Intermediate SQL

Dr. Shrutilipi Bhattacharjee, Assistant Professor, Dept. of IT, NIT Karnataka, India

Built-in Data Types in SQL

- **date:** Dates, containing a (4 digit) year, month and date
  - Example: **date** '2005-7-27'
- **time:** Time of day, in hours, minutes and seconds.
  - Example: **time** '09:00:30' **time** '09:00:30.75'
- **timestamp:** Date plus time of day
  - Example: **timestamp** '2005-7-27 09:00:30.75'
- **interval:** Period of time
  - Example: interval '1' day
  - Subtracting a date/time/timestamp value from another gives an interval value
  - Interval values can be added to date/time/timestamp values

Index Creation

- Many queries reference only a small proportion of the records in a table
- It is inefficient for the system to read every record to find a record with particular value
- 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
- We create an index with the **create index** command

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

Index Creation Example

create table *student*
(*ID* **varchar** (5),
name **varchar** (20) **not null**,
dept_name **varchar** (20),
tot_cred **numeric** (3,0) **default** 0,
primary key (*ID*))
create index *studentID_index* **on** *student*(*ID*)

- The query can be executed by using the index to find the required record, without looking at all records of *student*
- Indices are data structures used to speed up access to records with specified values for index attributes

select *
from *student*
where *ID* = '12345'

User-Defined Types

- **create type** construct in SQL creates user-defined type

  create type *Dollars* **as numeric (12, 2) final**

- Example:

  create table *department*
  (*dept_name* **varchar** (20),
  building **varchar** (15),
  budget Dollars);

Domains

- **create domain** construct in SQL-92 creates user-defined domain types

  create domain *person_name* **char**(20) **not null**

- Types and domains are similar
- Domains can have constraints, such as **not null**, specified on them

- Example:

  create domain *degree_level* **varchar**(10)
  constraint *degree_level_test*
  check (**value in** ('Bachelors', 'Masters', 'Doctorate'));

Large-Object Types

- Large objects (photos, videos, CAD files, etc.) are stored as a *large object*:
  - **blob**: binary large object: Object is a large collection of uninterpreted binary data (whose interpretation is left to an application outside of the database system)
  - **clob**: character large object: Object is a large collection of character data
- When a query returns a large object, a pointer is returned rather than the large object itself

Authorization

- We may assign a user several forms of authorizations on parts of the database
  - **Read:** Allows reading, but not modification of data
  - **Insert:** Allows insertion of new data, but not modification of existing data
  - **Update:** Allows modification, but not deletion of data
  - **Delete:** Allows deletion of data

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

- We may authorize the user all, none, or a combination of these types of privileges on specified parts of a database, such as a relation or a view

Authorization

- Forms of authorization to modify the database schema
  - **Index:** Allows creation and deletion of indices
  - **Resources:** Allows creation of new relations
  - **Alteration:** Allows addition or deletion of attributes in a relation
  - **Drop:** Allows deletion of relations

Authorization Specification in SQL

- The **grant** statement is used to confer authorization

  grant <privilege list> **on** <relation or view > **to** <user list>

- <user list> is:
  - a user-id
  - **public**, which allows all valid users the privilege granted
  - A role (more on this later)

- Example:

  grant select on *department* **to** Amit, Satoshi

- Granting a privilege on a view does not imply granting any privileges on the underlying relations

- The grantor of the privilege must already hold the privilege on the specified item (or be the database administrator)

Privileges in SQL

- **select**: Allows read access to relation, or the ability to query using the view

- Example: Grant users U_1, U_2, and U_3 **select** authorization on the *instructor* relation:

  grant select on *instructor* **to** U_1, U_2, U_3

- **insert**: The ability to insert tuples
- **update**: The ability to update using the SQL update statement
- **delete**: The ability to delete tuples
- **all privileges**: Used as a short form for all the allowable privileges

Revoking Authorization in SQL

- The **revoke** statement is used to revoke authorization

  revoke <privilege list> **on** <relation or view> **from** <user list>

- Example:

  revoke select on *student* **from** U_1, U_2, U_3

- <privilege-list> may be **all** to revoke all privileges the revokee may hold

- If <revokee-list> includes **public,** all users lose the privilege except those granted it explicitly

- If the same privilege was granted twice to the same user by different grantees, the user may retain the privilege after the revocation

- All privileges that depend on the privilege being revoked are also revoked

Roles

- A **role** is a way to distinguish among various users as far as what these users can access/update in the database

- To create a role we use:

  create role <name>

- Example:

  create role instructor

- Once a role is created we can assign “users” to the role using:

  grant <role> **to** <users>

Roles Example

- **create role** instructor;
  - **grant** *instructor* **to** Amit**;**

- Privileges can be granted to roles:

  grant select on *takes* **to** *instructor*;

- Roles can be granted to users, as well as to other roles

  create role *teaching_assistant*

  grant *teaching_assistant* **to** *instructor*;

  - *Instructor* inherits all privileges of *teaching_assistant*

- Chain of roles
  - **create role** *dean*;
  - **grant** *instructor* **to** *dean*;
  - **grant** *dean* **to** Satoshi;

Authorization on Views

create view *geo_instructor* **as**
(**select** *
from *instructor*
where *dept_name* = 'Geology');

- **grant select on** *geo_instructor* **to** *geo_staff*

- Suppose that a *geo_staff* member issues
  select *
  from *geo_instructor*;
- What if
  - *geo_staff* does not have permissions on *instructor?*
  - Creator of view did not have some permissions on *instructor?*

Other Authorization Features

- **references** privilege to create foreign key
  - **grant reference** (*dept_name*) **on** *department* **to** Mariano;
  - Why is this required?

- Transfer of privileges
  - **grant select on** *department* **to** Amit **with grant option**;
  - **revoke select on** *department* **from** Amit, Satoshi **cascade**;
  - **revoke select on** *department* **from** Amit, Satoshi **restrict**;
  - And more!

Next Lecture

Advanced SQL

Thank you for your attention...

Any question?

Contact:
Department of Information Technology, NITK Surathkal, India
6th Floor, Room: 13
Phone: +91-9477678768
E-mail: shrutilipi@nitk.edu.in