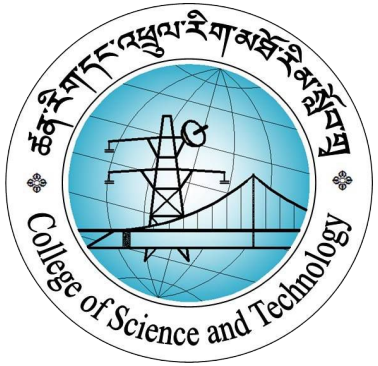


Royal University of Bhutan

DBS101 Database Systems Fundamentals

Lesson 7

Learning Outcomes

1. Implement indexes using SQL statements.
2. Understand SQL authorisation concepts.
3. Access SQL from a programming language.

Index Definition in SQL

An index on an attribute of a relation is a data structure that allows the database system to find those tuples in the relation that have a specified value for that attribute efficiently, without scanning through all the tuples of the relation.

Index Definition in SQL

Creating an index:

```
create index <index-name> on <relation-name> (<attribute-list>);
```

Example:

```
create index deptindex on instructor (dept name);
```

When a user submits an SQL query that can benefit from using an index, the SQL query processor automatically uses the index.

Index Definition in SQL

To declare that the search key is a candidate key:

```
create unique index <index-name> on <relation-name>(<attribute-list>);
```

Example:

```
create  unique index deptindex on instructor (dept name);
```

To delete an index

```
drop index index_name
```

Authorisation

Authorisations on data include:

- Authorisation to read data.
- Authorisation to insert new data.
- Authorisation to update data.
- Authorisation to delete data.

Each of these types of authorizations is called a **privilege.**

Authorisation

The SQL standard includes the privileges select, insert, update, and delete.

The SQL data-definition language includes commands to grant and revoke privileges.

The grant statement is used to confer authorization. Basic form:

grant<privilege list>

on<relation name>

to<user/role>

Authorisation

Select: The select authorization on a relation is required to read tuples in the relation.

Update: The update authorization on a relation allows a user to update any tuple in the relation.

- the list of attributes on which update authorization is to be granted optionally appears in parentheses immediately after the update keyword.

Authorisation

Insert: The insert authorization on a relation allows a user to insert tuples into the relation.

- may also specify a list of attributes; any inserts to the relation must specify only these attributes, and the system either gives each of the remaining attributes default values or sets them to null.

Delete: The delete authorization on a relation allows a user to delete tuples from a relation.

Execute: The execute privilege can be granted on a function or procedure, enabling a user to execute the function or procedure.

Authorisation

Note:

By default, a user/role that is granted a privilege is not authorized to grant that privilege to another user/role.

The user name public refers to all current and future users of the system. Thus, privileges granted to public are implicitly granted to all current and future users.

Authorisation

Revoke Authorization:

Basic form:

```
revoke<privilege list>
on<relation name>
to<user/role>
```

Authorisation on Roles

A set of roles is created in the database.
Authorizations can be granted to roles, in exactly the same fashion as they are granted to individual users.

Roles can be created in SQL as follows:

```
create role instructor;
```

the privileges of a user or a role consist of:

- All privileges directly granted to the user/role.
- All privileges granted to roles that have been granted to the user/role.

Note:there can be a chain of roles.

Authorisation on Views

A user who creates a view does not necessarily receive all privileges on that view.

For example, a user who creates a view cannot be given update authorization on a view without having update authorization on the relations used to define the view.

Authorisation on Schemas

The SQL standard specifies a primitive authorization mechanism for the database schema: Only the owner of the schema can carry out any modification to the schema, such as creating or deleting relations, adding or dropping attributes of relations, and adding or dropping indices.

SQL includes a references privilege that permits a user to declare foreign keys when creating relations.

Example:

```
grant references (dept name) on department to Mariano;
```

Transfer of privileges

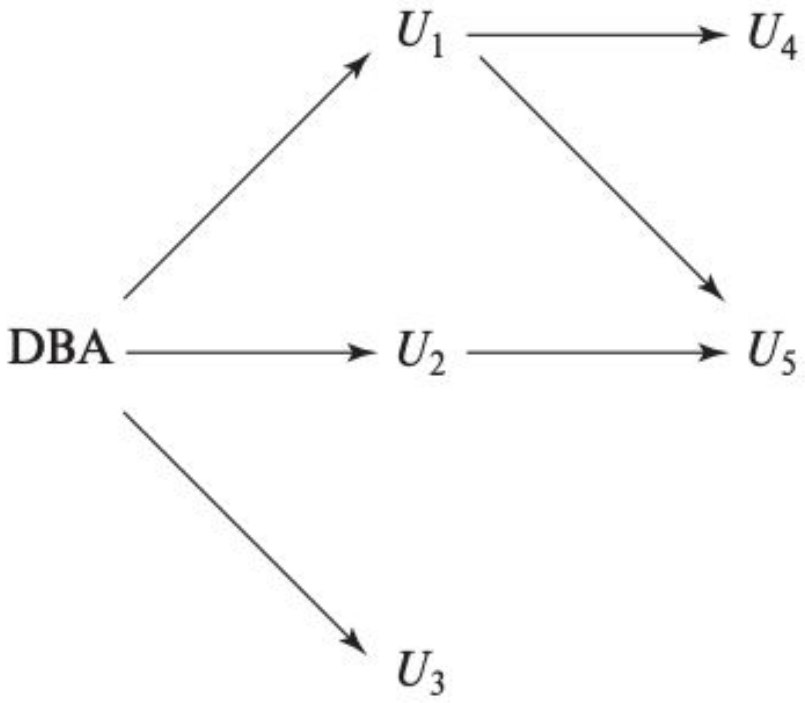
A user who has been granted some form of authorization may be allowed to pass on this authorization to other users.

By default, a user/role that is granted a privilege is not authorized to grant that privilege to another user/role.

For example, if we wish to allow Amit the select privilege on department and allow Amit to grant this privilege to others, we write:

grant select on department to Amit with grant option;

Authorisation grant graph



Revoking of privileges

Revoke command basic form:

```
revoke select on department from Amit;
```

To prevent cascading revocation:

```
revoke select on department from Amit, Satoshi restrict;
```

Row level Authorisation

Suppose, for example, that we wish to allow a student to see her or his own data in the takes relation but not those data of other users.

We can enforce such a restriction using row-level authorization, if the database supports.

Accessing SQL from a Programming Language

Reasons:

1. SQL lacks the expressiveness of general-purpose languages like C, Java, or Python, limiting its ability to express certain complex queries.
2. Nondeclarative actions such as generating reports or interacting with users cannot be performed directly within SQL, necessitating integration with general-purpose programming languages for comprehensive application development.

Accessing SQL from a Programming Language

There are two approaches to accessing SQL from a general-purpose programming language:

- Dynamic SQL
- Embedded SQL

Accessing SQL from a Programming Language

Dynamic SQL

A general-purpose program can connect to and communicate with a database server using a collection of functions (for procedural languages) or methods (for object-oriented languages).

Dynamic SQL enables programs to construct SQL queries as strings during runtime, submit them to the database, and retrieve results iteratively into program variables.

Two common standards for connecting to SQL databases are JDBC for Java and ODBC.

Accessing SQL from a Programming Language

Embedded SQL

Embedded SQL provides a means by which a program can interact with a database server.

However, under embedded SQL, the SQL statements are identified at compile time using a preprocessor, which translates requests expressed in embedded SQL into function calls. At runtime, these function calls connect to the database using an API that provides dynamic SQL facilities but may be specific to the database that is being

Accessing SQL from a Programming Language

JDBC

The JDBC standard defines an application program interface (API) that Java programs can use to connect to database servers.

Accessing SQL from a Programming Language

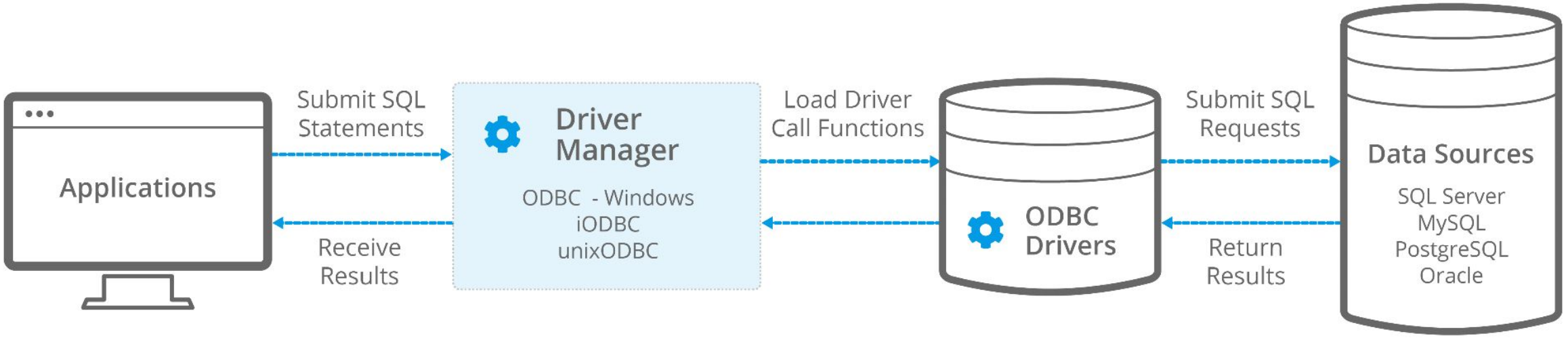
JDBC

The JDBC standard defines an application program interface (API) that Java programs can use to connect to database servers.

ODBC

The Open Database Connectivity (ODBC) standard defines an API that applications can use to open a connection with a database, send queries and updates, and get back results.

Accessing SQL from a Programming Language



Next Week

4.9 Functions and Procedures
4.10 Triggers
4.11 Recursive Queries
4.12 Advanced Aggregation Features —------> Flipped Class

OVER THE WEEKEND

Install pgadmin4