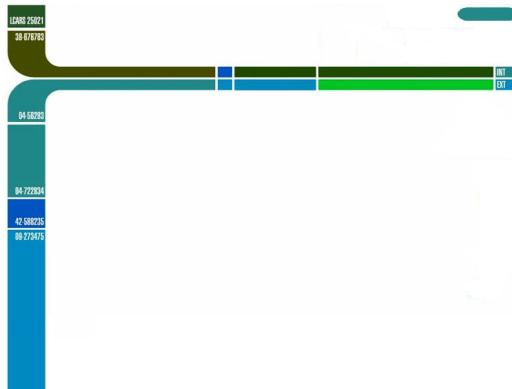


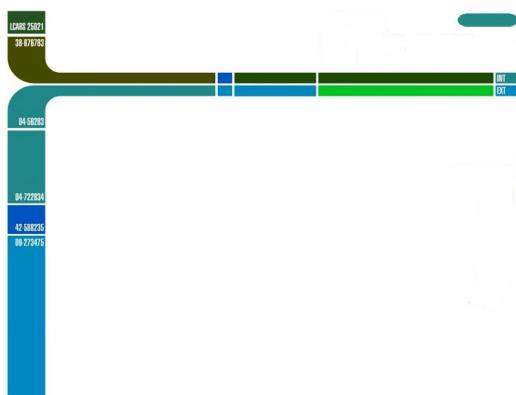
19.1
SC
C
2
0
1

Unit 19: Advanced SQL and
Access Control
SCC201 Databases
Based of slides from
John Mariani



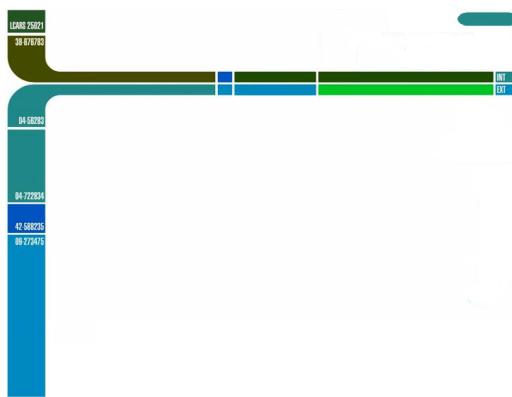
19.2

1 In this Unit . . .



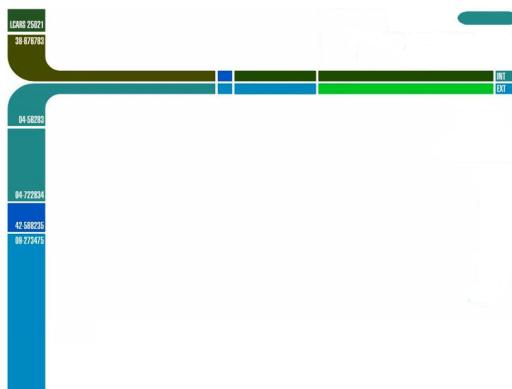
19.3

2 ADVANCED SQL : MORE ON SCHEMAS AND VIEWS



3 Introduction

- Following on from the earlier “Schemas and Views” Unit.
- How to
 - Set up a schema in SQL
 - Define tables in SQL
 - Define views in SQL
- Notice this material focuses on full SQL
 - Some of these features may not be available in certain subset implementations of SQL
(such as in base MySQL or SQLite)



19.5

4 Conceptual Schema Definition in SQL (1)

We will use our example from ER-to- relational mapping:

DEPARTMENT DName HoD NoOfEmps

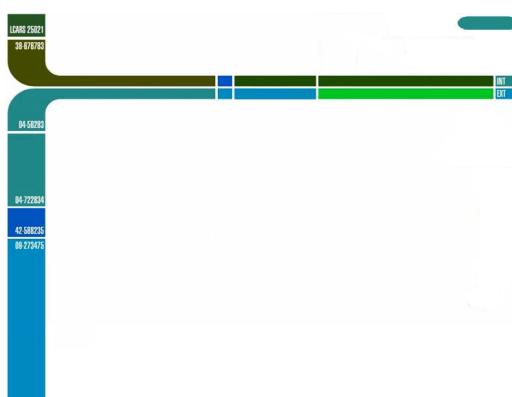
COURSE CName Description DName

STUDENT

FName LName RegNum BDate Address Gender DName

TAKES CName RegNum

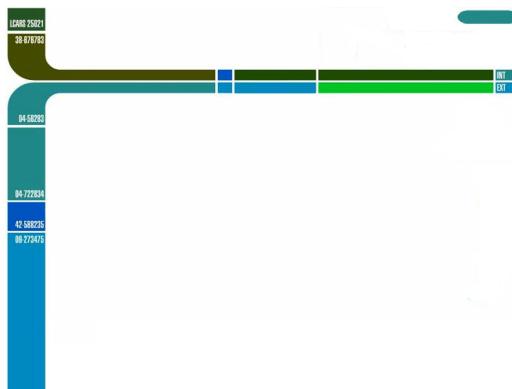
DEPT_LOCATIONS DName DLocation



5 Conceptual Schema Definition in SQL (2)

To create schema:

- CREATE SCHEMA <SCHEMA_NAME> AUTHORISATION <AUTH_IDENTIFIER>
 - In our example:
- CREATE SCHEMA UNIVERSITY AUTHORISATION JDOE



6 Conceptual Schema Definition in SQL (3)

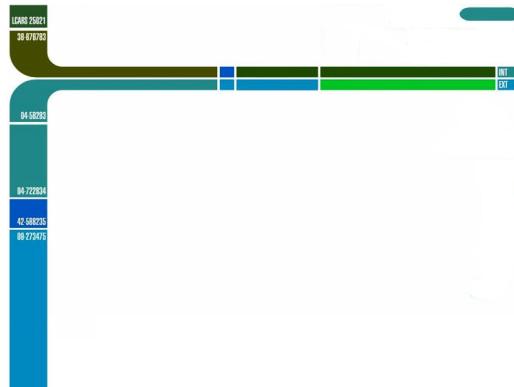
- To create tables for a schema explicitly:

```
CREATE TABLE <SCHEMA_NAME>. <TABLE_NAME>
<TABLE_DEFINITION>
```

- To create tables for a schema implicitly:

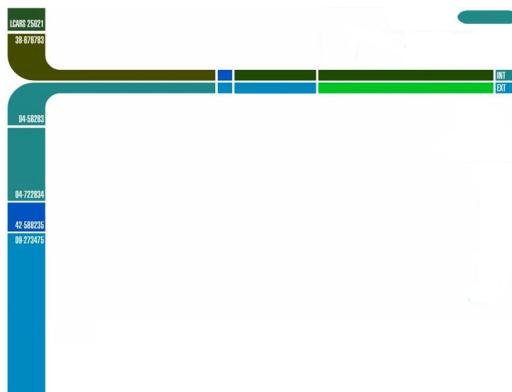
```
CREATE TABLE <TABLE_NAME>
<TABLE_DEFINITION>
```

Schema name specified in environment is used



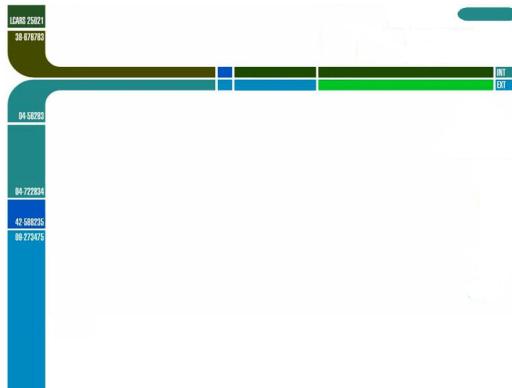
7 Conceptual Schema Definition in SQL (4)

```
CREATE TABLE STUDENT  
(FNAME VARCHAR(20) NOT NULL,  
LNAME VARCHAR(20) NOT NULL,  
REGNUM INT NOT NULL,  
BDATE DATE,  
ADDRESSVARCHAR(30) ,  
GENDER CHAR,  
DNAME VARCHAR(20) NOT NULL,  
PRIMARY KEY(REGNUM),  
FOREIGN KEY(DNAME) REFERENCES DEPARTMENT(NAME)  
);
```



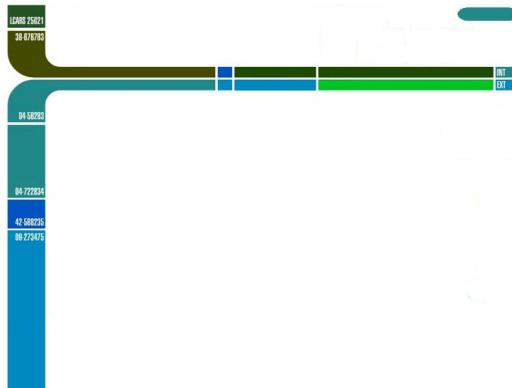
8 Conceptual Schema Definition in SQL (5)

```
CREATE TABLE DEPARTMENT  
(NAME VARCHAR(20) NOT NULL,  
HOD VARCHAR(20) NOT NULL,  
NUMOFEMPS INT NOT NULL DEFAULT 1,  
PRIMARY KEY(NAME)  
);  
CREATE TABLE COURSE  
(NAME VARCHAR(20) NOT NULL,  
DESCRIPTION VARCHAR(40),  
DNAME VARCHAR(20) NOT NULL,  
PRIMARY KEY(NAME),  
FOREIGN KEY(NAME) REFERENCES DEPARTMENT(NAME)  
);
```



9 Conceptual Schema Definition in SQL (6)

```
CREATE TABLE TAKES
(CNAME VARCHAR(20) NOT NULL,
REGNUM INT NOT NULL,
PRIMARY KEY(CNAME, REGNUM),
FOREIGN KEY(CNAME) REFERENCES COURSE(NAME),
FOREIGN KEY(REGNUM) REFERENCES STUDENT(REGNUM)
);
CREATE TABLE DEPT_LOCATIONS
(DNAME VARCHAR(20) NOT NULL,
LOCATION VARCHAR(20) NOT NULL,
PRIMARY KEY(DNAME, LOCATION),
FOREIGN KEY(DNAME) REFERENCES DEPARTMENT(NAME)
);
```



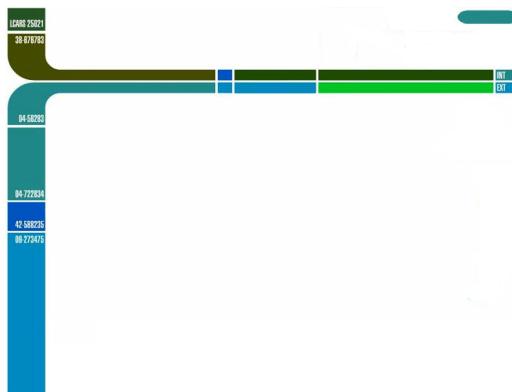
10 Schema Evolution using SQL

- We can use following three commands:

DROP SCHEMA

DROP TABLE

ALTER TABLE



11 Views in SQL (1)

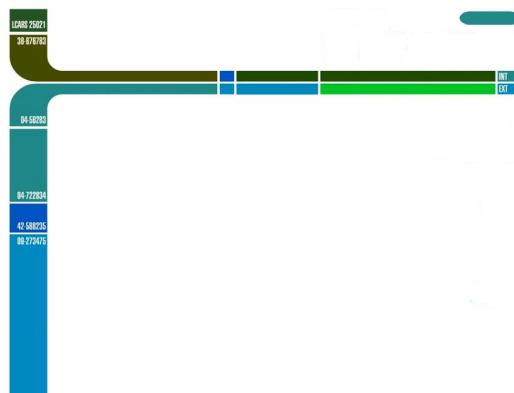
- Views are virtual tables

Do not necessarily exist in physical form

As opposed to base tables whose tuples are actually stored in a database

- If same query frequently executed on database it makes sense to define view based on results of query and use simpler query to retrieve tuples of interest from view

Particularly useful if original query is complex, e.g. involves a number of joins



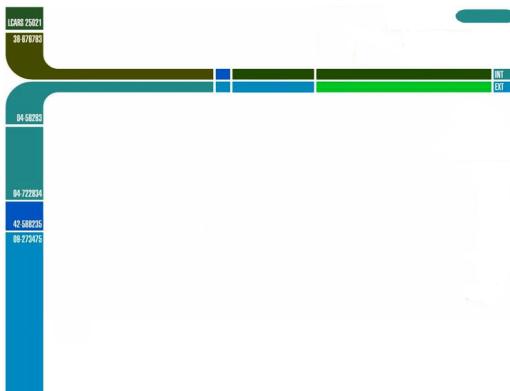
19.13

12 Views in SQL (2)

- Use the command:

```
CREATE VIEW <VIEW_NAME> AS <SQL_QUERY>
```

```
CREATE VIEW PHYSICS_STUDENTS AS SELECT *  
FROM STUDENT  
WHERE DNAME = 'Physics' ;
```



19.14

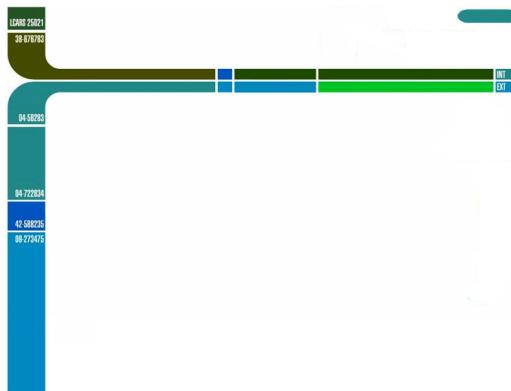
13 Views in SQL (3)

- Notice that virtual relations can be used in same way as base relations in SQL statements

```
SELECT FNAME, LNAME FROM PHYSICS_STUDENTS  
WHERE GENDER = 'M'
```

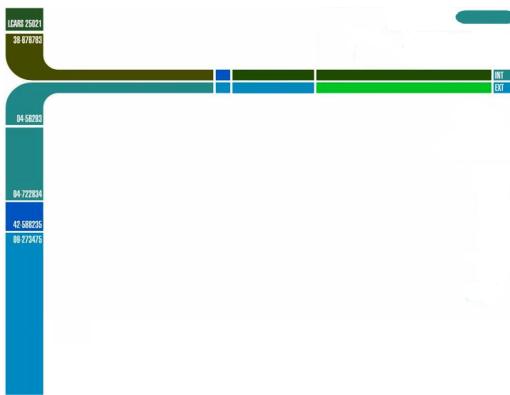
- Views can be dropped by using command:

```
DROP VIEW <VIEW_NAME> e.g.  
DROP VIEW PHYSICS_STUDENTS
```



19.15

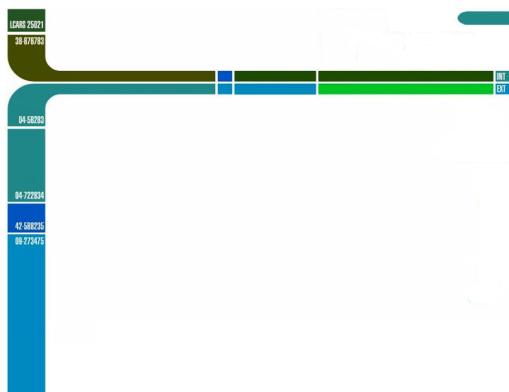
14 ACCESS CONTROL : SECURITY IN SQL



19.16

15 Mandatory Access Control

- Each database object is assigned a certain classification level
 - i.e. top secret, secret, confidential, unclassified
 - The levels form a strict ordering.
 - top secret > secret > confidential > unclassified
- Each subject (users or programs) is given a clearance level.
- To access an object, a subject requires the necessary clearance to read or write a database object.
- See the Bell-LaPadula access control model (1974).
- We will not cover this approach further in this course.

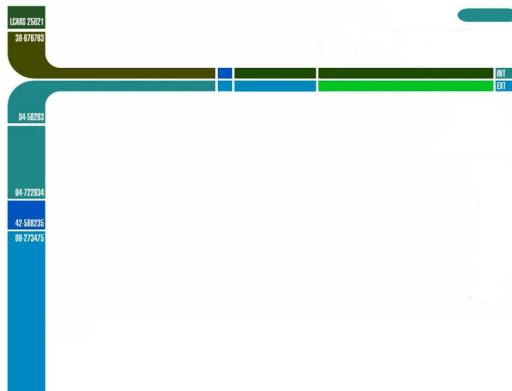


16 Discretionary Access Control

Each user is given appropriate access rights (or privileges) on specific database objects.

Users obtain certain privileges when they create an object and can pass some or all of these privileges to other users at their discretion.

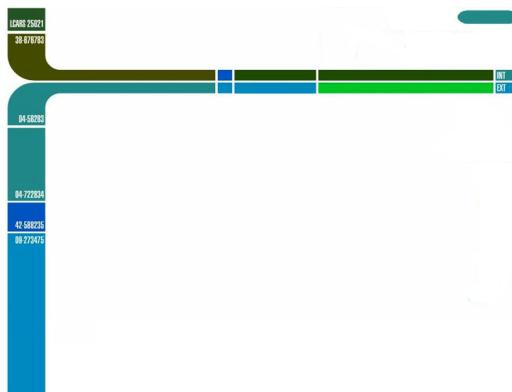
This approach is used in SQL.



17 Authorisation Identifier

- An SQL identifier used to establish the identity of a user.
 - The DBA sets up your username and usually a password.
 - Every SQL statement executed by the DBMS is performed on behalf of a specific user.
 - By the access rights associated with a user, we can determine

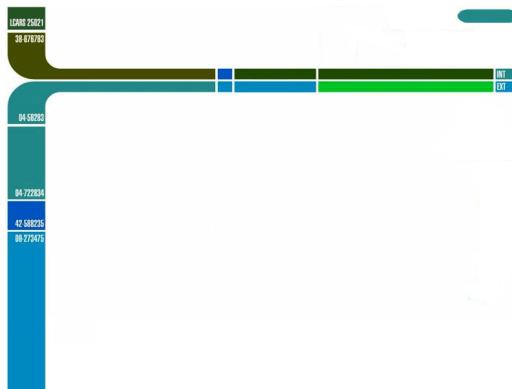
what database objects a user can reference and what operations can be performed by that user.



19.19

18 Ownership

- Each object created in SQL has an owner.
- The owner is identified by the authorisation identifier defined in the AUTHORIZATION clause of the schema to which the object belongs.
- The owner is initially the only person who knows that object exists and subsequently perform operations on that object.



19 Privileges

The ISO standard defines the following privileges, among others.

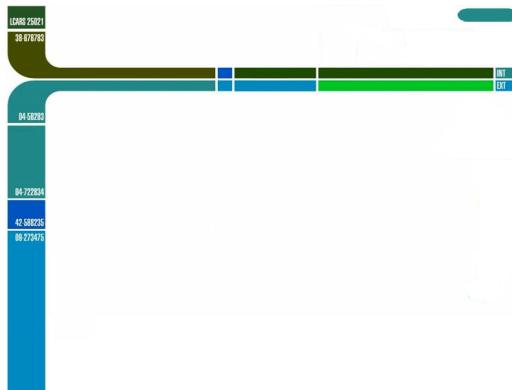
select to retrieve data from a table

insert to insert new rows into a table. Can be restricted to specific columns.

update to modify rows of data in a table. Can be restricted to specific columns.

delete to delete rows of data from a table

references to reference columns of a named table in integrity constraints. Can be restricted to specific columns.



19.21

20 Create Table

- When you create a table, you are the owner and have full privileges.
- Other users have no access, and must be GRANTed permissions by the owner.
- When you create a view, you are the owner of the view. But you may not have full privileges.
- You must have select privilege on the base table, in order to create the view in the first place.



19.22

21 GRANT {PrivilegeList | ALL PRIVILEGES}

```
ON ObjectName  
TO {AuthorizationList | PUBLIC}  
WITH GRANT OPTION
```

The GRANT command

select

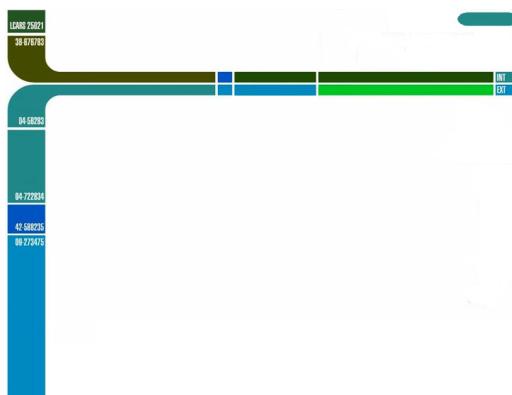
delete

insert [(columnName, [...])]

update [(columnName, [...])]

references [(columnName, [...])]

PrivilegeList

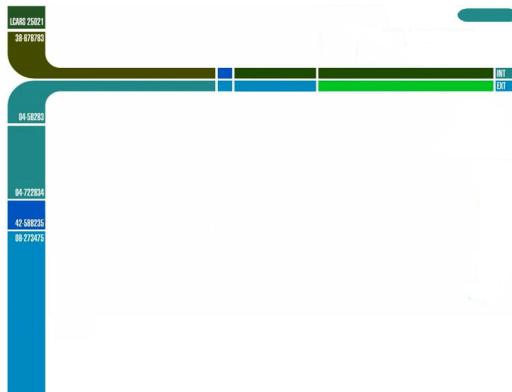


19.23

22 Examples

```
GRANT ALL PRIVILEGES  
ON Staff  
TO Manager  
WITH GRANT OPTION
```

The user Manager can now retrieve rows from the Staff table, and also insert, update and delete.
The Manager can pass these privileges onto other users.



19.24

23 Examples

GRANT SELECT, UPDATE (salary)

ON Staff

TO Personnel, Director

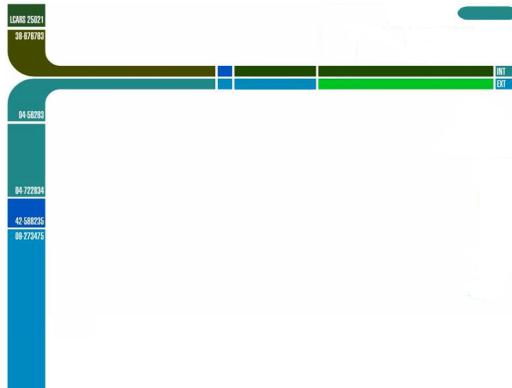
Gives the users Personnel and Director the privileges to select and update the salary column of the Staff table.

GRANT SELECT

ON Branch

TO PUBLIC

Gives all users the privilege
SELECT on the Branch table.

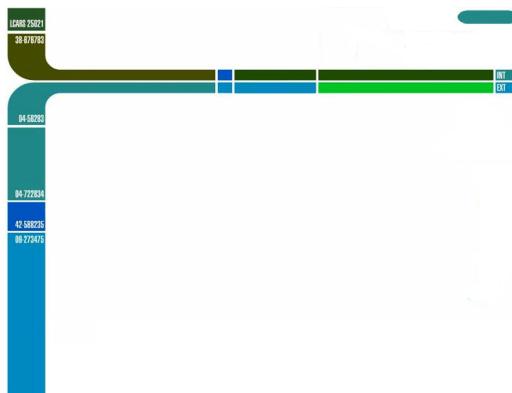


19.25

24 Revoking privileges from users

The REVOKE statement can take away all or some of the privileges previously GRANTED.

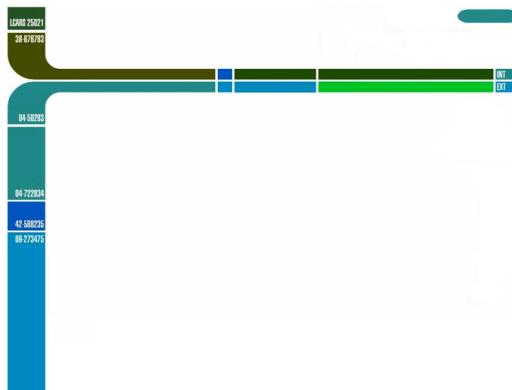
```
REVOKE {PrivilegeList | ALL PRIVILEGES}  
ON ObjectName  
FROM {AuthorizationList | PUBLIC}
```



19.26

25 Examples

```
REVOKE SELECT  
ON Branch  
FROM PUBLIC  
Revoke the SELECT  
privilege on the Branch  
table from all users.  
REVOKE ALL PRIVILEGES  
ON Staff  
FROM Director  
Revoke all privileges  
you have given to  
Director on the Staff  
table.
```



19.27

26 THE END

