A CRITIQUE ON FINAL TERM TOPICS

Normalization:

Normalization is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships.

Anomaly:

An error or inconsistency that may result when a user attempts to update a table that contains redundant data. Anomalies are caused when there is too much redundancy in the database's information.

There are three types of Anomaly:

1. Insertion Anomaly,

2. Deletion Anomaly,

3. Modification Anomaly.

Normal Form Rules:

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce-Codd Normal Form (BCNF)
- Fourth Normal Form (4NF)

First Normal Form (1NF)

A relation that contains no multivalued Attributes.

Second Normal Form (2NF)

A relation in First Normal Form in which every attribute in fully functionally dependent in the primary key or Partial Functional dependency should be removed.

A CRITIQUE ON FINAL TERM TOPICS

Partial Functional Dependency:

A functional dependency in which one or more non-key attribute are functionally dependent in part (but not all) of the primary key.

Functional Dependency:

A constrain between two attribute or two sets of attributes. The attributes of a table is said to be dependent on each other when an attribute of a table uniquely identifies another attribute of the same table.

Third Normal Form (3NF)

A relation in Second Normal Form has no Transitive Dependency present.

Transitive Dependency

A Functional Dependency between two (or more) non-key attributes which is said to be transitive if it is indirectly formed by two functional dependencies.

Boyce-Codd Normal Form(BCNF)

A relation in which every Determinant is a Candidate Key.

Determinant:

The Attributes or combination of Attributes that uniquely identifies a row in a relation.

<OR>

The Attribute on the left hand side of the arrow in a Functional Dependency.

Fourth Normal Form (4NF)

A relation in BCNF that contains no Multivalued Dependency.

Multivalued Dependency:

Multivalued dependency occurs when two attributes in a table are independent of each other but, both depend on a third attribute. A multivalued dependency

consists of at least two attributes that are dependent on a third attribute that's why it always requires at least three attributes.

JOIN

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

A Join can be broadly divided into two types:

- Inner Join
- Outer Join
- Inner Join

The inner join can be further divided into the following types:

- Equi Join
- Natural Join

The outer join can be further divided into three types:

- Left-Outer Join
- Right-Outer Join
- Full-Outer Join

Relational Algebra

Relational algebra is a procedural query language that works on relational model. The purpose of a query language is to retrieve data from database or perform various operations such as insert, update, delete on the data.

Projection Operator:

Projection Operator (π) is a unary operator in relational algebra that performs a projection operation. It displays the columns of a relation or table based on the specified attributes.

Select Operation:

A CRITIQUE ON FINAL TERM TOPICS

This operation is used to select rows from a table (relation) that specifies a user defined condition to select rows of user's choice.

SQL QUERY

INNER JOIN:

SELECT column_name(s) FROM table1

INNER JOIN table2 ON table1.column_name = table2.column_name;

LEFT JOIN: SELECT column_name(s) FROM table1

LEFT JOIN table2

ON table1.column_name = table2.column_name;

RIGHT JOIN:

SELECT column_name(s) FROM table1

RIGHT JOIN table2 ON table1.column_name = table2.column_name;

SELF JOIN:

SELECT column_name(s) FROM table1 T1, table1 T2

WHERE condition;

OUTER JOIN:

SELECT column_name(s) FROM table1

FULL OUTER JOIN table2

ON table1.column_name = table2.column_name WHERE condition;

HAVING:

SELECT column_name(s) FROM table_name

WHERE condition GROUP BY column_name(s)

HAVING condition ORDER BY column_name(s);

ANY:

A CRITIQUE ON FINAL TERM TOPICS

SELECT column_name(s) FROM table_name

WHERE column_name operator ANY

(SELECT column_name FROM table_name WHERE condition);

CREATE SEQUENCE:

CREATE SEQUENCE [schema.]sequence_name

[AS datatype]

[START WITH value]

[INCREMENT BY value]

[MINVALUE value | NO MINVALUE]

[MAXVALUE value | NO MAXVALUE]

[CYCLE | NO CYCLE]

[CACHE value | NO CACHE];

CREATE VIEW:

CREATE VIEW view_name AS SELECT column1, column2, ...

FROM table_name

WHERE condition;

UPDATE VIEW:

Update view_name

Set column_Name

Where condition;

INSERT INTO VIEW:

INSERT INTO view_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

DELETE VIEW:

DELETE FROM view_name

WHERE conditions;

DROP VIEW: DROP VIEW view_name;

CREATE USERS:

CREATE USERS user_name indentified by password;

CHANGE USERS:

ALTER USERS user_name indentified by password;

CREATE INDEX:

CREATE INDEX index_name

ON table_name (column1, column2);

GRANT:

- GRANT privilege_name

ON object_name

TO {user_name |PUBLIC |role_name}

[WITH GRANT OPTION];

- GRANT CREATE TABLE TO testing;

REVOKE:

- REVOKE privilege_name

ON object_name

FROM {user_name |PUBLIC |role_name}

- REVOKE CREATE TABLE FROM testing;

ROLE:

- CREATE ROLE role_name

[IDENTIFIED BY password];

A CRITIQUE ON FINAL TERM TOPICS

- CREATE ROLE testing

[IDENTIFIED BY pwd];

- DROP ROLE role_name;
- DROP ROLE testing;