Security and integrity

CMT220
Databases & Modelling

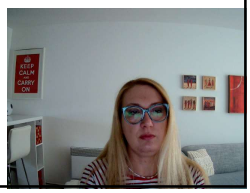
Cardiff School of **Computer Science & Informatics**
http://www.cs.cf.ac.uk

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

1

Lecture

- in this lecture we will consider a range of issues concerning database applications
- database security
  - aspects of security
  - access to databases
  - privileges and views
- database integrity
  - view updating
  - integrity constraints

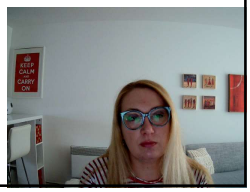


CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

2

Database security

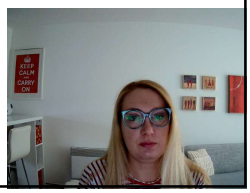
- database security is about controlling access to information
  - some information should be available freely
  - other information should only be available to certain users
- many different aspects of security:
  - legal issues
  - physical security
  - OS/network security
  - security policies and protocols
  - encryption and passwords
  - DBMS security



3

DBMS security support

- DBMS can provide some security
  - each user has an account name and password
  - these are used to identify a user and control their access to information
- DBMS verifies user's password and checks their permissions whenever they try to
  - view data
  - modify data
  - modify the database structure



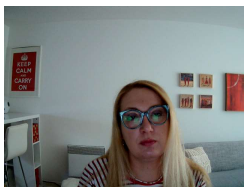
4

Privileges

5

Privileges

- SQL uses different privileges to control access to tables and other database objects
  - SELECT privilege
  - INSERT privilege
  - UPDATE privilege
  - DELETE privilege



- the owner (creator) of a database has all privileges on all objects in the database and can grant privileges to other users
- the owner (creator) of an object has all privileges on that object and can pass them on to others

6

6

Privileges in SQL

```
GRANT <privileges>
ON    <object>
TO    <users>
[WITH GRANT OPTION]
```

- `SELECT <columns>`
- `INSERT <columns>`
- `DELETE`
- `UPDATE <columns>`
- or `ALL`

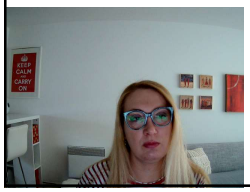
What?

- table or view

- list of users
- or `PUBLIC`

Who?

- allows the specified users to pass their privileges to other users



7

7

Examples

```
GRANT ALL
ON    Employee
TO    Manager
WITH GRANT OPTION
```

- the user `Manager` can do anything to the `Employee` table, and can allow other users to do the same (by using `GRANT` statements)

```
GRANT SELECT, UPDATE(Salary)
ON     Employee
TO     Finance
```

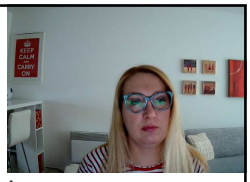
- the user `Finance` can view the entire `Employee` table, and can change values in its `Salary` column, but can**not** change any other values or pass on their privilege to other users

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

8

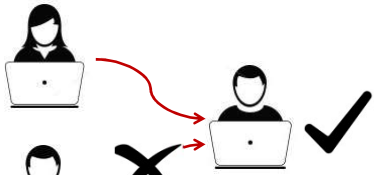
8

Removing privileges



- a previously granted privilege can be revoked using the following statement:

```
REVOKE <privileges>
ON     <object>
FROM   <users>
```

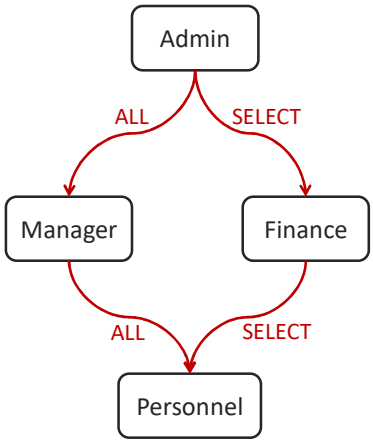


- if a user was granted the same privilege from some other user, then they will still keep it
- all other privileges that depend on the revoked one will be revoked automatically

9

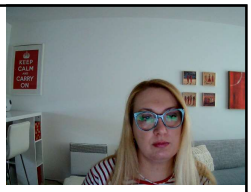
Example

1. Admin grants ALL to Manager WITH GRANT OPTION
2. Admin grants SELECT to Finance WITH GRANT OPTION
3. Manager grants ALL to Personnel
4. Finance grants SELECT to Personnel

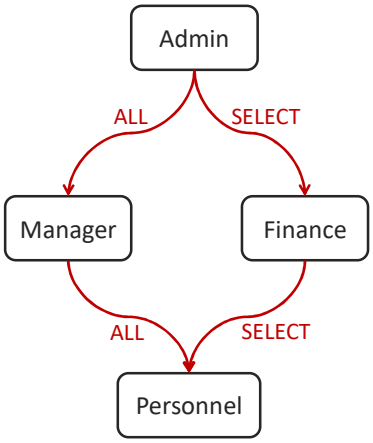


10

Example



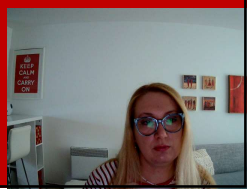
1. Manager revokes ALL from Personnel
   - Personnel still has SELECT privilege from Finance
2. Admin revokes SELECT from Finance
   - Personnel loses SELECT privilege



11

11

Views



12

Views

- privileges work at the level of tables
  - access can be restricted by columns
  - access can**not** be restricted by row!
- views, together with privileges, allow for customised access control
- a view is a table that is derived as the result of a SELECT statement
- SELECT statement can then be used with views in the same way tables can
- UPDATE statement can sometimes be used with views

13

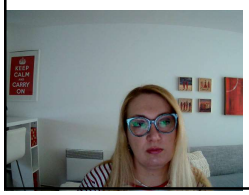
13

Creating views

the name of the newly created view

```
CREATE VIEW <name>
AS          <select statement>
```

a query that returns the rows and the columns of the view



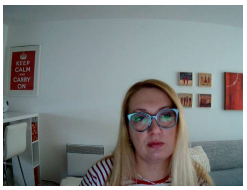
14

14

Example

ID	Name	Phone	Department	Salary
S123	J Evans	x70320	Finance	£30000
S234	L Davies	x70871	Marketing	£35000
S345	R Jenkins	x75600	Marketing	£35000
…	…	…	…	…

- we want each user to be able to view the names and phone numbers only of those employees in their own department
- note: in Oracle, you can refer to the current user using the keyword USER



```
CREATE VIEW OwnDept AS
     SELECT Name, Phone
     FROM   Employee
     WHERE  Department = (SELECT Department
                          FROM   Employee
                          WHERE  name = USER);

GRANT SELECT ON OwnDept TO PUBLIC;
```

15

15

Views and privileges

- views and privileges are combined to control access
  - create a view that contains the information needed
  - grant privileges to that view, rather than the underlying tables
- views are virtual tables
  - their content depends on the underlying tables
  - we can select from views just like a table
  - … but what about update, insert, and delete?

16

16

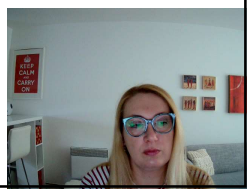
Updating views

- updates to the underlying tables will change the views
- for a view itself to be updatable, the defining query has to satisfy the following conditions:

1. every element in SELECT is a column name
2. should **not** use DISTINCT
3. view should be defined on a single table (no JOIN, UNION, etc.)
4. WHERE should **not** contain nested SELECT statements
5. should **not** use GROUP BY or HAVING

Access control with views and privileges

- to restrict a user's access to a table ...
    1. create a view of that table that shows only the information the user needs to see
    2. grant the user privileges on the view
    3. revoke any privileges the user has on the original table



Example

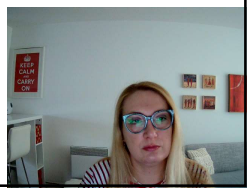
- `Employee(ID, Name, Salary, Department)`
- we want to let the user `John` see `Department` and `Name`, and be able to update `Department` (only)
- creating a view

```
CREATE VIEW JohnsView
AS SELECT Name, Department FROM Employee;
```

- setting the privileges

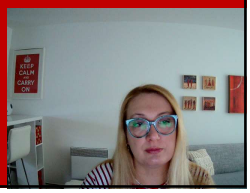
```
GRANT SELECT, UPDATE(Department)
ON JohnsView TO John;

REVOKE ALL ON Employee FROM John;
```



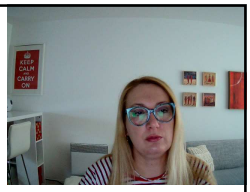
19

Database integrity



20

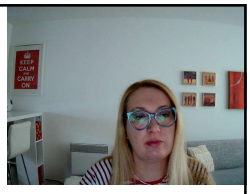
Security vs. integrity



- database **security** makes sure that the user is authorised to access information
- database **integrity** makes sure that (authorised) users use that information correctly
- integrity constraints:
  1. domain constraints apply to data types
  2. attribute constraints apply to columns
  3. relation constraints apply to rows in a single table
  4. database constraints apply between tables

21

Domains and attributes



- domain constraints are data types
- SQL statement: `CREATE DOMAIN` (note: not in Oracle)

```
CREATE DOMAIN Colour VARCHAR(15)
CONSTRAINT checkCol CHECK (VALUE IN ('red','blue',...);
```

- attributes are constrained by their domains

```
CREATE TABLE Dress
(
  size Int,
  col  Colour
);
```


22

Assertions

- assertions provide a way to specify relation and database constraints
- `CREATE ASSERTION <name> CHECK (<condition>);`
  - assertions state a Boolean condition that must always be true
  - no action is permitted that would make the condition false
  - <condition> can refer to one or several tables
  - often use EXISTS or NOT EXISTS
- note: not supported in Oracle!

23

23

Relation constraints

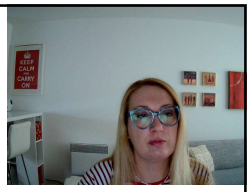
- to create a relation constraint we simply create an assertion that checks the given constraint on a single table
- e.g. in the `Employee` table, no `Bonus` should be more than 15% of the employee's `Salary`

```
CREATE ASSERTION checkSalaryBonus CHECK
(
     NOT EXISTS
     (
          SELECT *
          FROM   Employee
          WHERE  Bonus > 0.15*Salary
     )
);
```

24

24

Database constraints



- database constraints are similar to relation constraints, but they refer to multiple tables
- e.g. given tables `Student(ID, Name, Department)` and `Enrolment(ID, Module)`, make sure no computer science student takes more than 12 modules

```
CREATE ASSERTION CSEnrolment CHECK
(
     NOT EXISTS
     (
            SELECT *
            FROM   Student AS S
            WHERE  S.Department = 'computer science'
            AND    ((SELECT COUNT(*) FROM Enrolment AS E
                     WHERE  S.ID = E.ID) > 12
     )
);
```

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

25

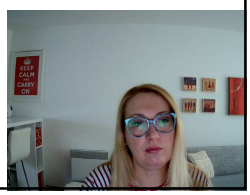
25

Constraints in Oracle

- Oracle does **not** support domains or assertions!
- ... but it does support row–level constraints using `CHECK <constraint>`
- row–level constraints are declared like other constraints:

```
CONSTRAINT <name> CHECK (<condition>);
```

- less general than assertions since the condition refers to a single row of a single table



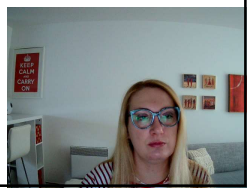
CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

26

Example

- add a check on the `Employee` table to make sure no employee's `Bonus` is more than 15% of their `Salary`

```
ALTER TABLE Employee
ADD CONSTRAINT checkSalaryBonus
CHECK (Bonus < 0.15*Salary);
```



27