NEERAJ®

ADVANCED DATABASE MANAGEMENT SYSTEMS

By:
Ashok Arora
B.Sc., PGDCA, A Level, B Level

Reference Book
Including
Solved Question Papers

New Edition

NEERAJ PUBLICATIONS
(Publishers of Educational Books)
(An ISO 9001 : 2008 Certified Company)
1507, Ist Floor, NAI SARAK, DELHI - 110 006
Ph.: 011-23260329, 45704411, 23244362, 23285501 Off. Mob. : 8510009878
E-mail: **info@neerajignoubooks.com**
Website: **www.neerajignoubooks.com**

Price
₹ 160/-

Published by:

NEERAJ PUBLICATIONS

Admn. Office : **Delhi-110 007**

Sales Office : **1507, 1st Floor, Nai Sarak, Delhi-110 006**

E-mail: **info@neerajignoubooks.com** *Website*: **www.neerajignoubooks.com**

Reprint Edition	*Typesetting by*: *Competent Computers*	*Printed at*: *Novelty Printer*

Notes:

1. *For the best & upto-date study & results, please prefer the recommended textbooks/study material only.*
2. *This book is just a Guide Book/Reference Book published by NEERAJ PUBLICATIONS based on the suggested syllabus by a particular Board /University.*
3. *The information and data etc. given in this Book are from the best of the data arranged by the Author, but for the complete and upto-date information and data etc. see the Govt. of India Publications/textbooks recommended by the Board/University.*
4. *Publisher is not responsible for any omission or error though every care has been taken while preparing, printing, composing and proof reading of the Book. As all the Composing, Printing, Publishing and Proof Reading etc. are done by Human only and chances of Human Error could not be denied. If any reader is not satisfied, then he is requested not to buy this book.*
5. *In case of any dispute whatsoever the maximum anybody can claim against NEERAJ PUBLICATIONS is just for the price of the Book.*
6. *If anyone finds any mistake or error in this Book, he is requested to inform the Publisher, so that the same could be rectified and he would be provided the rectified Book free of cost.*
7. *The number of questions in NEERAJ study materials are indicative of general scope and design of the question paper.*
8. *Question Paper and their answers given in this Book provide you just the approximate pattern of the actual paper and is prepared based on the memory only. However, the actual Question Paper might somewhat vary in its contents, distribution of marks and their level of difficulty.*
9. *Subject to Delhi Jurisdiction only.*

NEERAJ PUBLICATIONS

(Publishers of Educational Books)

(An ISO 9001 : 2008 Certified Company)

1507, 1st Floor, NAI SARAK, DELHI - 110 006

Ph. 011-23260329, 45704411, 23244362, 23285501, Off. Mob. : 8510009878

E-mail: **info@neerajignoubooks.com** *Website:* **www.neerajignoubooks.com**

CONTENTS

ADVANCED DATABASE MANAGEMENT SYSTEMS

■■

Sample Preview
of the
Solved
Sample Question
Papers

Published by:

NEERAJ
PUBLICATIONS
www.neerajbooks.com

QUESTION PAPER

(June – 2016)

(Solved)

ADVANCED DATABASE MANAGEMENT SYSTEMES

Time: 3 Hours] *[Maximum Marks : 100*

Note: *Question number* ***1*** *is* ***compulsory****. Answer any three questions from the rest.*

Q. 1. ***(a)*** **What are Triggers? Explain the significance of triggers with the help of an example.**

Ans. Ref.: See Chapter 3, Page No. 18, 'Triggers' and Page No. 19, Q.No. 8.

(b) **Explain the Log-Based recovery scheme, using deferrred database modification approach. Give a suitable example in your explanation.**

Ans. Other special log records exist to record significant events during transaction processing, such as the start of a transaction and the commit or abort of a transaction. We denote the various types of log records as:

- <Ti start>.Transaction Ti has started.
- <Ti, Xj, V1, V2> Transaction Ti has performed a write on data item Xj. Xj had value V1 before write, and will have value V2 after the write.
- < Ti commit> Transaction Ti has committed.
- < Ti abort> Transaction Ti has aborted.

Whenever a transaction performs a write, it is essential that the log record for that write be created before the database is modified. Once a log record exists, we can output the modification that has already been output to the database. Also we have the ability to undo a modification that has already been output to the database, by using the old-value field in the log records.

For log records to be useful for recovery from system and disk failures, the log must reside on stable storage. However, since the log contains a complete record of all database activity, the volume of data stored in the log may become unreasonable large.

The deferred-modification technique ensures transaction atomicity by recording all database modifications in the log, but deferring all write operations of a transaction until the transaction partially commits (i.e., once the final action of the transaction has been executed). Then the information in the logs is used to execute the deferred writes. If the system crashes or if the transaction aborts, then the information in the logs is ignored.

For example, T0 be transaction that transfers $50 from account A to account B:

```
T0: read (A);
  A: = A-50;
  Write (A);
   Read (B);
 B: = B + 50;
  Write (B).
```

(c) **What are mobile databases? List the characteristics of mobile databases. Discuss the challenges of implementing mobile databases.**

Ans. Ref.: See Chapter-14, Page No. 107, 'Mobile & Personal Database' and 'Characteristics of Mobile Databases' and Page No. 108, 'Distributed Data Management Issues in Mobile Computing'.

(d) **List the index implementation available in PostgreSQL. Explain each index available in PostgreSQL.**

Ans. PostgreSQL provides several index types: B-tree, Hash, GiST, SP-GiST and GIN. Each index type uses a different algorithm that is best suited to different types of queries. By default, the CREATE INDEX command creates B-tree indexes, which fit the most common situations.

B-trees can handle equality and range queries on data that can be sorted into some ordering. In particular, the PostgreSQL query planner will consider using a B-tree index whenever an indexed column is involved in a comparison using one of these operators:

<
<=
=
>=
>

Constructs equivalent to combinations of these operators, such as BETWEEN and IN, can also be implemented with a B-tree index search. Also, an IS NULL or IS NOT NULL condition on an index column can be used with a B-tree index.

The optimizer can also use a B-tree index for queries involving the pattern matching operators LIKE and ~ if the pattern is a constant and is anchored to the beginning of the string - for example, col LIKE 'foo%' or col ~ '^foo', but not col LIKE '%bar'. However, if your database does not use the C locale you will need to create the index with a special operator class to support indexing of pattern-matching queries. It is also possible to use B-tree indexes for ILIKE and ~*, but only if the pattern starts with non-alphabetic characters, i.e., characters that are not affected by upper/lower case conversion.

B-tree indexes can also be used to retrieve data in sorted order. This is not always faster than a simple scan and sort, but it is often helpful.

Hash indexes can only handle simple equality comparisons. The query planner will consider using a hash index whenever an indexed column is involved in a comparison using the = operator. The following command is used to create a hash index:

```
CREATE INDEX name ON table USING
hash (column);
```

GiST indexes are not a single kind of index, but rather an infrastructure within which many different indexing strategies can be implemented. Accordingly, the particular operators with which a GiST index can be used vary depending on the indexing strategy (the operator class). As an example, the standard distribution of PostgreSQL includes GiST operator classes for several two-dimensional geometric data types, which support indexed queries using these operators:

<<
&<
&>
>>
<<|
&<|
|&>
|>>
@>
<@
~=
&&

Many other GiST operator classes are available in the contrib collection or as separate projects.

GiST indexes are also capable of optimizing "nearest-neighbor" searches, such as

```
SELECT * FROM places ORDER BY location
<-> point '(101,456)' LIMIT 10;
```

which finds the ten places closest to a given target point. The ability to do this is again dependent on the particular operator class being used.

SP-GiST indexes, like GiST indexes, offer an infrastructure that supports various kinds of searches. SP-GiST permits implementation of a wide range of different non-balanced disk-based data structures, such as quadtrees, k-d trees, and suffix trees (tries). As an example, the standard distribution of PostgreSQL includes SP-GiST operator classes for two-dimensional points, which support indexed queries using these operators:

<<
>>
~=
<@
<^
>^

GIN indexes are inverted indexes which can handle values that contain more than one key, arrays for example. Like GiST and SP-GiST, GIN can support many different user-defined indexing strategies and the particular operators with which a GIN index can be used vary depending on the indexing strategy. As an example, the standard distribution of PostgreSQL includes GIN operator classes for one-dimensional arrays, which support indexed queries using these operators:

<@
@>
=
&&

Sample Preview of The Chapter

Published by:

NEERAJ PUBLICATIONS

www.neerajbooks.com

ADVANCED DATABASE MANAGEMENT SYSTEMS

DATABASE DESIGN AND IMPLEMENTATION

Relational Database Design

1

INTRODUCTION

A relational database is designed to minimise data redundancy, improve data consistency and sharing, data independence, etc. Enhanced-relationship (EER) model results from extending the original E-R model. In this chapter, we will learn the various tools for EER modeling. We will mainly discuss about functional dependency, multivalued dependency, normalisation, various normal forms, join dependency, inclusion dependency and temporal data.

CHAPTER AT A GLANCE

FEATURES OF GOOD DATABASE DESIGN

The various features of a good database design are as follows:

- Minimal data redundancy
- Improved data consistency
- Improved data sharing
- Improved data quality
- Improved data accessibility and responsiveness
- Enforcement of standards
- Increased productivity of application development
- Reduced program maintenance
- Data independence

ENHANCED ER TOOLS

Enhanced-relationship (EER) model is the model that has resulted from extending the original E-R model with the following new modeling constructs:

1. Supertype/Subtype relationship
2. Inheritance
3. Specialisation
4. Generalisation

1. Supertype/Subtype Relationship: The E-R model has been extended to include Supertype/Subtype relationships. A supertype is a generic entity type that has a relationship with one or more subtypes, whereas a subtype is a sub-grouping of those entities in an entity type, which is meaningful to the organisation.

2. Inheritance: Inheritance is the property by which subtype entities inherit values of all attributes of the supertype. A subtype is an entity type in its own right. An entity instance of a subtype represents the same entity instance of the supertype. An entity in a subtype must possess not only values for its own attributes, but also values for its attributes as a member of the supertype. The inheritance property makes it unnecessary to include supertype attributes redundantly with the subtypes.

3. Specialisation: Specialisation is the top-down process of defining one or more subtypes of the supertype, and forming supertype/subtype relationships. Each subtype is formed based on some distinguishing features such as attributes or relationships that are specific to the subtype.

4. Generalisation: Generalisation is the bottom-up process of defining a more general entity type from a set of more specialised entity types. Thus, generalisation groups the entity types along with their common attributes, while at the same time it preserves attributes that are specific to each subtype.

Completeness Constraint

Completeness constraint means whether an instance of a supertype must also be a member of at least one subtype. It has two possibilities: Total specialisation rule and Partial specialisation rule.

Disjoint Constraint

Disjointness constraint means whether an instance of a supertype may simultaneously be a member of two or more subtypes. It also has two possibilities: Disjoint rule and Overlap rule.

Converting EER Diagram to Table

EER diagram can be converted into a table by creating a table for each strong and each weak entity set. A table is also created for each binary m : n relationship set and a binary m : 1 or 1 : m relationship. For generalisation or specialisation hierarchy separate tables can be created for higher level and lower level entities. A separate table is created for composite attributes.

FUNCTIONAL DEPENDENCY: THEORY AND NORMALISATION

A functional dependency is a constraint between two attributes or two sets of attributes. It means for any relation R, attribute B is functionally dependent on attribute A if, for every valid instance of A, that value of A uniquely determines the value of B. The functional dependency of B on A is represented as follows:

$$A \rightarrow B$$

An attribute may be functionally dependent on two or more attributes. The attribute on the left-hand side of the arrow in a functional dependency is called a determinant. So, A is a determiner.

Primary Key, Composite Key and Candidate Key

Primary key is an attribute or a combination of attributes that uniquely identifies each row in a table. For example, ISBN could be a primary key in the relation 'Book'. Composite key is a primary key that consists of more than one attribute. For example, Account_Type and ACNO could be a composite key in relation 'Account'.

A candidate key is an attribute, or combination or attributes, that uniquely identifies a row in a relation. For example, Vehicle_No could be a candidate key in relation 'Vehicle' if all other attributes functionally dependent on Vehicle_No.

Normalisation

Normalisation is the process of decomposing relations with anomalies to produce smaller, well-structured relations. It is based on the analysis of functional dependencies. It can be accomplished in stages, each of which is called a normal form. A normal form is a state of a relation that results from applying simple rules regarding functional dependencies to that relation. The various normal forms are:

1. First normal form
2. Second normal form
3. Third normal form
4. Boyce/Codd normal form
5. Fourth normal form
6. Fifth normal form

MULTIVALUED DEPENDENCIES AND FOURTH NORMAL FORM

Boyce-Codd Normal Form

A relation is in Boyce-Codd normal form (BCNF) if and only if every determinant in the relation is a candidate key.

Fourth Normal Form

When a relation is in BCNF, there are no more any anomalies, which result from functional dependencies. But, some anomalies may result from multi-valued dependencies. A type of dependency that exists when there are at least three attributes (such as A, B, and C) in a relation, and for each value of A there is a well-defined set of values of B and a well-defined set of values of C. But the set of values of B is independent of set C, and *vice versa*. To remove the multivalued dependency from a relation, the relation is divided into two new relations. Each of these tables contains two attributes that have a multivalued relationship in the original relation. So, a relation is in fourth normal form (4NF) if it is in BCNF and contains no multivalued dependencies.

JOIN DEPENDENCIES AND 5NF/PJNF

The fifth normal form deals with a property called Lossless Joins, which occurs very rarely and is difficult to detect in practice. It deals with join-dependencies and cannot be constructed from several smaller relations. A relation R satisfies join dependency * $(R_1, R_2, ..., R_n)$ if and only if R is equal to the join of $R_1, R_2, ..., R_n$ where R_i are subsets of the set of attributes of R. A relation R is in fifth normal form if for all join dependencies, either $(R_1, R_2, ..., R_n)$ is a trivial join-dependency or every R_i is a candidate key for R

Fifth normal form is also called **Project-Join Normal Form (PJNF)**. It is also defined as follows:

A relation schema R having a set F of functional, multivalued, and join dependencies, is in PJNF (5NF), if for all the join dependencies in the closure of F that are of the form, $*(R_1, R_2, ..., R_n)$ where R_i are subsets of the set of attributes of R and either $(R_1, R_2, ..., R_n)$ is a trivial join-dependency or every R_i is a super key for R.

Definitions of PJNF from the viewpoint of the decomposition

Definition 1: A join dependency $*(R_1, R_2, ..., R_n)$ over a relation R is trivial if it is satisfied by every relation *r*(R).

Definition 2: A join dependency $*(R_1, R_2, ..., R_n)$ over a relation R applies to a relation scheme R if $R = R_1, R_2, ..., R_n$.

Definition 3: If R be a relation scheme having F as the set of Functional Dependencies and Join Dependencies over R, R will be in project-join normal form (PJNF) if for every Join Dependency $*(R_1,R_2,...,R_n)$ which can be derived by F that applies to R, either the Join Dependency is trivial or every R_i is a super key for R.

Definition 4: If R be a relation scheme having F as the set of Functional Dependencies and Join Dependencies over R, R will be in project-join normal form (PJNF) if for every Join Dependency $*(R_1,R_2,...,R_n)$ which can be derived by F that applies to R, is implied by the key Functional Dependencies of R.

INCLUSION DEPENDENCIES AND TEMPLATE DEPENDENCIES

Inclusion dependency formalise two important types of interrelation constraints that exist between relations – Referential integrity constraints and Class/subclass relationships.

Inclusion Dependency

An inclusion dependency, R.X < S.Y between two sets of attributes (X of a relation schema R, and Y of a relation schema S) is defined as follows:

If *r* and *s* are the relation state of R and S respectively at any specific time then $x(r(R)) = y(s(S))$

Template Dependencies

The template dependencies generalises the concepts of Inclusion Dependencies. A template dependency resembles a tableau. It is representation of the statement that a relation is invariant under a certain tableau mapping. It consists of a number of hypothesis rows that define certain variables with a special row at the bottom, which is called the conclusion row.

A relation *r* satisfies a template dependency, if and only if, a valuation that successfully maps the hypothesis rows to tuple in a relation *r*, finds a map for conclusion row to a tuple in *r*.

A template dependency (TD) on a relation scheme R is a pair T = (T,*w*) where $T=\{w_1,w_2, ... w_k\}$ is a set of hypothesis rows on R, and *w* is a single conclusion row on R. A relation *r*(R) satisfies TD T if for every valuation *p* of T such that *p*(T) = *r*, *p* can be extended to show that *p*(*w*) E *r*.

A template dependency is trivial if every relation over R satisfies it.

Difference between Template Dependency and Tableau

A Template Dependency differs from a tableau in the ways that a variable like (*a*, *b*, *c*, etc.) in the conclusion row need not appear in any of the hypothesis row, and variables may not be necessarily restricted to a single column.

DOMAIN KEY NORMAL FORM (DKNF)

DKNF is an attempt to define an ultimate normal form that takes into account all possible types of dependencies and constraints. A set of relations that are in DKNF must be free of anomalies. It is an important normal form as it offers a complete solution to avoid the anomalies.

MODELING TEMPORAL DATA

Temporal databases are those database applications that involve some aspect of time. In a temporal database system, we need to define database as a sequence of time based data in chronological order and need to resolve events that happen at the same time. The SQL support for temporal data includes data types such as DATE, TIME, TIMESTAMP, INTERVAL, PERIOD, etc.

CHECK YOUR PROGRESS

Q. 1. What is the use of EER diagram?

Ans. Enhanced-relationship (EER) model extends the original E-R model with the new modeling constructs such as Supertype /Subtype relationship, Inheritance, Specialisation, and Generalisation.

Q. 2. What are the constraints used in EER diagrams?

Ans. The basic constraints used in EER diagram are completeness and disjoint constraints.

Completeness Constraint: Completeness constraint means whether an instance of a supertype must also be a member of at least one subtype. It has two possibilities:

(a) **Total specialisation rule:** It specifies that each entity instance of the supertype must be a member of some subtype in the relationship.

(b) **Partial specialisation rule:** It specifies that an entity instance of the supertype cannot belong to any subtype.

Disjoint Constraint: Disjointness constraint means whether an instance of a stipertype may simultaneously be a member of two or more subtypes. It also has two possibilities:

(a) **Disjoint rule:** It specifies that if an entity instance of the supertype is a member of one subtype, it cannot simultaneously be a member of any other subtype.

(b) **Overlap rule:** It specifies that an entity instance can simultaneously be a member of two subtypes.

Q. 3. How is an EER diagram converted into a table?

Ans. EER diagram can be converted into a table by following these rules:

1. A table is created for each strong entity set.
2. A table is created for a weak entity set by including the primary key of the strong entity on which it depends.
3. A table is created for each binary m : n relationship set having the primary keys of both the participating entities.
4. For a binary m : 1 or 1 : m relationship, the primary key on the m side should be used as the primary key.
5. For generalisation or specialisation hierarchy separate tables can be created for higher level and lower level entities.
6. A separate table is created for composite attributes.

Q. 4. What are Multi-valued Dependencies? When we can say that a constraint X is multi-determined?

Ans. A multi-valued dependency is a constraint due to multi-valued attributes. A relation is in Boyce-Codd normal form (BCNF) if and only if every determinant in the relation is a candidate key. When a relation is in BCNF, there are no more any anomalies, which result from functional dependencies. But, some anomalies may result from multi-valued dependencies. A type of dependency that exists when there are atleast three attributes (such as A, B, and C) in a relation, and for each value of A there is a well-defined set of values of B and a well-defined set of values of C. But the set of values of B is independent of set C, and *vice versa*. To remove the multivalued dependency from a relation, the relation is divided into two new relations. Each of these tables contains two attributes that have a multivalued relationship in the original relation. So, a relation is in fourth normal form (4NF) if it is in BCNF and contains no multivalued dependencies.

A relation must have atleast three attributes out of which two should be multi-valued. So, a constraint X is multi-determined if atleast two of its attributes are multi-valued.

Q. 5 Decompose the following into 4NF.

EMP

ENAME	PNAME	DNAME
Dev	X	Sanju
Dev	Y	Sainyam
Dev	X	Sainyam
Dev	Y	Sanju

Ans.

EMP PROJECTS

ENAME	PNAME
DeV	X
Dev	Y

EMP DEPENDENTS

ENAME	DNAME
Dev	Saniu
Dev	Sainyam

Q. 6. Does the following relation satisfy MVDs with 4NF? Decompose the following relation into 5NF.