Computer Security: Principles and Practice

Chapter 5 – Database Security

Back again!

Paolo Atzeni
Stefano Ceri
Stefano Paraboschi
Riccardo Torlone

Basi di dati

Modelli e linguaggi di interrogazione

Terza edizione

McGraw-Hill

web site

ONE YEAR LATER

Database Security

Database:

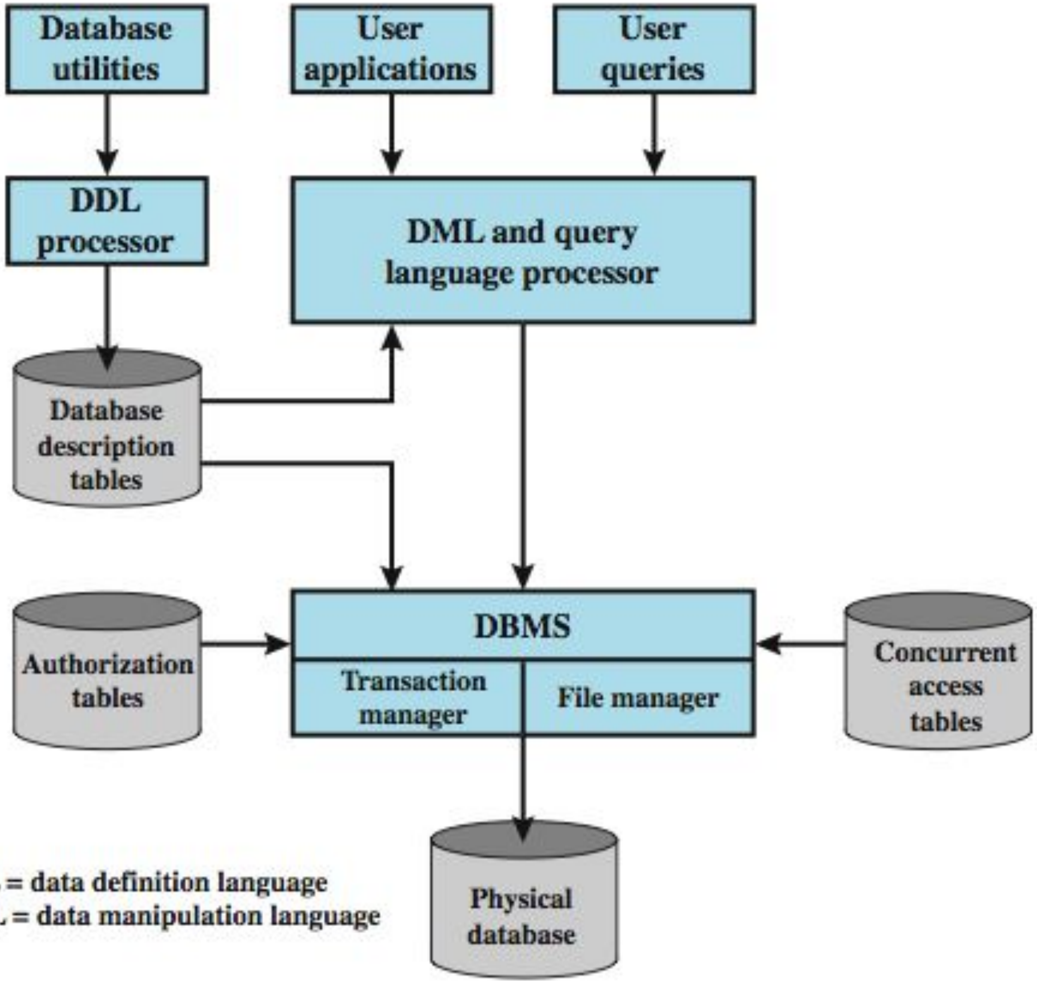
a structured set of data held in a computer ...

Database Security

Database:

a structured set of data held in a computer ...
….especially one that is accessible in various ways.

Database Security



DDL = data definition language
DML = data manipulation language

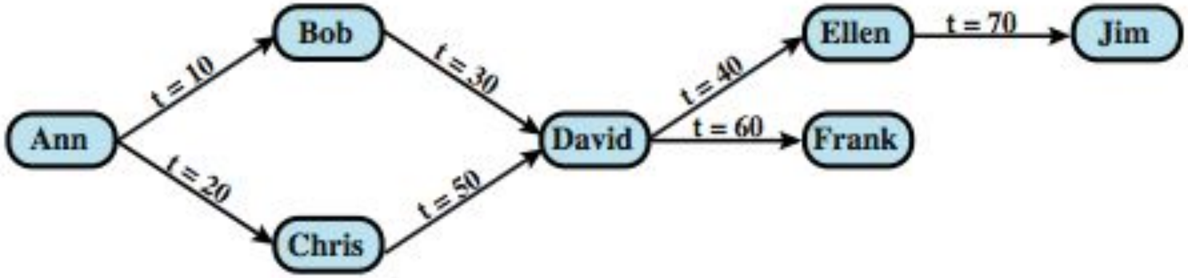
Database Access Control

- DBMS provide access control for database
- assume have authenticated user
- DBMS provides specific access rights to portions of the database
  - e.g. create, insert, delete, update, read, write
  - to entire database, tables, selected rows or columns
  - possibly dependent on contents of a table entry
- can support a range of policies:
  - centralized administration
  - ownership-based administration
  - decentralized administration

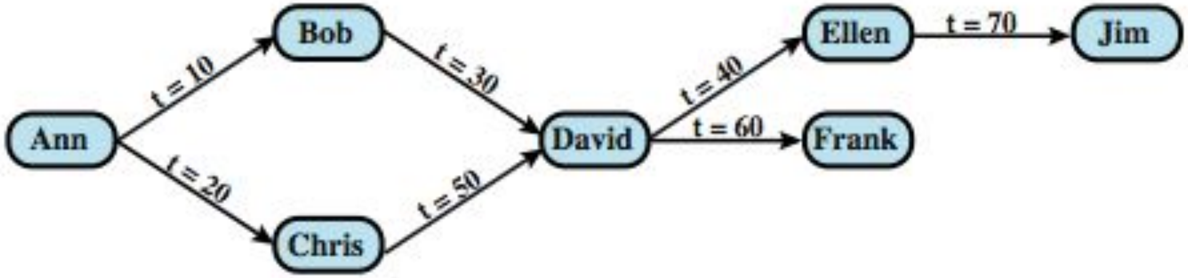
SQL Access Control

- two commands:
  - GRANT { privileges | role } [ON table] TO { user | role | PUBLIC } [IDENTIFIED BY password] [WITH GRANT OPTION]
    - e.g. GRANT SELECT ON ANY TABLE TO ricflair
  - REVOKE { privileges | role } [ON table] FROM { user | role | PUBLIC }
    - e.g. REVOKE SELECT ON ANY TABLE FROM ricflair
- typical access rights are:
  - SELECT, INSERT, UPDATE, DELETE, REFERENCES

Cascading Authorizations



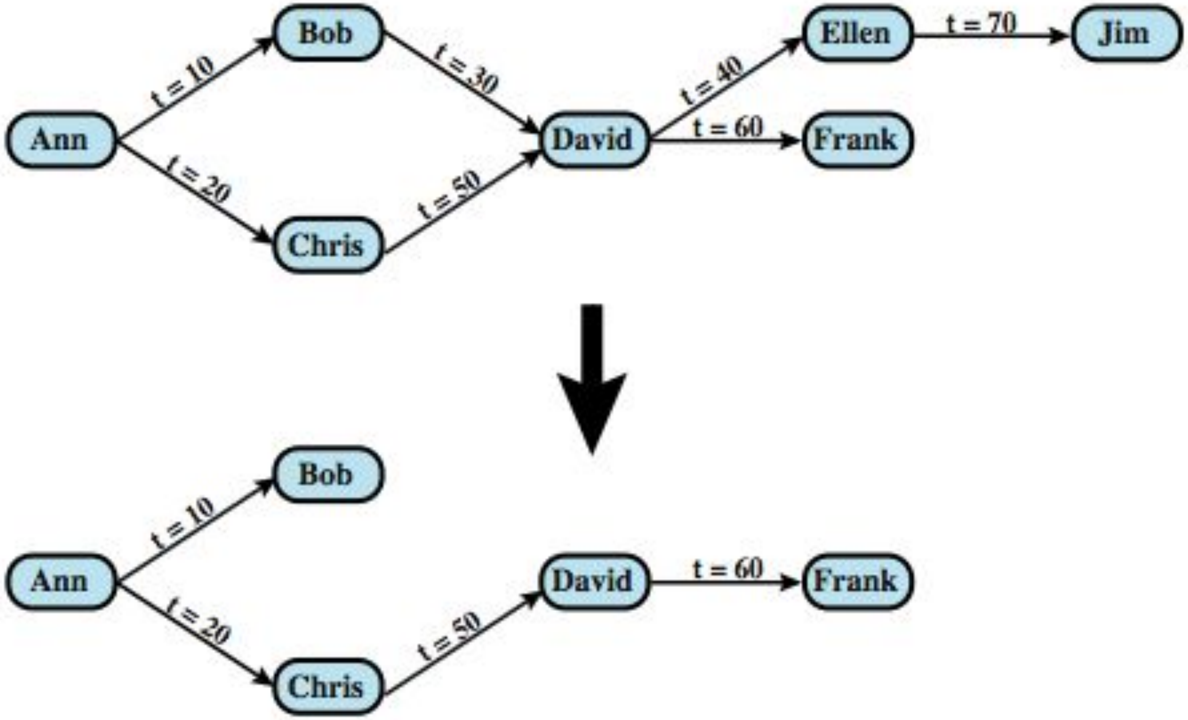
Cascading Authorizations



What if...

Bob revokes privilege from David?

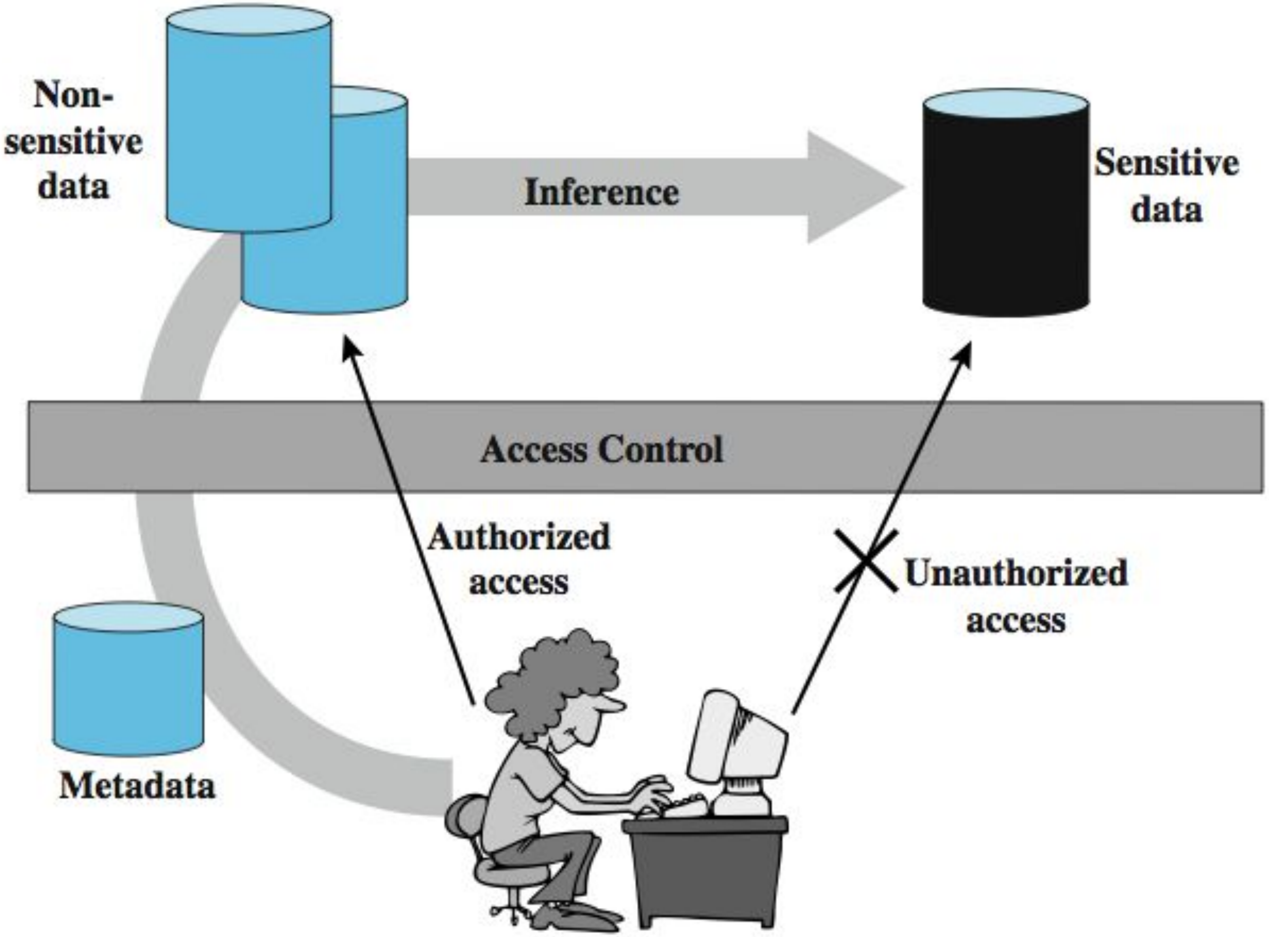
Cascading Authorizations



Role-Based Access Control

- role-based access control work well for DBMS
  - eases admin burden, improves security
- categories of database users:
  - application owner
  - end user
  - administrator
- DB RBAC must manage roles and their users
  - cf. RBAC on Microsoft’s SQL Server

Inference


Non-sensitive data
Inference
Sensitive data
Access Control
Authorized access
Unauthorized access
Metadata

Inference Example

posts		
id	**title**	**body**
1	Hello World	Lorem ipsum dolor sit amet, ...
2	Goodbye Heaven	Duis imperdiet lacus lobortis leo ...

Inference Example

secrets	
name	**value**
flag	FLAG{you_cant_reach_me}

Inference Example

```sql
SELECT * FROM `posts` WHERE `id` = '1'
```

```sql
SELECT * FROM `secrets`
  WHERE `name` = 'flag'
```

UNION

The UNION operator is used to **combine the result-set** of two or more SELECT statements.

- Each SELECT statement within UNION must have the **same number** of columns
- The columns must also have similar data types
- The columns in each SELECT statement must also be in the same order

UNION

union_fake_table		
id	**title**	**body**
1	Hello World	Lorem ipsum dolor sit amet, ...
2	Goodbye Heaven	Duis imperdiet lacus lobortis leo ...
3	flag	FLAG{you_cant_reach_me}

UNION

```
SELECT * FROM `posts` WHERE `id` = 'a'
UNION
SELECT 3,`name` `value` FROM `secrets`
  WHERE `name` = 'flag'
```

Information Schema - Tables

```
SELECT `table_schema`, `table_name`
  FROM INFORMATION_SCHEMA.tables
```

table_schema	table_name
database	posts
database	secrets

Information Schema - Columns

```
SELECT `table_name`, `column_name`
  FROM INFORMATION_SCHEMA.columns
  WHERE `table_name`   'secrets'
```

table_name	column_name
secrets	name
secrets	value

sqlite_master

```sql
SELECT `name`, `sqlite`
  FROM sqlite_master
```

name	sql
posts	CREATE TABLE posts (id INTEGER, title TEXT, body TEXT)
secrets	CREATE TABLE secrets (name VARCHAR(255), value TEXT)

Inference Countermeasures

- ➢ inference detection **at database design**
  - alter database structure or access controls
- ➢ inference detection **at query time**
  - by monitoring and altering or rejecting queries
- ➢ need some inference detection algorithm
  - a difficult problem
  - cf. employee-salary example

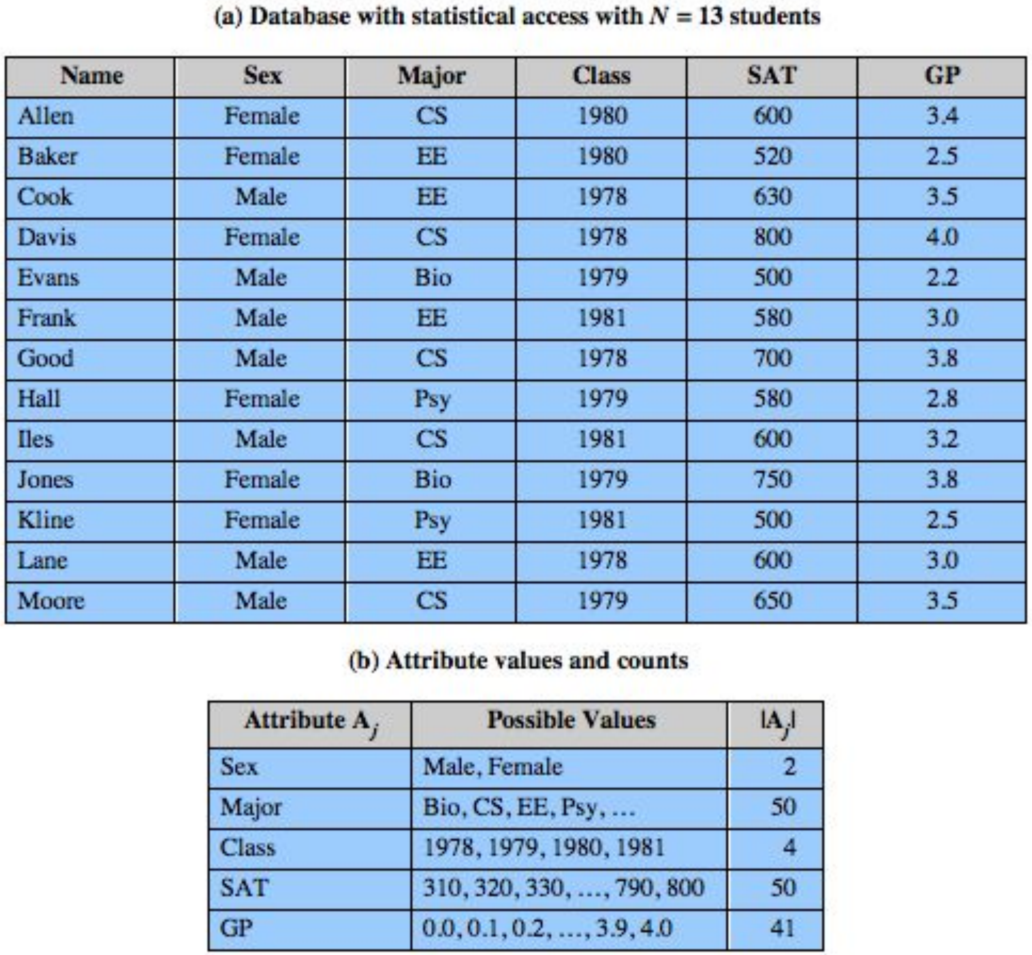
Statistical Databases

- provides data of a statistical nature
  - e.g. counts, averages
- two types:
  - **pure statistical database**
  - **ordinary database with statistical access**
    - some users have normal access, others statistical
- access control objective to allow statistical use without revealing individual entries
- security problem is one of inference

Statistical Database Security

- ➢ use a characteristic formula C
  - a logical formula over the values of attributes
  - e.g. `(`*`Sex`*`=Male) AND ((`*`Major`*`=CS) OR (`*`Major`*`=EE))`
- ➢ query set X(*C*) of characteristic formula *C*, is the set of records matching C
- ➢ a statistical query is a query that produces a value calculated over a query set

Statistical Database Example



(a) Database with statistical access with $N = 13$ students

Name	Sex	Major	Class	SAT	GP
Allen	Female	CS	1980	600	3.4
Baker	Female	EE	1980	520	2.5
Cook	Male	EE	1978	630	3.5
Davis	Female	CS	1978	800	4.0
Evans	Male	Bio	1979	500	2.2
Frank	Male	EE	1981	580	3.0
Good	Male	CS	1978	700	3.8
Hall	Female	Psy	1979	580	2.8
Iles	Male	CS	1981	600	3.2
Jones	Female	Bio	1979	750	3.8
Kline	Female	Psy	1981	500	2.5
Lane	Male	EE	1978	600	3.0
Moore	Male	CS	1979	650	3.5

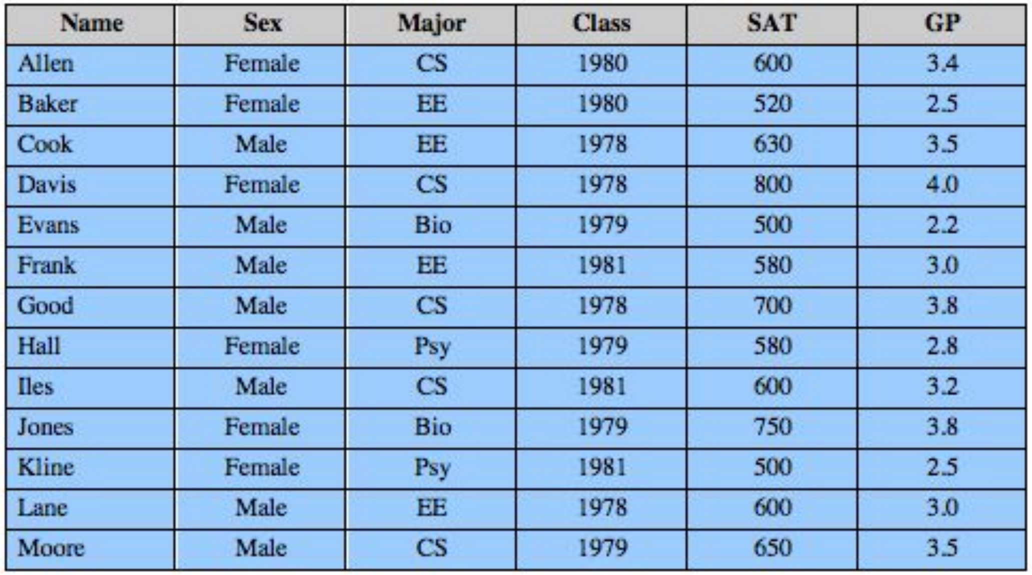
(b) Attribute values and counts

Attribute A_j	Possible Values	$\lvert A_j \rvert$
Sex	Male, Female	2
Major	Bio, CS, EE, Psy, ...	50
Class	1978, 1979, 1980, 1981	4
SAT	310, 320, 330, ..., 790, 800	50
GP	0.0, 0.1, 0.2, ..., 3.9, 4.0	41

Statistical Database Example

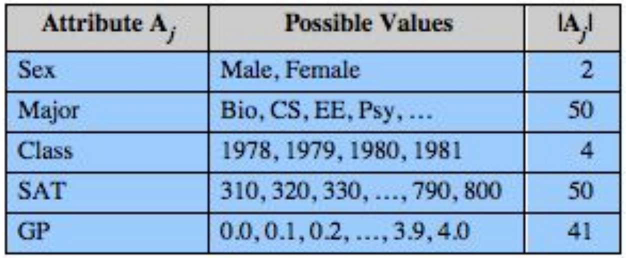
Grade of a student should not be revealed to Adv... (not even of Baker, EE student!)...

(a) Database with statistical access with $N = 13$ students



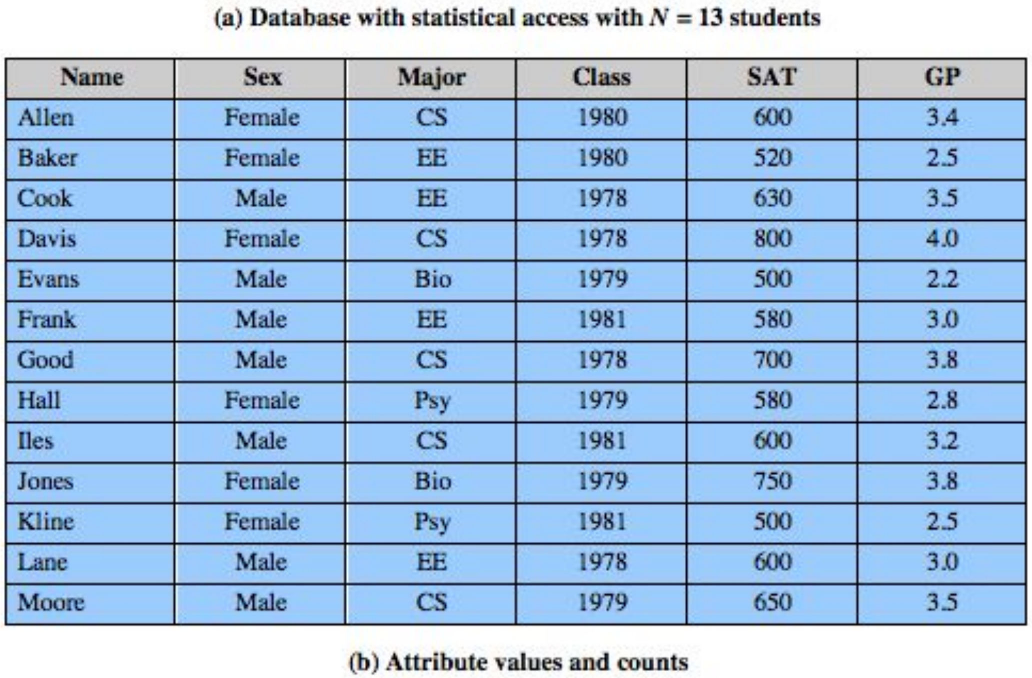
Name	Sex	Major	Class	SAT	GP
Allen	Female	CS	1980	600	3.4
Baker	Female	EE	1980	520	2.5
Cook	Male	EE	1978	630	3.5
Davis	Female	CS	1978	800	4.0
Evans	Male	Bio	1979	500	2.2
Frank	Male	EE	1981	580	3.0
Good	Male	CS	1978	700	3.8
Hall	Female	Psy	1979	580	2.8
Iles	Male	CS	1981	600	3.2
Jones	Female	Bio	1979	750	3.8
Kline	Female	Psy	1981	500	2.5
Lane	Male	EE	1978	600	3.0
Moore	Male	CS	1979	650	3.5

(b) Attribute values and counts



Attribute A_j	Possible Values	$\lvert A_j \rvert$
Sex	Male, Female	2
Major	Bio, CS, EE, Psy, ...	50
Class	1978, 1979, 1980, 1981	4
SAT	310, 320, 330, ..., 790, 800	50
GP	0.0, 0.1, 0.2, ..., 3.9, 4.0	41

Statistical Database Example



(a) Database with statistical access with $N = 13$ students

Name	Sex	Major	Class	SAT	GP
Allen	Female	CS	1980	600	3.4
Baker	Female	EE	1980	520	2.5
Cook	Male	EE	1978	630	3.5
Davis	Female	CS	1978	800	4.0
Evans	Male	Bio	1979	500	2.2
Frank	Male	EE	1981	580	3.0
Good	Male	CS	1978	700	3.8
Hall	Female	Psy	1979	580	2.8
Iles	Male	CS	1981	600	3.2
Jones	Female	Bio	1979	750	3.8
Kline	Female	Psy	1981	500	2.5
Lane	Male	EE	1978	600	3.0
Moore	Male	CS	1979	650	3.5

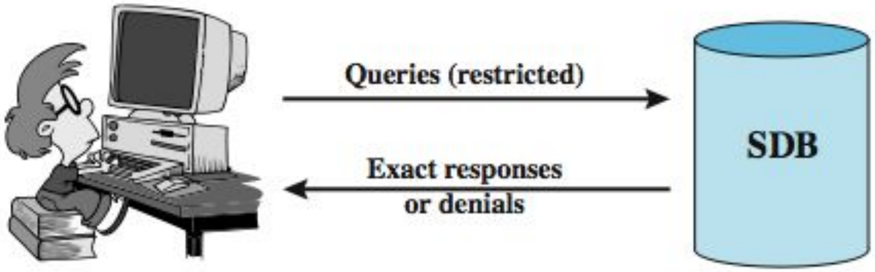
(b) Attribute values and counts

…

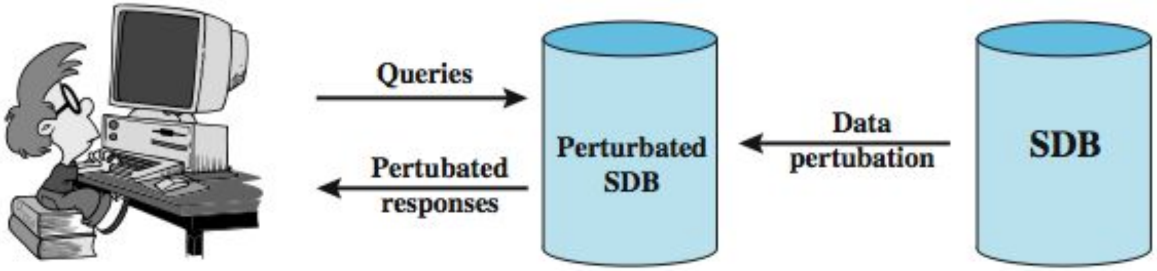
Count (EE*Female)=1

Sum(EE*Female,GP)=2.5

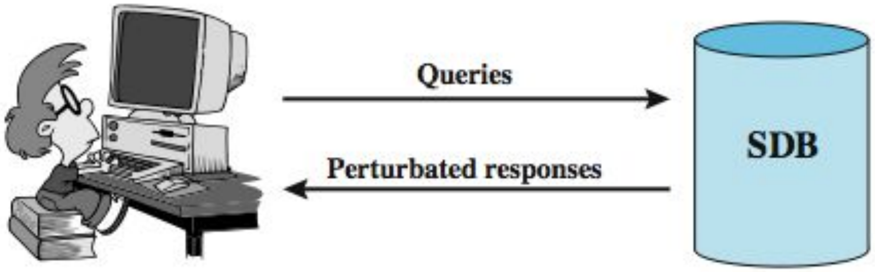
Protecting Against Inference



(a) Query set restriction



(b) Data perturbation



(c) Output perturbation

Query Restrictions

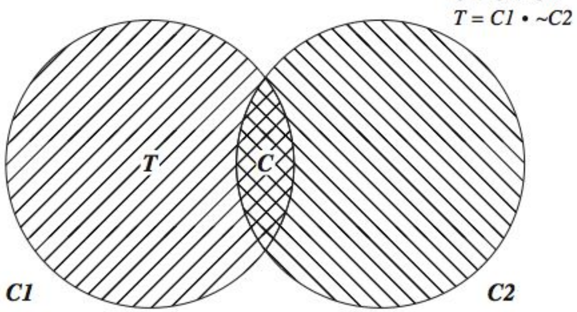
- query set overlap control
  - limit overlap between new & previous queries
  - has problems and overheads
- partitioning
  - cluster records into exclusive groups
  - only allow queries on entire groups
- query denial and information leakage
  - denials can leak information
  - to counter must track queries from user

Perturbation

- ➢ add noise to statistics generated from data
  - will result in differences in statistics
- ➢ data perturbation techniques
  - data swapping
  - generate statistics from probability distribution
- ➢ output perturbation techniques
  - random-sample query
  - statistic adjustment
- ➢ must minimize loss of accuracy in results

Tracker Attacks

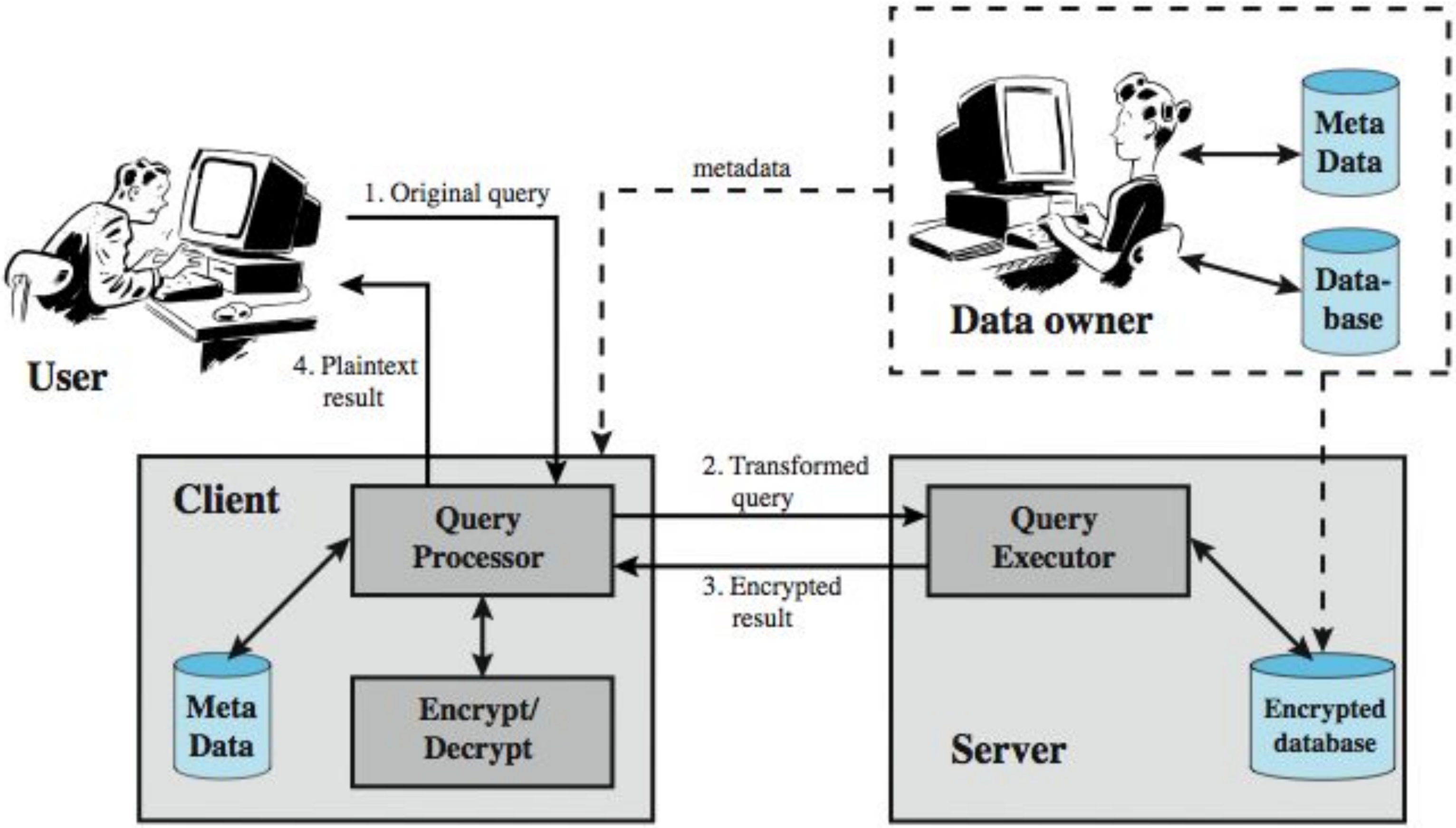
- ➢divide queries into parts
  - C = C1.C2
  - count(C.D) = count(C1) - count (C1. ~*C2)*
- ➢combination is called a tracker
- ➢each part acceptable query size
- ➢overlap is desired result



Database Encryption

- databases typical a valuable info resource
  - protected by multiple layers of security: firewalls, authentication, O/S access control systems, DB access control systems, and database encryption
- can encrypt
  - entire database - very inflexible and inefficient
  - individual fields - simple but inflexible
  - records (rows) or columns (attributes) - best
    - also need attribute indexes to help data retrieval
- varying trade-offs

Database Encryption



A Simple Query Involving Input from the User

```
SELECT * FROM `users`
  WHERE `email` = 'hello@example.com'
    AND `password` = 'abc123456'
```

Average Developer

```
$userQuery = mysqli_query
  “SELECT * FROM users
   WHERE email = '” . $_POST['email'] . “'
    AND password = '” . $_POST 'password'] .
  “'”

 ;
```

Login Request

User Input: hello@example.com
abc123456

```
SELECT * FROM `users`
  WHERE `email` = 'hello@example.com'
    AND `password` = 'abc123456'
```

Login Request

User Input:

```
admin@example.com' OR 1 = '1
abc123456
```

```
SELECT * FROM `users`
   WHERE `email` = 'admin@example.com'
      OR 1 = '1'
      AND `password` = 'abc123456'
```

Take Home Message

Use the language sanitization functions.
Do NOT build your own.

Prepared Statement

```
$stmt = $pdo->prepare(“SELECT * FROM users
WHERE email = ? AND password = ?”);

$stmt->execute(
   [$_POST['email'], $_POST['password']]
);
```

Library ensures input will always be just input.

No control allowed.