GALGOTIAS UNIVERSITY

School of Computing Science and Engineering

Program: BCA

Course Code:BCAC2102

Course Name: Database Management System

Lecture-14

Topic- Views

Faculty:-Dr. Satyajee Srivastava

Lecture-13(RECAP)

Topic- Queries

Objective :

Understand Queries and How To write Queries?

Lecture-13

Database Basics:

Data item:

The data item is also called as field in data processing and is the smallest unit of data that has meaning to its users.

Eg: "e101","sumit"

Entities and attributes:

An entity is a thing or object in the real world that is distinguishable from all other objects

Eg:

Bank,employee,student

Attributes are properties are properties of an entity.

Eg:

Empcode,ename,rolno,name

Lecture-13

Database language :

1) Data definition language(DDL) :

DDL is used to define database objects .The conceptual schema is specified by a set of definitions expressed by this language. It also give some details about how to implement this schema in the physical devices used to store the data. This definition includes all the entity sets and their associated attributes and their relation ships. The result of DDL statements will be a set of tables that are stored in special file called data dictionary.

Lecture-13

2) Data manipulation language(DML) :

A DML is a language that enables users to access or manipulate data stored in the database. Data manipulation involves retrieval of data from the database, insertion of new data into the database and deletion of data or modification of existing data.

There are basically two types of DML:

- **procedural**: Which requires a user to specify what data is needed and how to get it.
- **non-rocedural:** which requires a user to specify what data is needed with out specifying how to get it.

3) Data control language(DCL):

This language enables user to grant authorization and canceling authorization of database objects.

Lecture-14

Topic- Views

Objective :

Understand Views and How To write Views ?

Lecture-14

Database Objects

Object	Description
Table	Basic unit of storage; composed of rows
View	Logically represents subsets of data from one or more tables
Sequence	Generates numeric values
Index	Improves the performance of some queries
Synonym	Gives alternative names to objects

Lecture-14

VIEW

View is a logical table. It is a physical object which stores data logically. View just refers to data that is tored in base tables.

A view is a logical entity. It is a SQL statement stored in the database in the system tablespace.

Data for a view is built in a table created by the database engine in the TEMP tablespace.

Lecture-14

INDEX

Indexes are pointres that maps to the physical address of data. So by using indexes data manipulation becomes faster.

An index is a performance-tuning method of allowing faster retrieval of records. An index creates an entry for each value that appears in the indexed columns.

Lecture-14

Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.

Lecture-14

Creating , deleting and updating Views

Lecture-14

Sample Tables

StudentDetails

S_ID	NAME	ADDRESS
1	Harsh	Kolkata
2	Ashish	Durgapur
3	Pratik	Delhi
4	Dhanraj	Bihar
5	Ram	Rajasthan

Lecture-14

Sample Tables

StudentMarks

ID	NAME	MARKS	AGE
1	Harsh	90	19
2	Suresh	50	20
3	Pratik	80	19
4	Dhanraj	95	21
5	Ram	85	18

Lecture-14

We can create View using **CREATE VIEW** statement. A View can be created from a single table or multiple tables.

Syntax:

```
CREATE VIEW view_name AS

SELECT column1, column2.....

FROM table_name

WHERE condition;
```

view_name: Name for the View

table_name: Name of the table

condition: Condition to select rows

Lecture-14

Examples:

- **Creating View from a single table:**
  - In this example we will create a View named DetailsView from the table StudentDetails.

    Query:

```
CREATE VIEW DetailsView AS
SELECT NAME, ADDRESS
FROM StudentDetails
WHERE S_ID < 5;
```

    To see the data in the View, we can query the view in the same manner as we query a table.

Lecture-14

```
SELECT * FROM DetailsView;
```

Output:

NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar

Lecture-14

- In this example, we will create a view named StudentNames from the table StudentDetails.

Query:

```
CREATE VIEW StudentNames AS
SELECT S_ID, NAME
FROM StudentDetails
ORDER BY NAME;
```

If we now query the view as,

```
SELECT * FROM StudentNames;
```

Lecture-14

Output:-

S_ID	NAMES
2	Ashish
4	Dhanraj
1	Harsh
3	Pratik
5	Ram

- **Creating View from multiple tables**: In this example we will create a View named MarksView from two tables StudentDetails and StudentMarks. To create a View from multiple tables we can simply include multiple tables in the SELECT statement. Query:

```
CREATE VIEW MarksView AS
SELECT StudentDetails.NAME, StudentDetails.ADDRESS, StudentMarks.MARKS
FROM StudentDetails, StudentMarks
WHERE StudentDetails.NAME = StudentMarks.NAME;
```

To display data of View MarksView:

```
SELECT * FROM MarksView;
```

Output:

NAME	ADDRESS	MARKS
Harsh	Kolkata	90
Pratik	Delhi	80
Dhanraj	Bihar	95
Ram	Rajasthan	85

Lecture-14

DELETING VIEWS

We have learned about creating a View, but what if a created View is not needed any more? Obviously we will want to delete it. SQL allows us to delete an existing View. We can delete or drop a View using the DROP statement.

Syntax:

```
DROP VIEW view_name;
```

view_name: Name of the View which we want to delete.

For example, if we want to delete the View **MarksView**, we can do this as:

```
DROP VIEW MarksView;
```

Lecture-14

UPDATING VIEWS

There are certain conditions needed to be satisfied to update a view. If any one of these conditions is **not** met, then we will not be allowed to update the view.

1. The SELECT statement which is used to create the view should not include GROUP BY clause or ORDER BY clause.
2. The SELECT statement should not have the DISTINCT keyword.
3. The View should have all NOT NULL values.
4. The view should not be created using nested queries or complex queries.

The view should be created from a single table. If the view is created using multiple tables then we will not be allowed to update the view

Lecture-14

UPDATING VIEWS

There are certain conditions needed to be satisfied to update a view. If any one of these conditions is **not** met, then we will not be allowed to update the view.

1. The SELECT statement which is used to create the view should not include GROUP BY clause or ORDER BY clause.
2. The SELECT statement should not have the DISTINCT keyword.
3. The View should have all NOT NULL values.
4. The view should not be created using nested queries or complex queries.

The view should be created from a single table. If the view is created using multiple tables then we will not be allowed to update the view

Lecture-14

- We can use the **CREATE OR REPLACE VIEW** statement to add or remove fields from a view.

  Syntax:
- CREATE OR REPLACE VIEW view_name AS
- SELECT column1,coulmn2,..
- FROM table_name
- WHERE condition;

For example, if we want to update the view **MarksView** and add the field AGE to this View from **StudentMarks** Table, we can do this as:

Lecture-14

```
CREATE OR REPLACE VIEW MarksView AS
SELECT StudentDetails.NAME, StudentDetails.ADDRESS, StudentMarks.MARKS,
StudentMarks.AGE
FROM StudentDetails, StudentMarks
WHERE StudentDetails.NAME = StudentMarks.NAME;
```

If we fetch all the data from MarksView now as:

```
SELECT * FROM MarksView;
```

Output:

NAME	ADDRESS	MARKS	AGE
Harsh	Kolkata	90	19
Pratik	Delhi	80	19
Dhanraj	Bihar	95	21
Ram	Rajasthan	85	18

Lecture-14

- **Inserting a row in a view:**

We can insert a row in a View in a same way as we do in a table. We can use the INSERT INTO statement of SQL to insert a row in a View.**Syntax**:

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

- **view_name**: Name of the View

Example:

In the below example we will insert a new row in the View DetailsView which we have created above in the example of "creating views from a single table".

```
INSERT INTO DetailsView(NAME, ADDRESS)
VALUES("Suresh","Gurgaon");
```

If we fetch all the data from DetailsView now as,

```
SELECT * FROM DetailsView;
```

Lecture-14

Output:

NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar
Suresh	Gurgaon

Lecture-14

- **Deleting a row from a View:**

  Deleting rows from a view is also as simple as deleting rows from a table. We can use the DELETE statement of SQL to delete rows from a view. Also deleting a row from a view first delete the row from the actual table and the change is then reflected in the view.**Syntax**:

- DELETE FROM view_name
- WHERE condition;

- **view_name**:Name of view from where we want to delete rows
- **condition**: Condition to select rows

Lecture-14

Example:

In this example we will delete the last row from the view DetailsView which we just added in the above example of inserting rows.

```
DELETE FROM DetailsView
WHERE NAME="Suresh";
```

If we fetch all the data from DetailsView now as,

```
SELECT * FROM DetailsView;
```

Output:

NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar

Lecture-14

Advantages of Views

To restrict data access

To make complex queries easy

To provide data independence

To present different views of the same data

Lecture-14

(Assignment)

- Creating a simple view
- Creating a complex view
- Creating a view with a check constraint
- Attempting to modify data in the view
- Removing views

Thank You