Unit-05

Relational Database Design

Features of good relational designs

1) Normalization
2) Consistency and Integrity
3) Efficient Indexing
4) Appropriate Data Types
5) Clear Relationships
6) Data Integrity Constraints
7) Flexibility and Scalability
8) Optimized Query Performance
9) Documentation
10) Security Measures
11) Concurrency Control
12) Data Partitioning and Archiving
13) Backup and Recovery Planning

Keys

A key in DBMS is an attribute or a set of attributes that help to uniquely identify a tuple (or row) in a relation (or table). Keys are also used to establish relationships between the different tables and columns of a relational database. Individual values in a key are called key values.

1. Super Key

Super key is an attribute or set of attributes that can be used to identify the row in a table. It is the set of all the keys which help to identify rows in a table uniquely. This means that all those columns of a table than capable of identifying the other columns of that table uniquely will all be considered super keys. Super Key is the superset of a candidate key (explained below). The Primary Key of a table is picked from the super key set to make the table's identity attribute.

Features of Super Key:

- Uniqueness
- Redundancy Allowed
- Not Necessarily Minimal

2. Candidate Keys

Minimum subset of super key. A candidate key is a specific type of field in a relational database that can identify each unique record independently of any other data. Candidate keys are those attributes that uniquely identify rows of a table. The Primary Key of a table is selected from one of the candidate keys. So, candidate keys have the same properties as the primary keys explained above. There can be more than one candidate key in a table.

Features of Candidate Keys:

- Uniqueness
- Irreducibility
- Minimality

3. Primary Key

A primary is a single column value used to identify a database record uniquely. It has the following attributes. A primary key cannot be NULL. A primary key value must be unique. The primary key values should rarely be changed. The primary key must be given a value when a new record is inserted.

Features of Primary Keys

- Uniqueness
- Not Null
- Fixed Values
- Single Attribute or Composite
- Indexed
- Used in Relationships

4. Alternate Key

As stated above, a table can have multiple choices for a primary key; However, it can choose only one. So, all the keys which did not become the primary Key are called alternate keys.

5. Foreign Key

Foreign Key references the primary key of another Table! It helps connect your Tables. A foreign key can have a different name from its primary key. It ensures rows in one table have corresponding rows in another. Unlike the Primary key, they do not have to be unique. Foreign keys can be null even though primary keys cannot.

6. Composite Key

A composite key is a primary key composed of multiple columns used to identify a record uniquely. In our database, we have two people with the same name as Robert Phil, but they live in different places. Hence, we require both Full Name and Address to identify a record uniquely. That is a composite key.

Functional Dependencies

→ A functional dependency is a constraint that specifies the relationship between two sets of attributes where one set can accurately determine the value of other sets.
→ It is denoted as X → Y, where X is a set of attributes that can determine the value of Y.
→ Y is functionally dependent on X.
→ The attribute set on the left side of the arrow, X is called Determinant, while on the right side, Y is called the Dependent.
→ Functional Dependency helps to maintain the quality of data in the database. It plays a vital role in finding the difference between good and bad database design.

Student_ID(Primary Key)	Student_Name	Semester	Hostel
1	Anirudh	3	A3
2	Nikhil	1	N1
3	Tharun	3	T3
3	Tharun	4	T4
1	Anirudh	4	A4

In this example, if we know the value of Student_ID, we can obtain Student_Name, Semester, Hotel etc. By this, we can say that the city, Employee Name, and salary are functionally dependent on student ID.

1. Partial Functional Dependency

→ A functional dependency X->Y is a partial dependency if Y is functionally dependent on X and Y can be determined by any proper subset of X.
→ For example, we have a relationship AC->B, A->D, and D->B. Now if we compute the closure of {A+} =ADB Here A is alone capable of determining B, which means B is partially dependent on AC.

Student_id	Roll_no	course
1	5	BBA
2	6	BEDICT

Here, we can see that both the attributes Student_id and Roll_no alone can uniquely identify a course. Hence, we can say that the relationship is partially dependent.

2. Fully Functional Dependency

→ All non-key attributes are fully functionally dependent on the primary key.

→ An attribute is fully functional dependent on another attribute, if it is Functionally Dependent on that attribute and not on any of its proper subsets.

EmployeeID	EmployeeName	DeptID
Emp1	John	Dept001
Emp2	Tina	Dept003
Emp3	Carlos	Dept001

We remove the **DeptName** attribute from the Employees table and create a new table Departments:

DeptID	DeptName
Dept001	Finance
Dept002	Human Resources
Dept003	Sales

Now the relations between the tables are fully dependent, or in 2NF.

---

3. Transitive Dependency

→ Given a relation R(A,B,C) then dependency like A–>B, B–>C is a transitive dependency, since A–>C is implied

→ A Transitive Dependency is a type of functional dependency which happens when "t" is indirectly formed by two functional dependencies. Let's understand with the following Transitive Dependency Example.

Company	CEO	Age
Microsoft	Satya Nadella	51
Google	Sundar Pichai	46
Alibaba	Jack Ma	54

{Company} -> {CEO} (if we know the compay, we know its CEO's name)

{CEO} -> {Age} If we know the CEO, we know the Age

Therefore, according to the rule of rule of transitive dependency: {Company} -> {Age} should hold, that makes sense because if we know the company name, we can know his age.

Note: You need to remember that transitive dependency can only occur in a relation of three or more attributes.

4. Multivalued Dependency

Multivalued dependency occurs in the situation where there are multiple independent multivalued attributes in a single table.

A multivalued dependency is a complete constraint between two sets of attributes in a relation. It requires that certain tuples be present in a relation. Consider the following Multivalued Dependency Example to understand.

Car_model	Maf_year	Color
H001	2017	Metallic
H001	2017	Green
H005	2018	Metallic
H005	2018	Blue
H010	2015	Metallic
H033	2012	Gray

In this example, maf_year and color are independent of each other but dependent on car_model. In this example, these two columns are said to be multivalue dependent on car_model. This dependence can be represented like this:

car_model -> maf_year

car_model-> colour

Anomalies

→ Data anomalies are inconsistencies in the data stored in a database as a result of an operation such as update, insertion, and/or deletion
→ Database anomaly is normally the flaw in databases which occurs because of poor planning and storing everything in a flat database.
→ Generally, this is removed by the process of normalization which is performed by splitting/joining of tables.
→ There are three types of anomalies: update, deletion, and insertion anomalies.
→ For example, each employee in a company has a department associated with them as well as the student group they participate in.

1. Insertion Anomaly

→ Insertion Anomalies happen when inserting vital data into the database is not possible because other data is not already there.
→ For example, if a system is designed to require that a customer be on file before a sale can be made to that customer, but you cannot add a customer until they have bought something, then you have an insert anomaly.

id	Student	Course id	Course name	
001	Samikshya	002	Java	
002	Samjhana	005	C++	
002	Samjhana	005	C++	
003	Saru	NULL	NULL	

If course is new then first we must add the course in database and then only assign course id and name to student . This creates insertion anomalies

2. Update Anomaly

→ An update anomaly is a data inconsistency that results from data redundancy and a partial update.
→ For example, to change an employee's title due to a promotion.

→ If the data is stored redundantly in the same table, and the person misses any of them, then there will be multiple titles associated with the employee. The end user has no way of knowing which is the correct title.

3. Deletion Anomaly

→ A deletion anomaly is the unintended loss of data due to deletion of other data.

→ For example, For example, if a single database record contains information about a particular product along with information about a salesperson for the company and the salesperson quits, then information about the product is deleted along with salesperson information.

Decomposition

→ The term decomposition refers to the process in which we break down a table in a database into various elements or parts. Thus, decomposition replaces a given relation with a collection of various smaller relations. Thus, in a database, we can make any table break down into multiple tables when we want to collect a particular set of data.

→ Decomposition must always be lossless. This way, we can rest assured that the data/information that was there in the original relation can be reconstructed accurately on the basis of the decomposed relations. In case the relation is not decomposed properly, then it may eventually lead to problems such as information loss.

Types of Decomposition

1. Lossless Decomposition

A decomposition is said to be lossless when it is feasible to reconstruct the original relation R using joins from the decomposed tables. It is the most preferred choice. This way, the information will not be lost from the relation when we decompose it. A lossless join would eventually result in the original relation that is very similar.

For example, show 2nF Decomposition

2. Lossy Decomposition

Just like the name suggests, whenever we decompose a relation into multiple relational schemas, then the loss of data/information is unavoidable whenever we try to retrieve the original relation. Example : make yourself.

Normal forms: 1NF, 2NF, 3NF and BCNF

Normalization is a process in relational database design that organizes tables and their attributes to reduce data redundancy and dependency. The goal is to eliminate data anomalies, improve data integrity, and make the database structure more efficient. There are several normal forms, each addressing specific issues related to data organization. The most common normal forms are:

1. 1NF (First Normal Form)

Rules:

Rule 1: Each table cell should contain a single value.

Rule 2: Each record needs to be unique.

→ It was defined to disallow multivalued attributes, composite attributes, and their combinations. It states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. REMEDY:Form new relations for each multivalued attribute or nested relation.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

1NF Example

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

---

2. Second Normal Form (2NF)

Rules:

Rule 1: Be in 1NF

Rule 2: it should not have Partial Dependency.

→ The second normal form (2NF) is based on the concept of full functional dependency. A functional dependency X Y is a full functional dependency if removal of any attribute A from X means that the dependency does not hold anymore. A functional dependency X Y is a partial dependency if

some attribute A & X can be removed from X and the dependency still holds. Definition: A relation schema R is in 2NF if every nonprime attribute A in R is fully functionally dependent on the primary key of R. The test for 2NF involves testing for functional dependencies whose left-hand side attributes are part of the primary key.

1NF Example

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

---

3. Third Normal Form (3NF)

Rules:

Rule 1: Be in 2NF

Rule 2: Has no transitive functional dependencies, must have

To move our 2NF table into 3NF, we again need to divide our table.

→ It is based on the concept of transitive dependency. A functional dependency X → Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both X → Z and Z → Y hold. Definition: A relation schema R is in 3NF if it satisfies 2NF and no nonprime attribute of R is transitively dependent on the primary key. REMEDY: Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s)

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

---

4. BCNF

→ Boyce-Codd Normal Form (BCNF) is a higher normal form in relational database design, addressing certain types of dependencies that may still exist even after applying the Third Normal Form (3NF). BCNF is an extension of the Third Normal Form and ensures that a relation is free from certain types of anomalies related to functional dependencies.

- 4NF (Fourth Normal Form) Rules

– If no database table instance contains two or more, independent and multivalued data describing the relevant entity, then it is in 4th Normal Form.

- 5NF (Fifth Normal Form) Rules

– A table is in 5th Normal Form only if it is in 4NF and it cannot be decomposed into any number of smaller tables without loss of data.