Database Management System (DBMS)

Lecture-6

Dharmendra Kumar

August 30, 2020

Data Control Language

Data Control Language(DCL) is used to control privileges in Database.
To perform any operation in the database, such as for creating tables or views, a user needs privileges.
In DCL, we have following two commands:
GRANT: It is used to provide any user access privileges or other priviliges for the database.
REVOKE: It is used to take back permissions from any user.

Transaction Control Language

Transaction Control Language(TCL) commands are used to manage transactions in the database. These are used to manage the changes made to the data in a table by DML statements. It also allows statements to be grouped together into logical transactions.

In TCL, we have following three command-

COMMIT Command: COMMIT command is used to permanently save any transaction into the database.

When we use any DML command like INSERT, UPDATE or DELETE, the changes made by these commands are not permanent, until the current session is closed. The changes made by these commands can be rolled back. To avoid that, we use the COMMIT command to mark the changes as permanent.

Following is commit command's syntax,

COMMIT;

DCL and TCL

ROLLBACK command

This command restores the database to last commited state. It is also used with SAVEPOINT command to jump to a savepoint in an ongoing transaction. If we have used the UPDATE command to make some changes into the database, and realise that those changes were not required, then we can use the ROLLBACK command to rollback those changes, if they were not commited using the COMMIT command.

Following is rollback command's syntax,

ROLLBACK TO savepoint_name;

SAVEPOINT command

SAVEPOINT command is used to temporarily save a transaction so that you can rollback to that point whenever required.

Following is savepoint command's syntax,

SAVEPOINT savepoint_name;

Example

```
INSERT INTO class VALUES(5, 'Rahul');
COMMIT;
UPDATE class SET name = 'Abhijit' WHERE id = '5';
SAVEPOINT A;
INSERT INTO class VALUES(6, 'Chris');
SAVEPOINT B;
INSERT INTO class VALUES(7, 'Bravo');
SAVEPOINT C;
SELECT * FROM class;
```

Database Users and Administrators

People who work with a database can be categorized as database users or database administrators.

Database Users

There are four different types of database-system users, differentiated by the way they expect to interact with the system. Different types of user interfaces have been designed for the different types of users.

Database Users(cont.)

Naive users/Parametric users

Naive users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. For example, a bank teller who needs to transfer $50 from account A to account B invokes a program called transfer.

Application programmers

Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces. Rapid application development (RAD) tools are tools that enable an application programmer to construct forms and reports without writing a program.

Database Users(cont.)

Sophisticated users

Sophisticated users interact with the system without writing programs. Instead, they form their requests in a database query language. They submit each such query to a query processor, whose function is to break down DML statements into instructions that the storage manager understands. Analysts who submit queries to explore data in the database fall in this category. They use some of the tools like Online Analytical Processing(OLAP), Data mining.

Database Users(cont.)

Specialized users

Specialized users are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. Among these applications are computer-aided design systems, knowledge base and expert systems, systems that store data with complex data types (for example, graphics data and audio data), and environment-modeling systems.

Database Administrator

A person who has such central control of both the data and the programs that access those data over the system is called a database administrator (DBA). The functions of a DBA are the followings:-

- **Schema definition:** The DBA creates the original database schema by executing a set of data definition statements in the DDL.
- **Storage structure and access-method definition:** DBA decides what structure to be used to store the data and what method to be used to access that.

Database Administrator(cont.)

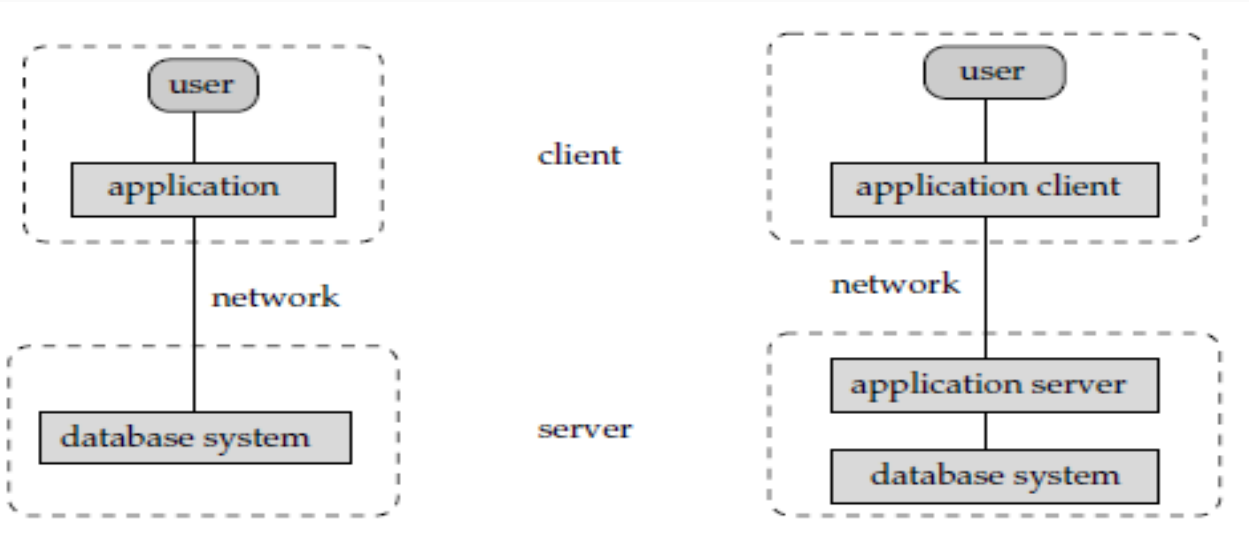
- **Schema and physical-organization modification:** The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.
- **Granting of authorization for data access:** By granting different types of authorization, the database administrator can regulate which parts of the database various users can access.

Database Administrator(cont.)

- **Routine maintenance:** Examples of the database administrator's routine maintenance activities are:
  - Periodically backing up the database, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.
  - Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
  - Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

Application Architectures

Most users of a database system today are not present at the site of the database system, but connect to it through a network. We can therefore differentiate between client machines, on which remote database users work, and server machines, on which the database system runs. Database applications are usually partitioned into two or three parts, as in Figure .



Application Architectures

In a **two-tier architecture,** the application is partitioned into a component that resides at the client machine, which invokes database system functionality at the server machine through query language statements. Application program interface standards like ODBC and JDBC are used for interaction between the client and the server.

In contrast, in a **three-tier architecture,** the client machine acts as merely a front end and does not contain any direct database calls. Instead, the client end communicates with an application server, usually through a forms interface. The application server in turn communicates with a database system to access data. The business logic of the application, which says what actions to carry out under what conditions, is embedded in the application server, instead of being distributed across multiple clients.

Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.