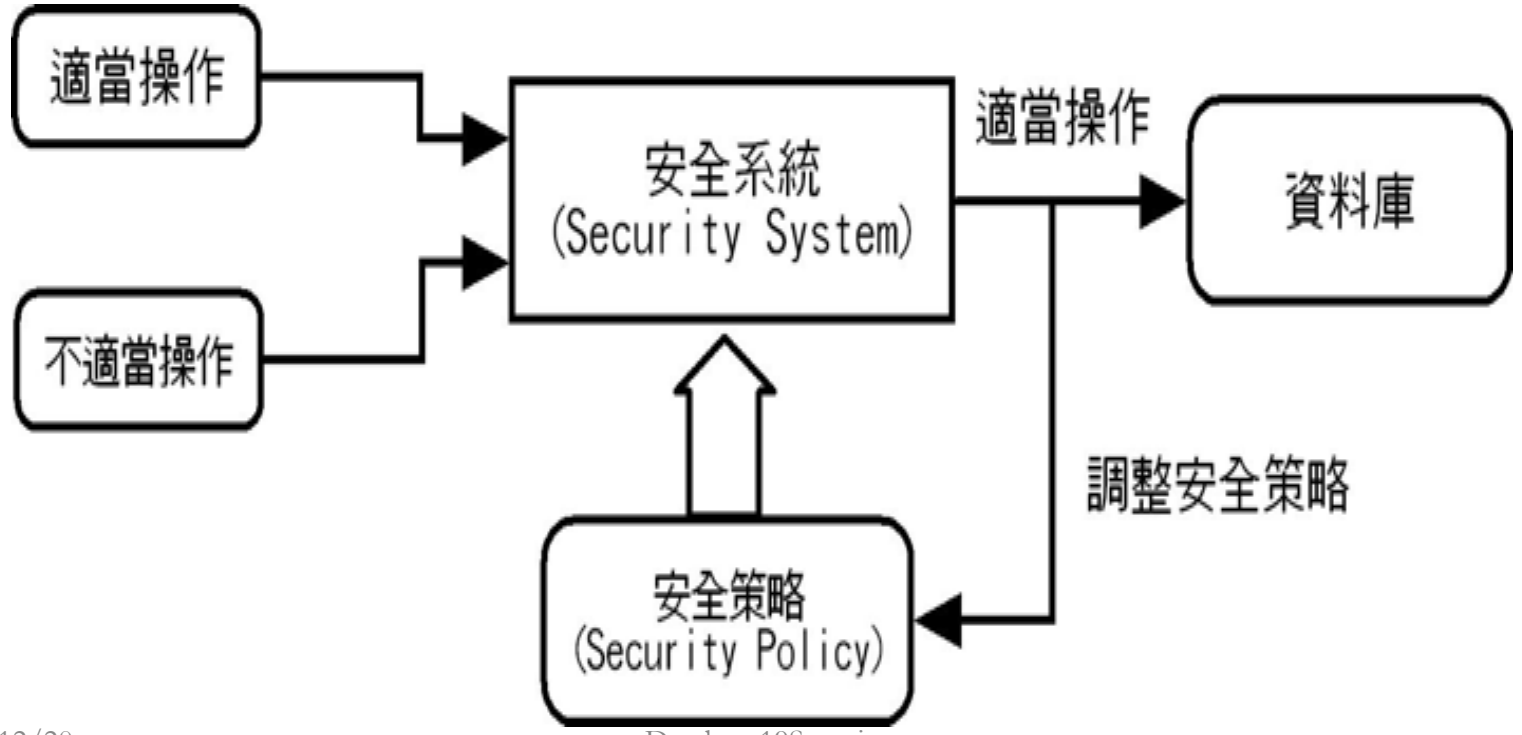
考一題

資料庫安全管理

- **資料庫安全的基礎**
- **存取控制**
- **資訊流控制**
- **推論控制**
- **資料加密**

資料庫安全的基礎

- 資料庫安全（Database Security）是資料庫系統的安全機制，保護資料庫儲存的資料，不讓沒有得到授權的使用者進行存取



資料庫安全的術語說明

- **安全性（Security）**：保護資訊不被未授權的讀取、更改和刪除；使用者只能存取擁有授權的資料
- **策略（Policy）**：定義保護資料的原則
- **機制（Mechanism）**：使用硬體或軟體程序來強制執行策略（Policy）定義的原則(規則)

資料庫安全層級

- **資料庫的安全層級，除了資料庫系統外，還包括作業系統的軟硬體和使用者**
  - **資料庫系統層級**（Database System Level）：限制使用者只能存取允許存取的資料(**本章主題**)
  - **作業系統層級**（Operation System Level）：指作業系統使用者權限(造成)的資料庫安全問題
  - **網路層級**（Network Level）：**在網路上**傳送的資料可能未經授權而被截取，所以在網路上傳送資料需要進行資料加密（Encryption）以保障資料安全 考題有開
  - **實體層級**（Physical Level）：電腦硬體的損壞或錯誤，都有可能造成資料庫資料的損壞
  - **使用者層級**（Human Level）：人為錯誤或非專業人員的控管也會產生資料庫安全問題(取消離職員工帳號、定時更改密碼)

資料庫安全的目標

- 保密性(C) (Confidentiality)
- 完整性(I) (Integrity)
- 可用性(A) (Availability)
- 驗證性(A) (Authentication)

NEXT

資料庫安全的目標–保密性 (Confidentiality)

- 採用適當的安全機制，預防、制止和偵測未授權的資料或資源存取，造成資料的外洩
  - 要有機制使部門員工不能查詢經理薪水和其他員工獎金

RETURN

資料庫安全的目標–完整性 (Integrity)

- 完整性是指確保維持資料原來的狀態，只允許有權限的使用者可以修改資料內容；即要有機制預防、制止和偵測未授權的資料遭竄改，以維持資料的正確性
  - 要有機制防止部門員工更改自己的薪水
  - 要有機制防止公司的商品售價、業績、銷售量等資料被隨意更改

不能讓其他未經授權的人來亂改

RETURN

資料庫安全的目標–可用性(Availability)

- 可用性是當合法使用者(被授權可使用的資源)要求使用資訊系統時，必須確保資訊與系統能夠持續營運、正常使用
  - 例如：每月5日是發薪日，系統不能因故，導致無法匯款到各位員工的銀行帳戶，簡單的說，資料庫系統不能停擺，不論何種情況，一定需要維持系統的正常運作

應該隨時可用

RETURN

資料庫安全的目標–驗證性(Authentication)

- 驗證性是指資料庫管理系統應具備檢查和確認使用者的身份，以提供適當的安全機制，及實施安全機制於哪一個資源
  - 例如：使用者Joe擁有權限存取Students關聯表，資料庫管理系統需要確認使用者真的是Joe，才能讓他存取Students關聯表

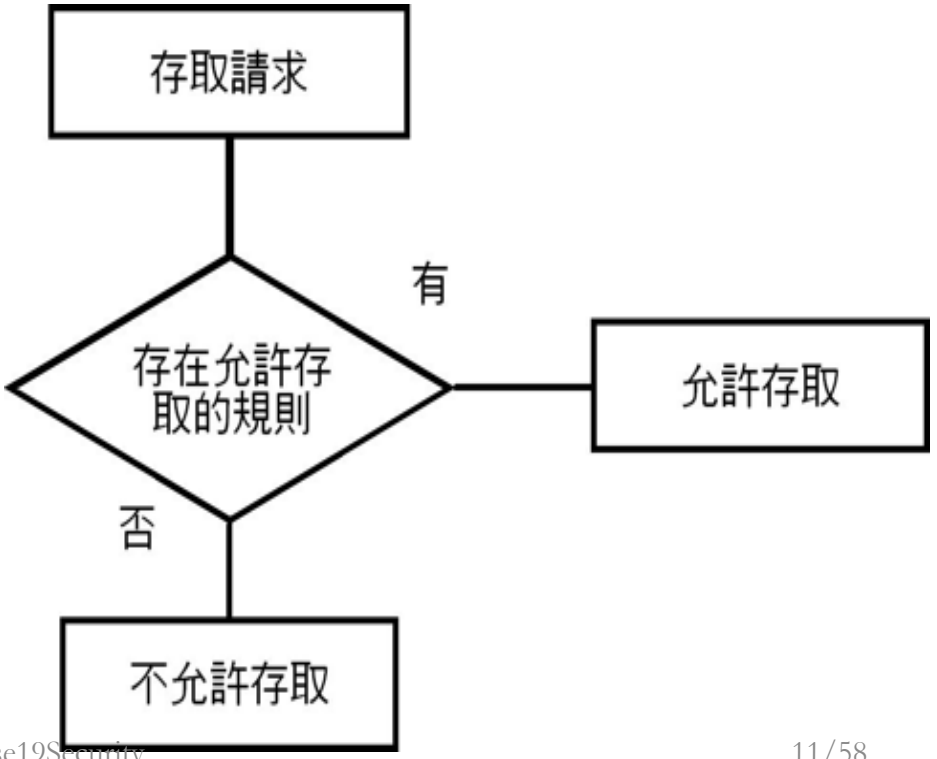
RETURN

資料庫的安全策略

- 「安全策略」（Security Policy）是一套強制遵守的原則，用來維護系統的安全，這是一種概念性原則(高階指引)，而不是實作的詳細規則

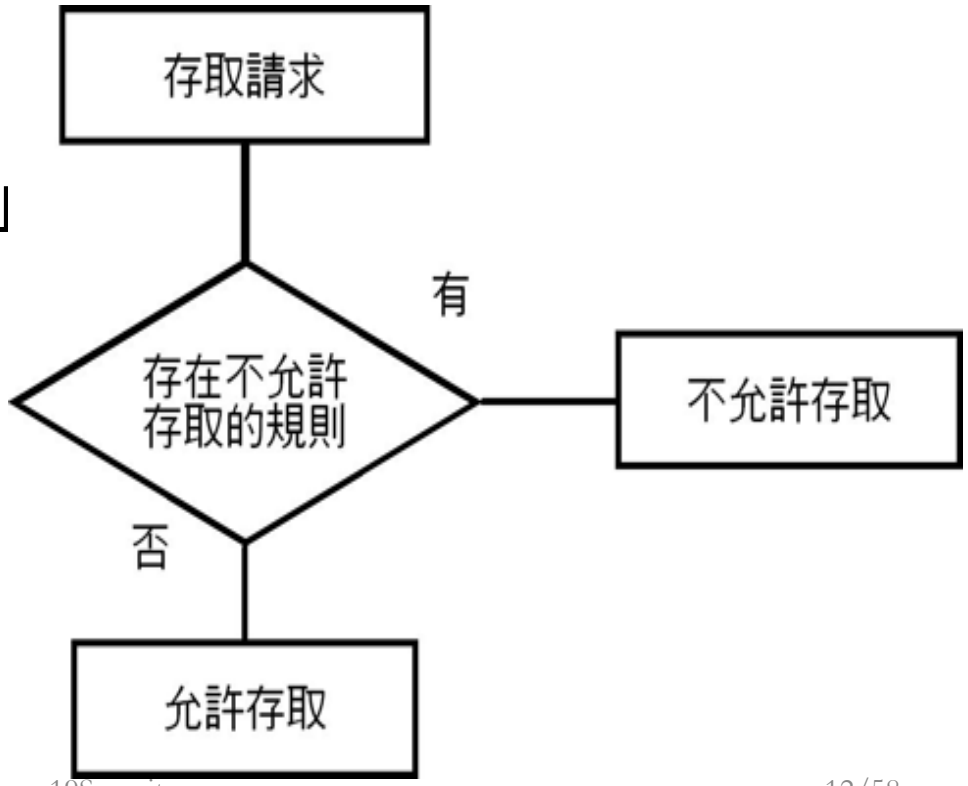
資料庫的安全策略–最小特權策略 (Minimum Privilege Policy)

- 最小特權策略是一種擁有最強資料保密性的安全策略，使用者需要擁有對某資料的某權限，才能存取此資料，對於沒有指定是否擁有權限的資料，一律拒絕存取
- 使用這種安全策略的系統稱為「封閉系統」(Closed Systems)，實作的權限規則處理流程，如右圖所示：



資料庫的安全策略–最大特權策略 (Maximum Privilege Policy)

- 最大特權策略是一種擁有最大資料可用性的安全策略，使用者如果沒有被設定為不允許存取權限，就一定可以存取
- 使用這種安全策略的系統稱為「開放系統」(Open Systems)，實作的權限規則處理流程，如右圖所示：



SQL隱碼威脅

- SQL資料隱碼(**Injection**)威脅是指將某些資料放入SQL指令中，進行資料庫攻擊，例如：竄改資料庫或擷取敏感資料，它也可能被用來執行系統層級命令，導致系統拒絕服務應用程式，隱碼威脅類型包括
  - 非法擴大權限
  - 濫用權限(系統管理者擅改學生資料)
  - 阻絕服務(**Denial of service (DOS) attack**)：藉由網路塞爆緩衝區或耗光資源，讓使用者無法使用網路或資料
  - 薄弱的驗證機制：攻擊者取得合法使用憑證，而冒用該使用者身分

CCIAA)

各種不同類型的資料隱碼攻擊1/3

- 竄改SQL：

  SELECT * FROM users WHERE username = ‘jake’and PASSWORD = ‘jakespasswd’.

  攻擊者可嘗試竄改SQL敘述如下：

  SELECT * FROM users WHERE username = ‘jake’and （PASSWORD = ‘jakespasswd’or ‘x’=‘x’.

- 攻擊者只要知道有使用者‘jake’即可，即可登入DB執行‘jake’的權限

各種不同類型的資料隱碼攻擊2/3

- 注入程式碼：是利用電腦程式因為處理無效資料而發生的錯誤，而新增加額外的SQL敘述或在現有的SQL敘述新增命令
- 攻擊者可注入或引入程式碼進入電腦程式中，而改變執行路徑；是駭入系統或破壞系統以取得資訊常用的方法

各種不同類型的資料隱碼攻擊3/3

- 注入函式呼叫：在有弱點的SQL敘述中插入某個資料庫函式或作業系統函式呼叫，來竄改資料或進行需要特權的系統呼叫
  - 訂做的資料庫套裝軟體或資料庫函式中包含的函式，也可能被利用來當作SQL查詢的一部份
  - 特別是動態建立的SQL查詢，最可能被利用，因為他們是在執行期間才建立的

SQL資料隱碼攻擊相關的風險

- 資料庫特徵
- 阻絕服務
- 略過驗證
- 找出可注入的參數
- 執行遠端命令
- 擴大權限

防禦SQL資料隱碼攻擊的技術

- 針對所有Web可存取的程序和函式，規定某些程式設計規則
  - 繫結變數：也稱為參數，可抵禦資料隱碼攻擊並改善效能。以JAVA和JDBC程式碼為例：使用者ID與PWSD不顯示在程式中

```
PreparedStatement stmt = conn.prepareStatement( "SELECT * FROM
    EMPLOYEE WHERE EMPLOYEE_ID=? AND PASSWORD=?");
stmt.setString(1, employee_id);
stmt.setString(2, password);
```

  - 篩選輸入：設法防堵Escape字元的輸入，防堵SQL竄改輸入
  - 函式的安全性：強化標準或自訂函式的安全性

資料庫的安全策略–安全策略

- 資料庫的安全策略可以分成很多類別，這些類別就是達成資料庫安全目標的相關方法，如下所示：
  - 存取控制（Access Control）
  - 資訊流程控制（Information Flow Control）
  - 推論控制（Inference Control）
  - 資料加密（Encryption）

NEXT

存取控制

- 存取控制（Access Control）是控制資料或資源的讀取、寫入和刪除，明確指定誰可以存取和如何存取哪些資料或資源。存取控制策略（Access Control Policy）可以分為數種，如下所示：
  - 自由選擇存取控制（Discretionary Access Control, DAC）
  - 強制存取控制（Mandatory Access Control, MAC）
  - 角色基礎存取控制（Role-Based Access Control）

RETURN

自由選擇存取控制

GRANT、REVOKE

- 自由選擇存取控制（Discretionary Access Control, DAC）透過授予或撤回使用者擁有指定物件讀取、寫入和刪除權限
- 其觀念是源於所有權（Ownership），使用者一旦擁有物件的所有權，就可以授予或撤回其具有所有權物件的使用權限

自由選擇存取控制–資料庫管理系統的自由選擇存取控制

- 在資料庫管理系統的自由選擇存取控制是使用授予（Grant）或撤回（Revoke）方式來指定使用者的權限，使用者權限的層級可以分為兩種，如下所示：
  - **帳戶層級權限（Accout Level Privileges）**：授予使用者與關聯表內容無關的權限；如授以CREATE DATABASE、CREATE TABLE、CREATE VIEW之權限
  - **關聯表層級權限（Relation Level Privileges）**：授予使用者與關聯表內容有關的權限，如授以對Student資料表SELECT、UPDATE、DELETE等權限

RETURN

強制存取控制–說明

- 強制存取控制（Mandatory Access Control, MAC）屬於高階和集中式的安全性控制，這是一種管理者基礎的存取控制，存取控制是由系統管理者指定，將物件和使用者權限分類成多種不同階層的安全等級，使用者只能存取授予的安全等級，或等級低的物件

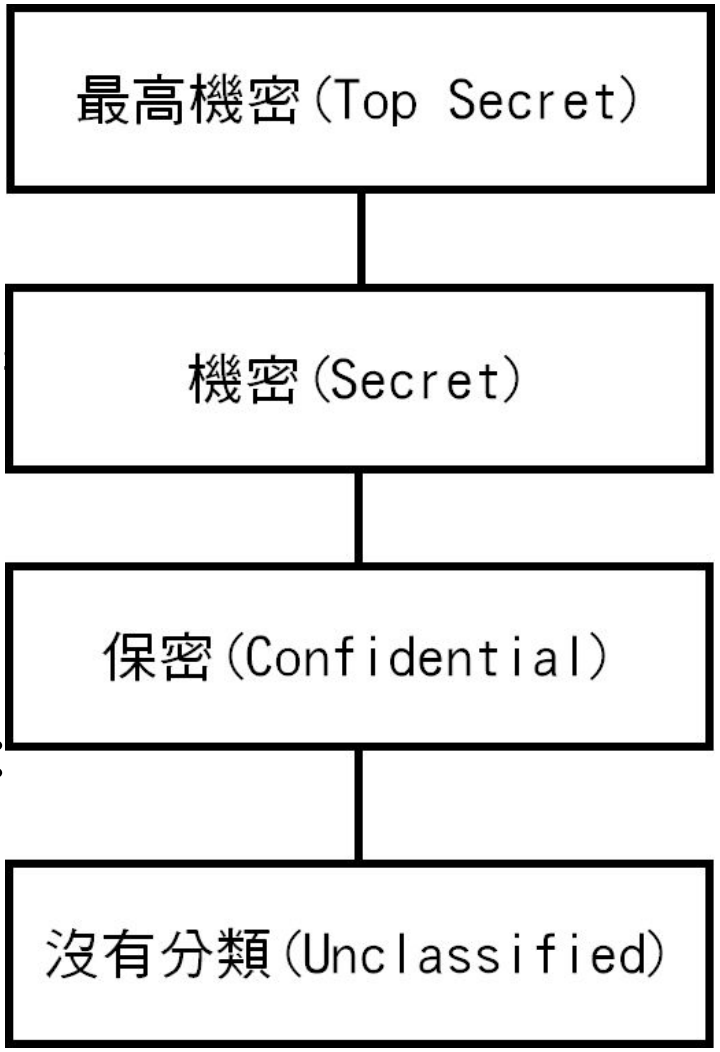
將使用者分級
資料也分級

使用者僅可存取相同層級的資料
或低

寫入：高層級

強制存取控制–BLP模型

- BLP模型是一種安全模型（Security Model），屬於一種「狀態機器模型」（State-Machine Model）這是1975年由Bell Lapadula建立的安全模型，以安全等級（Security Level）來處理存取控制，一共分成四級，如右圖所示：



強制存取控制–屬性的狀態值

- BLP使用屬性的狀態值來定義資料的安全性和讀寫規則，其說明如下所示：
  - **簡單安全屬性(特性)（The Simple Security Property，簡稱ss-property）**：不能往上讀取（no-read-up），使用者不允許讀取比它安全等級高的資料
  - **星號屬性(特性)（The Star-property，簡稱*-property）**：不能往下寫入（no-write-down），使用者不允許寫入比它安全等級低的資料 因為可能導致資料外洩

強制存取控制–資料庫管理系統的強制存取控制

- 強制存取控制在關聯表需要新增屬性記錄資料的安全等級，每一個屬性擁有安全等級分類屬性。例如：Students關聯表使用強制存取控制的關聯表綱要，如下所示：

Students (sid, csid, name, cname, birthday, cbirthday, GPA, cGPA)

Students

sid	name	birthday	GPA
S001(U)	陳會安(S)	1967/9/3(S)	3.7(S)
S002(U)	江小魚(C)	1978/2/2(C)	3.0(C)
S003(U)	張三丰(U)	1982/3/3(U)	3.2(C)
S004(U)	李四方(U)	1981/4/4(U)	2.9(C)

強制存取控制範利

Students

sid	name	birthday	GPA
S001(U)	陳會安(S)	1967/9/3(S)	3.7(S)
S002(U)	江小魚(C)	1978/2/2(C)	3.0(C)
S003(U)	張三丰(U)	1982/3/3(U)	3.2(C)
S004(U)	李四方(U)	1981/4/4(U)	2.9(C)

- 一位C等級使用者僅能見到之資料

Students

sid	name	birthday	GPA
S001(U)	NULL (C)	NULL (C)	NULL (C)
S002(U)	江小魚(C)	1978/2/2(C)	3.0(C)
S003(U)	張三丰(U)	1982/3/3(U)	3.2(C)
S004(U)	李四方(U)	1981/4/4(U)	2.9(C)

- 一位U等級使用者僅能見到之資料

Students

sid	name	birthday	GPA
S001(U)	NULL (U)	NULL (U)	NULL (U)
S002(U)	NULL (U)	NULL (U)	NULL (U)
S003(U)	張三丰(U)	1982/3/3(U)	NULL (U)
S004(U)	李四方(U)	1981/4/4(U)	NULL (U)

RETURN

角色基礎存取控制–說明

- 角色基礎存取控制（Role–Based Access Control）的存取權限是基於使用者在公司或組織所扮演角色來決定其權限
- 存取權限使用角色名稱來分類和組織，當使用者授予角色權限後，使用者就擁有該角色所付予的權限

角色基礎存取控制–SQL Server的伺服器角色

伺服器角色	說明
System Administrators	SQL Server 的系統管理者，擁有最大權限的使用者，即 sysadmin
Security Administrators	管理登入與 CREATE DATABASE 指令的權限，可以讀取錯誤記錄檔，即 securityadmin
Server Administrators	負責設定伺服器範圍的組態選項和關閉伺服器，即 serveradmin
Setup Administrators	管理連接伺服器的相關設定與預存程序，即 setupadmin
Process Administrators	管理 SQL Server 的處理程序(Process)，即 processadmin
Disk Administrators	管理磁碟的資料庫檔案，即 diskadmin
Database Creators	擁有建立、更改、卸除資料庫和更改資料庫屬性的權限，即 dbcreator
Bulk Insert Administrators	擁有執行 BULK INSERT 指令的權限，即 bulkadmin

RETURN

資訊流控制

- 資訊流控制（Information Flow Control）是指在執行程式時，控制程式存取的資料不會從高保護物件明顯或不明顯的流向低保護物件
  - 例如：將物件X複製到物件Y就擁有資訊流從X到Y，因為從物件X讀取資料，然後寫入物件Y
  - 以資料庫安全來說，資訊流控制可以進一步防範木馬程式（Trojan Horse）等駭客程式來非法存取資料庫的資料

資訊流控制的解決方案

- 給予每個程式一個安全等級(通常稱為許可證(clearance))
- 讀取：當程式的安全等級與此記憶區段至少同樣高才能執行(只能讀取安全等級較低的資料)
- 寫入：當程式的安全等級至少與此區段同樣低時才可寫入(只能寫入安全等級較高的資料)
  - 例如某個安全等級S的軍用程式，只能讀取安全等級U或C的物件，而且也只能寫入安全等級S或TS的物件中

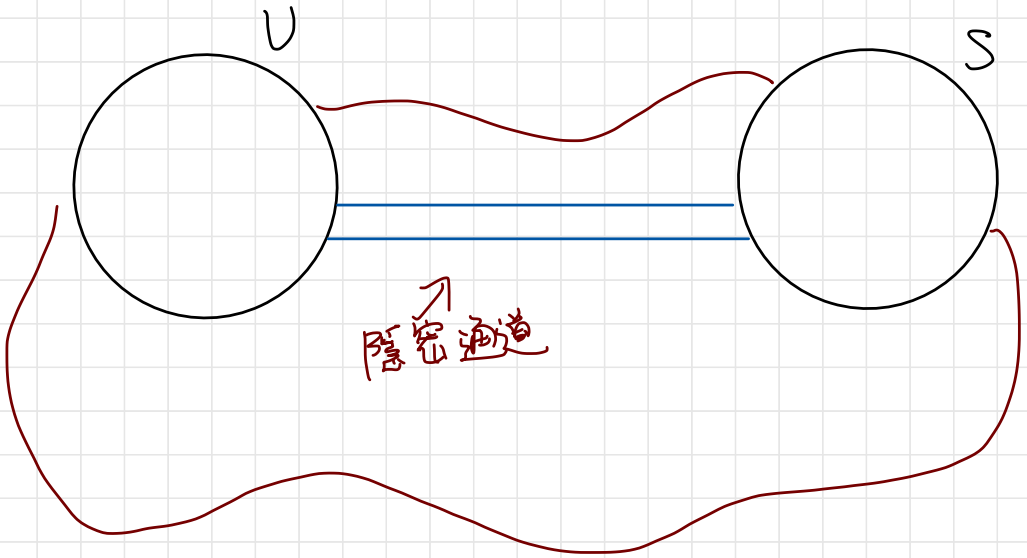
與前面的原則相同，讀取：低、寫入：高
不一樣的地方：隱密通道（下一頁）

隱密通道

- 隱密通道 (covert channel) 是允許透過不適當的方法，讓資訊從較高安全等級流向較低安全等級
- 在分散式資料庫有二節點，其使用者安全等級分別為S和U，假設兩節點都要Commit，但根據星號特性(*property)S端不能寫入或傳遞資訊給U端，但若之間設一秘密通道，讓機密資料無條件在U端交付，而在S端先定義雙方都同意的方式，讓特定資訊能從S端傳到U端

RETURN

資訊流



推論控制

- 資料庫安全除了存取控制外，因為資料流控制也可以算是一種程式的存取控制，還需要避免推論的資料庫安全問題，推論控制（Inference Control）是保護資料不被非直接存取，禁止使用者以推論方式來取得沒有授權的資訊

推論的範例–查詢敏感資訊 (Queries Conditioned on Sensitive Information)

- 在資料庫查詢敏感資訊所導致的推論問題，例如：員工薪水屬於部門的敏感資訊，雖然不允許使用者直接查詢指定員工的薪水，不過透過WHERE子句，仍然可以查出員工薪水的敏感資訊，SQL指令敘述如下所示：

```
SELECT name FROM Employees
WHERE salary < 20000
```

推論的範例–統計推論 (Statistical Inference)

- 資料庫查詢的統計資料是使用SQL聚合函數，其導致推論的原因是使用者以其知識使用SQL語言的聚合函數從統計資料推導出沒有授權存取的資料
- 例如：SQL語言查詢學生的最高成績，如下所示：

```
SELECT MAX(GPA) FROM Students
WHERE name LIKE “會安”
```

推論的範例–資料庫完整性導致的推論 (Inferences via Database Integrity)

- 資料庫完整性限制條件也有可能產生推論問題。例如：資料庫擁有限制條件，如下所示：

  X+Y = 100

- 在上述限制條件中，如果使用者擁有存取Y的權限，當知道Y的值後，經簡單運算就可以知道X的值

推論的範例–遺失資料 (Missing Data)

- 遺失資料是指低安全等級的使用者在查詢資料時，發現某些資料不存在，這些遺失資料反而啟發使用者，洩漏了機密資料

推論控制的解決方案

- 禁止在資料庫執行統計查詢時，其選取的值組範圍明顯小於一個數量，例如：禁止只有一筆值組的統計查詢
- 禁止重複一序列的相似查詢，而且每次查詢選取的值組範圍都是相同範圍
- 遺失資料的推論可以使用「多重快照」（Polyinstantiation）技術，視使用者的安全等級，允許同一個關聯表可以存在不同的值組內容，如此即可欺騙低安全等級的使用者，讓他們不知道到底哪一次貨運包含機密貨物

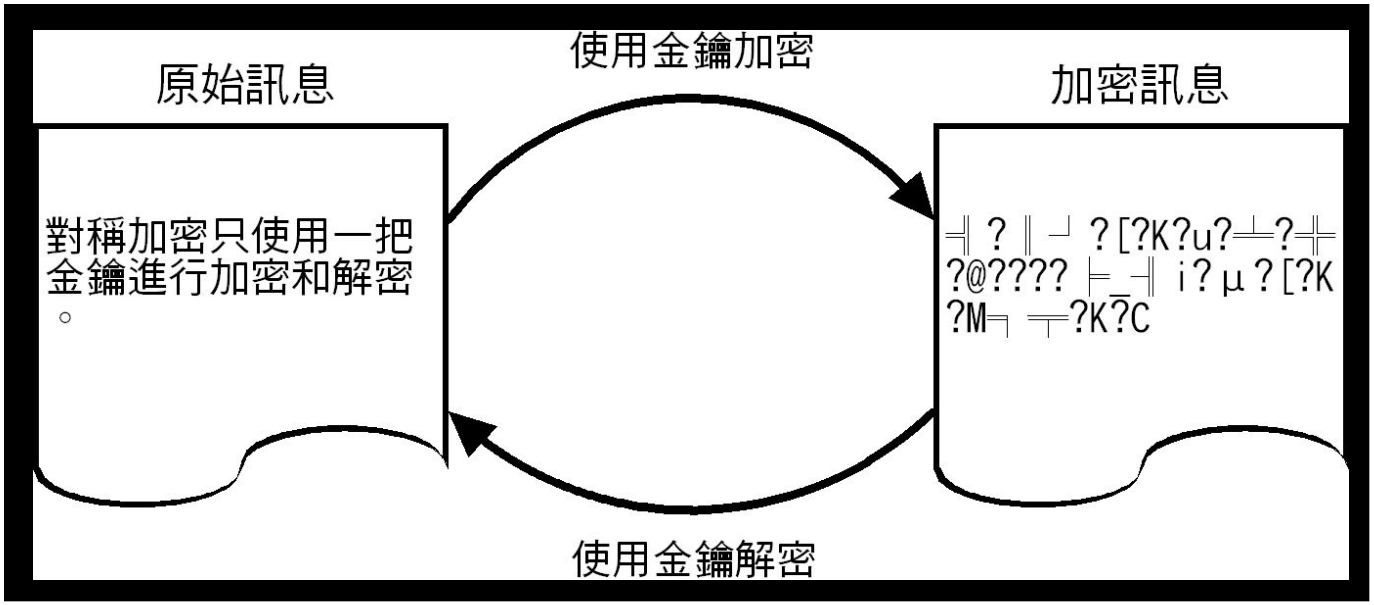
RETURN

資料加密

- 資料加密（Encryption）是將資料庫的資料使用數學運算的演算法加密（Encrypt）成不是一眼就可以看出的一堆亂碼，以保護資料的安全，因為只有授權使用者才可以解密（Decrypt）成原來資料，看到資料的內容

對稱加密

- 「對稱加密」（Symmetric Encryption）只使用同一把金鑰（Key）進行資料的加密和解密，如下圖所示：

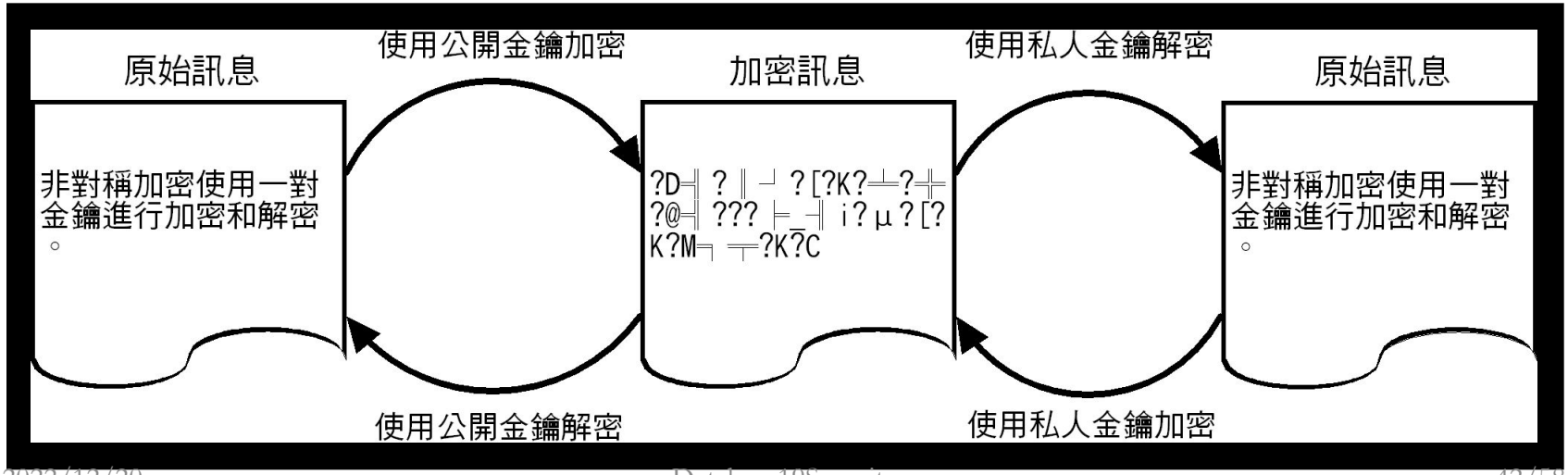


非對稱加密–說明

- 「非對稱加密」（Asymmetric Encryption）是使用一對金鑰（Key）來加密和解密資料，公開金鑰密碼系統（Public Key Cryptograpy）是1976年由Whitfield Diffie和Martin Hellman開發的密碼系統，這就是一種非對稱加密（Asymmetric Encryption）

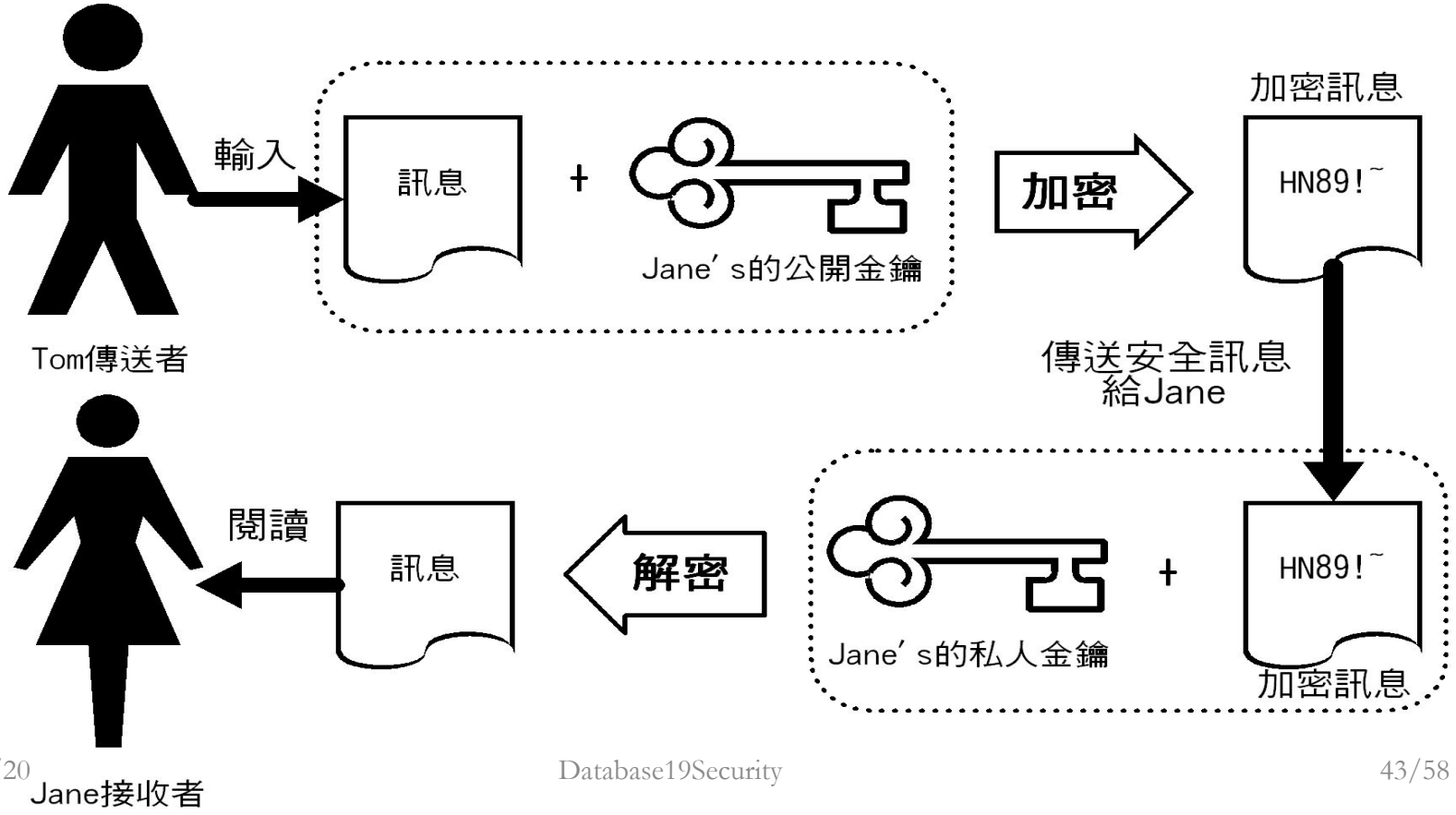
非對稱加密–圖例

- 非對稱加密系統使用兩把一對金鑰，一把稱為公開金鑰（Public Key），一把稱為私人金鑰（Private Key），這是一對金鑰，如果使用其中一把金鑰替資料加密，就只能使用另一把金鑰才可以解密資料，看到資料內容，如下圖所示：



使用非對稱加密傳送安全訊息

- 傳送者需要使用接收者的公開金鑰來加密訊息，接收者是使用私人金鑰來進行解密



使用非對稱加密建立數位簽章

- 「數位簽章」（Digital Signatures）如同手寫簽名，其目的是為了證明確實是某位使用者送出的訊息，換個角度，使用者可以使用數位簽章證明這是我送出的訊息
- 使用非對稱加密建立數位簽章的過程和上一節相反，他是使用發信者的私人金鑰來加密建立數位簽章，收信者是使用公開金鑰來驗證發信者的身份

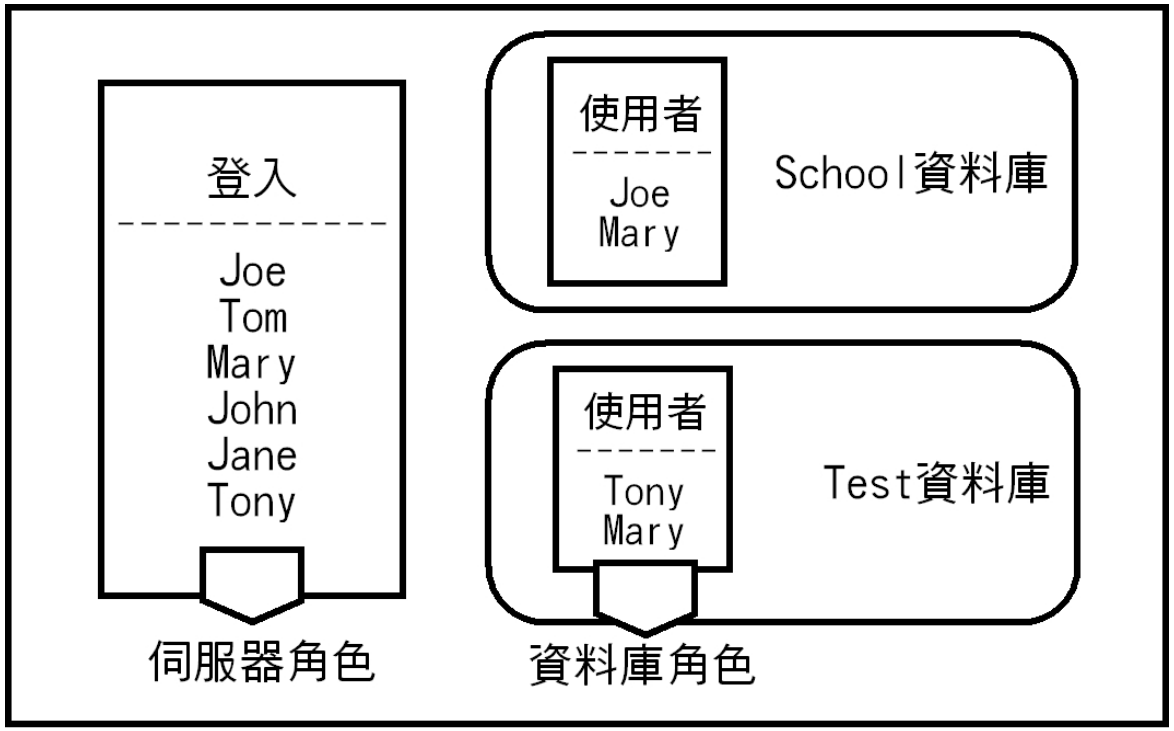
RETURN

使用者管理–說明

- 使用者管理是資料庫管理系統的身份識別系統，因為只有在資料庫管理系統擁有使用者帳號和密碼的使用者，才允許建立資料庫連線，可以使用、設計、建立和維護資料庫
- 資料庫系統的使用者管理是一套與作業系統不同的使用者管理機制，分別擁有不同的使用者帳號清單。換句話說，在作業系統擁有的使用者帳號，並不表示資料庫管理系統也擁有使用者帳號

使用者管理–SQL Server使用者帳號

- SQL Server的使用者帳號共分為兩種：登入和資料庫使用者，如下圖所示：



SQL Server資料庫伺服器

使用者管理–使用者帳號的建立

- 在SQL Server資料庫管理系統建立使用者帳號分成兩個部分，首先建立登入帳號，即允許登入SQL Server伺服器的使用者帳號，然後才能針對指定或目前資料庫，建立資料庫使用者
- Transact–SQL是使用CREATE LOGIN指令來建立登入帳號，以便驗證使用者登入SQL Server伺服器，其基本語法如下所示：

```
CREATE LOGIN login_name [WITH PASSWORD =
    'password']
[FROM WINDOWS]
```

- 在建立登入帳號後，我們可以針對指定資料庫或目前資料庫，使用CREATE USER指令建立資料庫使用者，其基本語法如下所示：

```
CREATE USER user_name [FOR LOGIN login_name]
```

使用者管理–使用者帳號的刪除建立

- 當使用者沒有連線資料庫時，我們可以使用DROP USER指令刪除資料庫使用者，其基本語法如下所示：

DROP USER user_name

- 上述語法可以刪除user_name使用者。至於刪除登入帳號是使用DROP LOGIN，其基本語法如下所示：

DROP LOGIN login_name

- 上述語法可以刪除login_name的登入使用者

使用者管理–指定資料庫使用者的角色1

- SQL Server可以指定資料庫使用者快速授予使用者的存取權限。主要資料庫角色和權限說明，如下表所示：

資料庫角色	說明
public	所有使用者都擁有此角色的權限，可以瀏覽資料表、檢視和執行預存程序，但沒有存取權限
db_owner	資料庫的擁有者，預設的資料庫使用者 dbo 就屬於此角色，他擁有資料庫的全部權限
db_datareader	使用者擁有查詢資料庫記錄的權限，也就是執行 SELECT 指令
db_datawriter	使用者擁有資料表記錄的新增、刪除和更新權限，也就是執行 INSERT、DELETE 和 UPDATE 指令
db_accesadmin	此角色可以建立和管理資料庫使用者

使用者管理–更改登入密碼

- 資料庫登入密碼的更改是使用ALTER LOGIN指令，其基本語法如下所示：

ALTER LOGIN login_name WITH PASSWORD = new_password

- 上述語法可以將密碼改成新密碼new_password

GRANT授予權限–敘述權限

- 敘述權限（Statement Permission）是指定資料庫的管理權限，例如：更改資料表設計、建立檢視和備份資料庫等權限，即第16–2–1節的帳戶層級權限（Accout Level Privileges）
- GRANT指令授予敘述權限的基本語法，如下所示：

GRANT { ALL [PRIVILEGES] | privileges [,privileges…] }

TO user_name [,user_name ...]

- 上述語法的ALL或ALL PRIVILEGES是所有權限，也可以指明授予的管理權限privileges，TO子句是授予權限的資料庫使用者user_name

GRANT授予權限–物件權限

- 物件權限（Object Permission）是指資料庫物件的存取權限，包含資料表、檢視、預存程序和函數物件的存取，即第16–2–1節的關聯表層級權限（Relation Level Privileges）
- GRANT指令授予物件權限時，需加上ON子句來指定是針對哪一個物件，其基本語法如下所示：

```
GRANT { ALL [ PRIVILEGES ]
      | privileges [ (column [ ,columnn]) ]
  [,privileges...] }
ON table_name
TO user_name [,user_name ...]
```

- 上述語法可以在ON子句指定授予哪一個table_name資料表的權限，在privileges權限的括號中指定擁有哪些欄位的權限

GRANT OPTION傳遞權限

- GRANT OPTION指令是GRANT指令的選項子句，表示當GRANT指令授予使用者擁有的相關權限後，擁有權限的使用者還可以將權限授予其他使用者，其基本語法如下所示：

```
GRANT { ALL [ PRIVILEGES ]
      | privileges [ (column [ ,columnn]) ]
  [,privileges...] }
ON table_name
TO user_name [,user_name ...] WITH GRANT
  OPTION
```

- 上述語法和之前GRANT指令相似，只是在最後加上WITH GRANT OPTION選項子句，表示擁有權限的user_name使用者清單，可以再將權限傳遞給其他使用者

REVOKE撤回權限–敘述權限

- REVOKE指令可以徹回資料庫使用者的敘述權限（Statement Permission），其基本語法如下所示：

REVOKE { ALL [PRIVILEGES] |
privileges [,privileges...] }
FROM user_name [,user_name ...]

- 上述語法的ALL或ALL PRIVILEGES是所有權限，也可以指明徹回的系統權限，FROM子句是徹回權限的使用者清單

REVOKE撤回權限–物件權限

- REVOKE指令如果是徹回物件權限（Object Permission）時，還需加上ON子句來指定針對的物件，其基本語法如下所示：

```
REVOKE { ALL [ PRIVILEGES ]
        | privileges [ (column [ ,columnn]) ]
  [,privileges...] }
ON table_name
FROM user_name [,user_name ...]
```

- 上述語法可以在ON子句指定徹回哪一個table_name資料表的權限，在privileges權限的括號中可以指定徹回哪些欄位的權限

使用檢視實作資料庫安全–說明

- 視界（Views）在SQL Server稱為檢視，可以配合GRANT指令來實作資料庫安全規則，限制只能存取資料表指定的記錄或欄位

使用檢視實作資料庫安全–限制存取記錄

- 限制存取資料表的記錄可以使用檢視來實作。首先在SQL Server建立關聯表值組子集的檢視後，再將檢視的查詢權限授予指定的資料庫使用者
- 例如：限制使用者Joe在Instructors資料表，只允許查詢資訊系CS的講師清單，如下所示：

```
CREATE VIEW CS_View AS
SELECT * FROM Instructors
WHERE department = 'CS‘
GO
GRANT SELECT ON CS_View
TO Joe
```

使用檢視實作資料庫安全–限制存取欄位

- 限制存取資料表的欄位也可以使用檢視配合GRANT指令來建立，或直接使用GRANT指令來達成
- 例如：限制使用者Joe只能查詢Students資料表的連絡資料，不包含成績GPA欄位，如下所示：

```
CREATE VIEW CONTACT_View AS
SELECT sid, name, tel, birthday FROM
   Students
GO
GRANT SELECT ON CONTACT_View
TO Joe
```