regardless of whether all keys or only primary keys are considered.



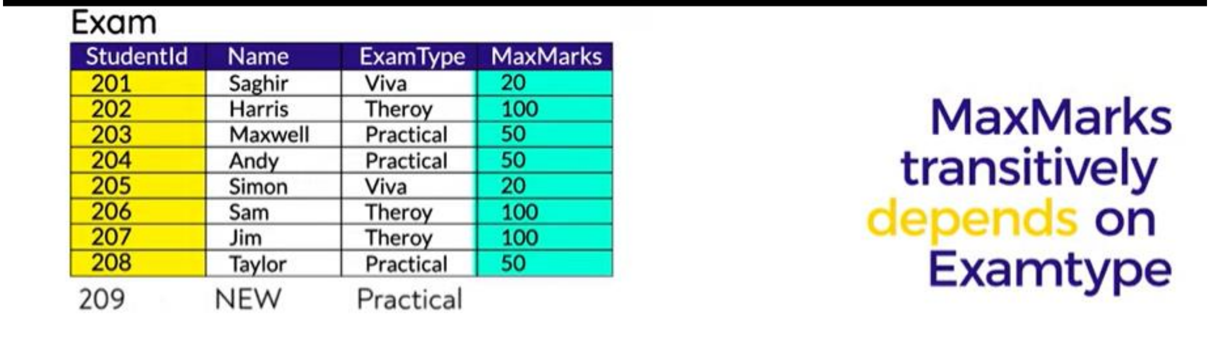
Third Normal Form 3NF

Any non key field dependent on other non key field

1 - Should be in 1NF,2NF

2 - No TRANSITIVE DEPENDENCY

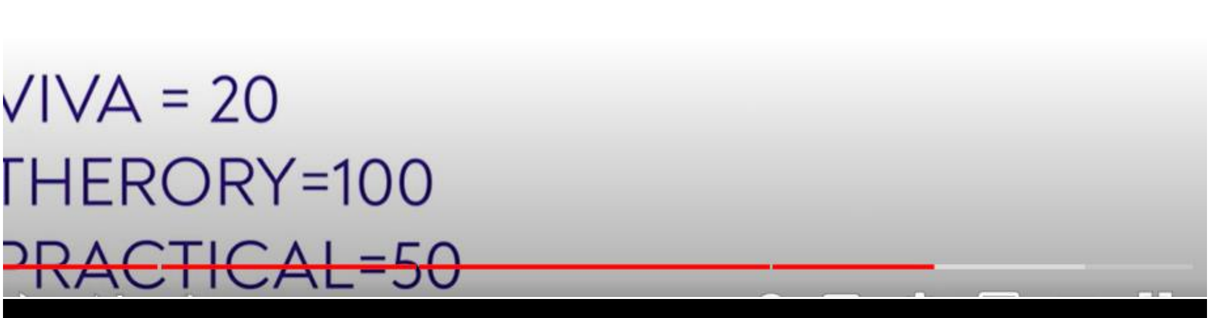
3 - Occurs when you can guess value of any column from non key colum



Exam

StudentId	Name	ExamType	MaxMarks
201	Saghir	Viva	20
202	Harris	Theroy	100
203	Maxwell	Practical	50
204	Andy	Practical	50
205	Simon	Viva	20
206	Sam	Theroy	100
207	Jim	Theroy	100
208	Taylor	Practical	50
209	NEW	Practical	

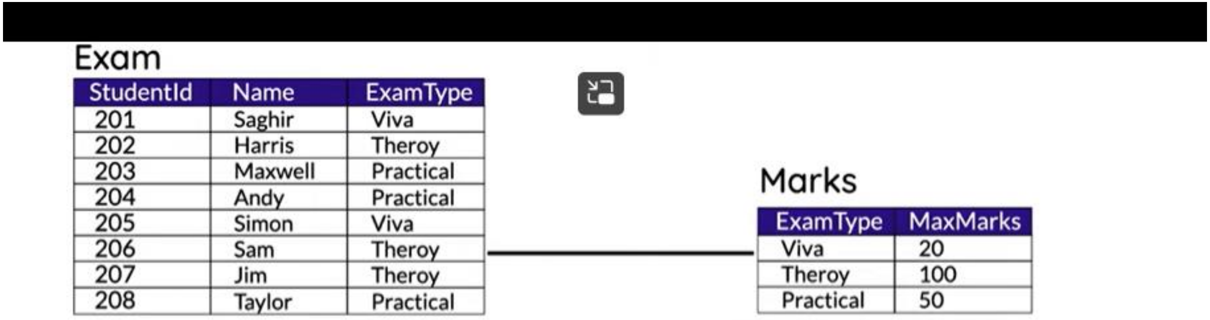
MaxMarks transitively depends on Examtype



VIVA = 20

THERORY=100

PRACTICAL=50



Exam

StudentId	Name	ExamType
201	Saghir	Viva
202	Harris	Theroy
203	Maxwell	Practical
204	Andy	Practical
205	Simon	Viva
206	Sam	Theroy
207	Jim	Theroy
208	Taylor	Practical

Marks

ExamType	MaxMarks
Viva	20
Theroy	100
Practical	50

To summarize, the general definitions of 2NF and 3NF do not differ when considering only primary keys. The distinction primarily arises when discussing the first normal form (1NF), where the requirement of having atomic attributes applies regardless of the key considered.

6.What is multivalued dependency and when does it arise? Does a relation with two or more columns always have an MVD? Show with an example.

A multivalued dependency (MVD) is a type of dependency that occurs when a relation exhibits a relationship between two sets of attributes, where each set can have multiple values independently of the other. It arises when a relation contains attributes that are functionally dependent on only a part of the primary key.

Not all relations with two or more columns have an MVD. An MVD exists when there is a non-trivial functional dependency between two sets of attributes that are not fully dependent on the primary key. Let's consider an example to illustrate this:

Suppose we have a relation called "EmployeeSkills" with the following attributes:

- EmployeeID (primary key)

- SkillSet

- Certification

In this example, the EmployeeID uniquely identifies each employee. However, an employee can have multiple skills, and each skill can have multiple certifications. Let's assume that an employee can possess multiple skills and each skill can have multiple certifications.

In this scenario, the relation "EmployeeSkills" exhibits an MVD between the attributes SkillSet and Certification. The MVD arises because each skill in the SkillSet can have multiple certifications associated with it, independently of other skills. This means that the attributes SkillSet and Certification are functionally dependent on each other, but not on the entire primary key (EmployeeID).

To illustrate this further, consider the following example instances of the "EmployeeSkills" relation:

EmployeeID	SkillSet	Certification
1	Programming	C++
1	Programming	Java
2	Database	SQL
2	Database	Oracle

In this example, EmployeeID 1 has the skill "Programming" with certifications in both C++ and Java, while EmployeeID 2 has the skill "Database" with certifications in SQL and Oracle. The MVD exists because the values in the SkillSet attribute are independent of the Certification attribute, and vice versa.

It's important to note that not all relations will have MVDs. They arise in specific cases where there are functional dependencies between attribute sets that are not fully dependent on the primary key.

5. Define Boyce Code.How does it differ from 3NF and why is it stronger than 3NF?

Boyce-Codd Normal Form (BCNF) is a normal form in database normalization theory. It is named after R. F. Boyce and E. F. Codd, who introduced it in the 1970s. BCNF is a stricter form of normalization than the third normal form (3NF).

To understand the difference between BCNF and 3NF, let's first define the third normal form. The third normal form requires that a relation must be in second normal form (2NF) and should not have any transitive dependencies. In simpler terms, it means that non-key attributes should depend only on the primary key and not on other non-key attributes

BCNF takes the concept of 3NF further by addressing a specific type of dependency called functional dependency. In BCNF, a relation is in BCNF if, for every non-trivial functional dependency (X → Y), X is a superkey. A superkey is a set of attributes that can uniquely identify each tuple in a relation. In other words, BCNF ensures that there are no non-trivial functional dependencies where the determinant (X) is not a superkey.

The key difference between BCNF and 3NF lies in the treatment of partial dependencies. In 3NF, a partial dependency occurs when a non-key attribute depends on only part of the primary key. 3NF allows partial dependencies as long as they are not transitive. However, BCNF disallows partial dependencies altogether, making it a stronger form of normalization.

Let's consider an example to illustrate the difference:

Suppose we have a relation called Students with the following attributes: StudentID (primary key), StudentName, CourseID, and CourseName. The functional dependencies are as follows:

1. StudentID → StudentName (Each student has a unique name)

2. CourseID → CourseName (Each course has a unique name)

3. StudentID, CourseID → StudentName (Each student takes a specific course.

In this example, the relation is in 3NF because there are no transitive dependencies. However, it is not in BCNF because the third functional dependency violates BCNF. The determinant (StudentID, CourseID) is not a superkey because it does not uniquely identify each tuple in the relation.

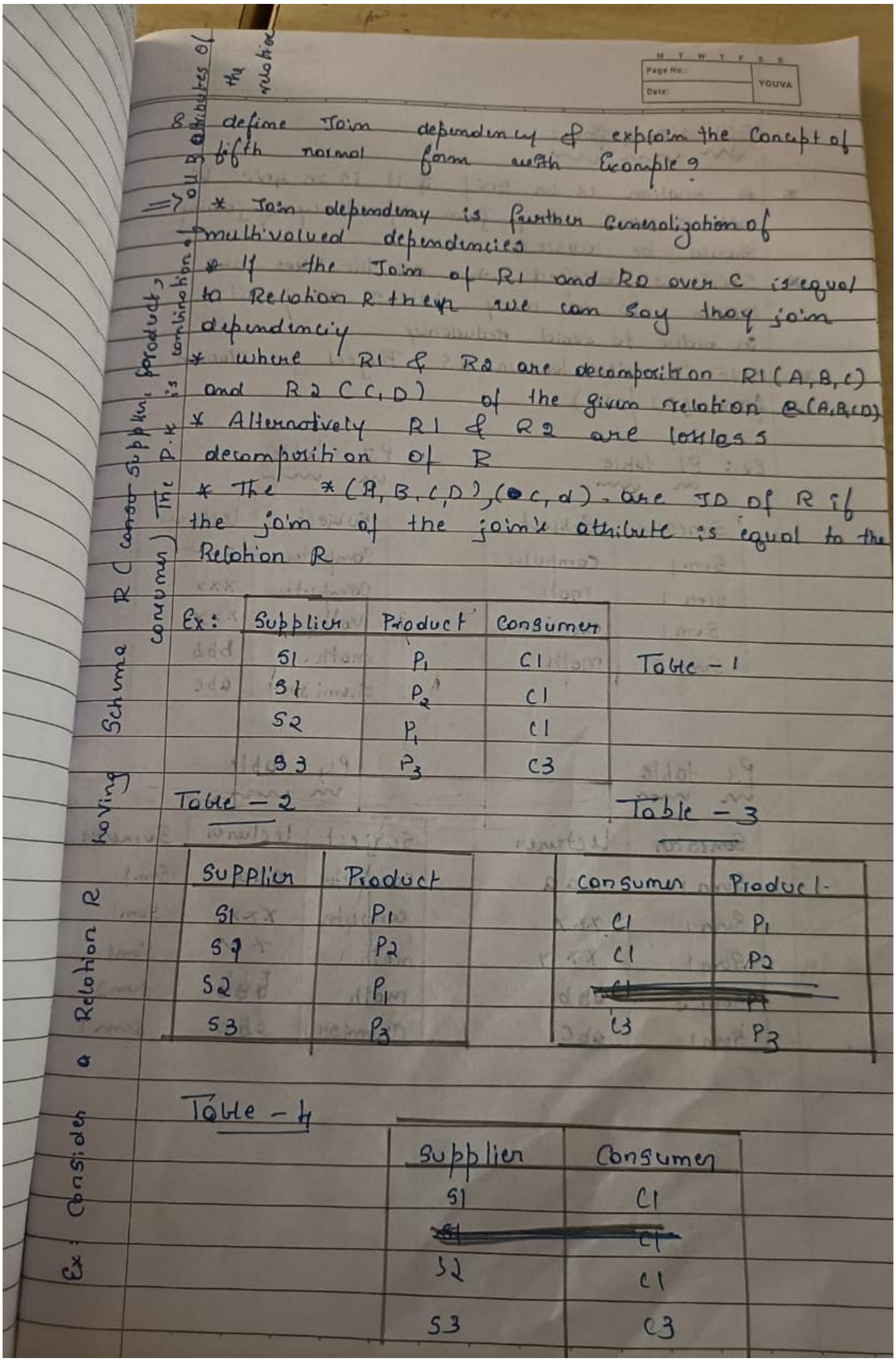
To bring the relation to BCNF, we need to decompose it into two separate relations:

1. Students (StudentID, StudentName)

2. Courses (CourseID, CourseName)

By doing so, each relation will satisfy BCNF, as the determinants are superkeys in both cases. This decomposition ensures that there are no partial dependencies and all non-trivial functional dependencies comply with BCNF.

8. Define join dependency and explain fifth normal form



Page No.:
Date:
YOUVA

8. define Join dependency & explain the Concept of fifth normal form with Example ?

⇒ * Join dependency is further Generalization of multivalued dependencies

* If the Join of R1 and R2 over C is equal to Relation R then we can say thay join dependency

* where R1 & R2 are decomposition R1 (A, B, C) and R2 (C, D) of the given relation R (A, B, C, D)

* Alternatively R1 & R2 are lossless decomposition of R

* The *(A, B, C, D), (C, d). are JD of R if the join of the join's attribute is equal to the Relation R

Ex: Consider a Relation R having Schema R (consist Supplier, Product, consumer) The P.K is combination of all 3 attributes of the relation

Ex:

Supplier	Product	Consumer
S1	P_1	C1
S1	P_2	C1
S2	P_1	C1
S3	P_3	C3

Table - 1

Table - 2

Supplier	Product
S1	P1
S1	P2
S2	P_1
S3	P_3

Table - 3

Consumer	Product
C1	P_1
C1	P2
~~C1~~	~~P1~~
C3	P3

Table - 4

Supplier	Consumer
S1	C1
~~S1~~	~~C1~~
S2	C1
S3	C3

Page No.:
Date:
YOUVA

Fifth Normal form:

- * A relation is in 5NF if it is in 4NF & not contains any join dependency & joining should be lossless
- * 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redudancy
- * 5NF is also known as Project join Normal form (PJ/NF)

Ex: P1 table

Semester	Subject
Sem 1	computer
Sem 1	math
Sem 1	chemistry
Sem 2	math

P2 table

Subject	Lecturer
Computer	aaa
Computer	xxx
math	xxx
math	bbb
chemistry	abc

P3 table

Semester	Lecturer
Sem 1	aaa
Sem 1	xxy
Sem 1	xxy
Sem 2	bbb
Sem 1	abC

P4 table

Subject	Lecturer	Semester
computer	aaa	Sem 1
computer	xxx	Sem 1
math	xxx	Sem 1
math	bbb	Sem 2
Chemistry	abc	Sem 1