Part 11: Security: Access Rights in SQL

References:

- Elmasri/Navathe:Fundamentals of Database Systems, 3rd Edition, 1999. Chap. 22, "Database Security and Authorization"
- Silberschatz/Korth/Sudarshan: Database System Concepts, 3rd Edition, McGraw-Hill, 1999. Section 19.1, "Security and Integrity"
- Kemper/Eickler: Datenbanksysteme (in German), Ch. 12, Oldenbourg, 1997.
- Lipeck: Skript zur Vorlesung Datenbanksysteme (in German), Univ. Hannover, 1996.
- Date/Darwen: A Guide to the SQL Standard, Fourth Edition, Addison-Wesley, 1997.
- van der Lans: SQL, Der ISO-Standard (in German, there is an English version), Hanser, 1990.
- Oracle8 SQL Reference, Oracle Corporation, 1997, Part No. A58225-01.
- Oracle8 Concepts, Release 8.0, Oracle Corporation, 1997, Part No. A58227-01.
- Don Chamberlin: A Complete Guide to DB2 Universal Database. Morgan Kaufmann, 1998.
- Microsoft SQL Server Books Online: Accessing and Changing Data, Administering SQL Server.
- K. Nagel: Informationsbroschüre zum Bundesdatenschutzgesetz, ("Information about the German Federal Data Privacy Law", in German), 10. Auflage, Oldenbourg, 2001.

Objectives

After completing this chapter, you should be able to:

- Explain some possible attacks.
- Explain some DBMS features related to security.
- Use the `GRANT` and `REVOKE` commands of SQL.
- Enumerate some other security-related features of database management systems.

  In this text, Oracle, DB2, and SQL Server are discussed.

Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

DB Security: Motivation (1)

- Information is a valuable asset: E.g. customer data must be kept safe from competitors.
- There are laws regulating the privacy of certain information: One might be sued for allowing private information (e.g. medical records, credit card numbers) to become public.
- The confidentiality of certain information (e.g. salaries) must be protected within a company.

DB Security: Motivation (2)

- If an intruder deletes all data, a company will be out of business immediately.
- Unauthorized changes/falsification of data must be prohibited, e.g. employees should not be able to change their own salary.

  Certain business rules — who may do what — must be enforced.

Security Requirements (1)

It should be possible to

- ensure that only legitimate users have access to the database (user identification/authentication).
- define which user can perform which operations on which database objects (authorization).
- log the actions of the users (auditing).
- ensure the privacy and integrity of the data also in a networked (client-server) environment.

Security Requirements (2)

It should be possible to

- limit the use of resources (disk space, CPU time) for specific users (quotas).

  Or else a single user can bring the complete database to a standstill.
- manage groups of users with the same rights.
- ensure that nobody has direct access to the data (circumventing the database access control).
- separate responsibilities of different administrators.

User Authentication (1)

- This is normally done with secret passwords.
- Some databases perform their own user identification, some rely on the operating system.

  Both have advantages and disadvantages: It is convenient not to have to enter passwords twice, and it is also good if application programs do not have to contain passwords. However, in a client-server environment, it might be not so clear that the user on the client is really the person he/she claims to be (and that the client is the computer it claims to be). Some client operating systems might also have a weaker security model than the server (think e.g. of a PC where anybody can enter a boot disk/CD).

User Authentication (2)

- It is potentially dangerous that the same password is used again and again.

  If an intruder somehow observes the password, he/she can also use it. Some banks send their customers lists of passwords each of which can be used only once.

- It is possible to force users to change passwords periodically e.g. every month.

  This might result in weaker passwords than if the user has time to think about it. Systems forcing the user to change his/her password generally also require that the old and the new password differ significantly (e.g. it is not enough to change only one letter), and that the user cannot switch back to the old password too soon.

User Authentication (3)

- Never store a password for another computer (server) on a disk.

  It has happened that a public domain utility program emailed the password for an online service to the hacker. Even if the password is somehow encrypted, the computer of the hacker can be made to look like the computer on which the utility was run.
  Also think about HTTP cookies containing passwords.

- Choose a password which is not easy to guess.

  Even if direct decryption is not possible, a fast encryption routine can check many words for a match. If a hacker gets the encrypted password, he can try all words in a dictionary, names of persons/pop groups, all these words reversed, and all short words consisting only of lowercase letters or only of digits (phone numbers).

User Authentication (4)

- If a system discovers several unsuccessful login attempts, it should ring an alarm bell and possibly lock the account.

  In addition, there is often an artificial delay before the system tells the user that the password was incorrect. In this way a hacker program cannot try many words at full speed.

- Many DBMS have certain default passwords for the DBA. Set new passwords immediately!

  E.g. in Oracle, `SYSTEM` has the password `MANAGER`, and the password for `SYS` is `CHANGE_ON_INSTALL`. A hacker will know this. In SQL Server, the most powerful account `sa` (system administrator) has by default no password.

User Authentication (5)

- Remove guest accounts.

  Everybody knows that Oracle has an account `SCOTT` with password `TIGER`. SQL Server also has a `guest` account, to which NT users otherwise unknown to the database are mapped. Check the accounts in the system periodically and make sure that each is really needed.

- Since most DBMS are client-server systems, the database server is automatically "on the net" (like a web server).

  So even if the hacker cannot get an operating system account on the server machine, he/she might be able to connect to the DBMS server. Oracle normally waits on port 1521 for connections, and the default SID is ORCL.

User Authentication (6)

- If a password is sent without encryption over the internet, other people may be able to see it.
  - ◇ It is possible to listen to all packages sent over the local Ethernet.

    Ethernet packages are broadcast to all connected computers, but normally a low software layer in the operating system ignores packages for other computers.
  - ◇ The route a package takes over the global internet is not predictable.

    Some gateway might be operated by a bad guy or might already be conquered by a hacker.

User Authentication (7)

- When entering the password, be sure that you are connected with the right program/computer.

  E.g. there were earlier programs which looked like a UNIX login prompt, but instead sent the password to a hacker. Insufficient care with the search path for commands might result in a getting a hacker program when calling "`sqlplus`". On the internet, it is possible that computers "pretend" to be some other computer.
- Don't use the same password for several systems.

  Don't ever create accounts in web shops with the same password as your Oracle account. The shop personnel might be able to read it.
- Never tell a password to official sounding strangers on the phone.

Permissions/Privileges (1)

- Often different users of a DB have different rights.
- Commands can be understood as consisting of
  - ◇ a "subject" (the user who executes it),
  - ◇ a "verb" (the operation, e.g. "`INSERT`), and
  - ◇ an "object" (e.g. the "`COURSES`" table).
- DBMS allow defining which operations a user can apply to which objects (see `GRANT` below).

  So the DBMS stores a set of triples (u, o, d) stating that user u can perform operation o on database object d. Operations are basically `SELECT`, `INSERT`, `UPDATE`, `DELETE`, objects are e.g. tables.

Permissions/Privileges (2)

- However, there are at least two problems with the user-operation-object model:
  - ◇ Commands like "`CREATE TABLE`" do not refer to existing objects, but it could be beneficial to restrict their use.
  - ◇ Large companies might need different administrators with different rights (not a single "root" user who can do everything).
- Every major DBMS has some solution to these problems, but each solution is somewhat different.

Permissions/Privileges (3)

- Views and stored procedures provide a way to encapsulate database objects:
  - ◇ Although a user cannot `SELECT` from a table directly, he/she might be able to access certain columns/rows or summary data via a view.

    Also data changes can be limited via views, see below.
  - ◇ Although a user cannot do a direct `INSERT` into a table, he/she might use a procedure which performs such an insertion after additional tests.

Permissions/Privileges (4)

- There are two basic security models. Normal DBMS implement only the first:
  - ◇ "Discretionary Access Control" gives the administrators the ability to grant privileges to users at their discretion.

    I.e. they can assign access rights in whatever way they think is right.
  - ◇ "Mandatory Access Control" supports a classification of users and data, and allows a user to read data only at the same or lower security level as his/her clearance.

Auditing

- It should be possible to log user actions.

  Also unsuccessful attempts to execute commands (because of insufficient privileges) should be logged.
- In this way, it might at least afterwards be possible to find who is responsible for a problem.

  Or from which account a hacker has broken into the system.
  Of course, the auditing information itself must be sufficiently secure so that the hacker cannot delete it.
- The auditing must be done very selectively, or it will be difficult to find something interesting in too much data (and a lot of file space is needed).

Data Security

- Nobody should have direct access to the data (bypassing the database).

  At least nobody who has not also the DBA rights.
- The data is often stored non-encrypted in operating system files. Access to these files permits access to the data even without a DB account.
- Backup tapes must also be locked away.
- In Germany a PC containing an AIDS register was stolen. In the US, several notebooks from the CIA went missing.

Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

Data Privacy Law (1)

- This section explaines the German data privacy law.

  "Bundesdatenschutzgesetz". I am not a law expert, and the available space is not sufficient for explaining the entire law. You cannot make me or my university responsible for any error or missing information in my course materials.

- The purpose of the law is to protect the individuals from restrictions in their constitutional rights caused by using data about to them.

  The law covers data about single identifiable persons. It does not cover aggregated or anonymized data, or data used only for private purposes. It is basically intended for data on computers, but covers also certain other situations ("non-automized files").

Data Privacy Law (2)

- Storing data about persons is permitted only if
  - ◇ the affected person has agreed,
  - ◇ the data must be stored by law,
  - ◇ the data is necessary for fullfiling a contract (or something similar) with the affected person,
  - ◇ the data is necessary for the reasonable interest of the company/agency and there is no reason to believe that it might be in the greater interest of the affected person to forbid this, or
  - ◇ the data is publically available.

Data Privacy Law (3)

- Personal data must be collected for a specific purpose. The data may later be used for a different purpose only under certain conditions.

  (1) It is necessary for a reasonable interest of the company and there is no reason to believe that there might be a greater interest of the affected person to forbid this.
  (2) The data are publically available or may be published.
  (3) It is necessary for protecting a reasonable interest of a third person (this is restricted, e.g. if the data are about illegal activities).
  (4) It is necessary for the police, for protecting lifes, etc.
  (5) It is used for advertisements or market research and contains only name, address, year of birth, job, and membership in a certain group of persons, and there is no reason to believe that there is contrary interest of the affected person. One may explicitly exclude this use.
  (6) For research purposes under certain conditions.

Data Privacy Law (4)

- Persons must be informed when data is stored about them, unless they know it already.

  However, there are many exceptions:
  (1) The data must be stored by law.
  (2) The data are taken from a public source and the effort to inform all people would be too high.
  (3) The data must be kept secret by law or because of a more important interest of a third person.
  (4) The information would endanger the business purpose of the agency that stores the data, and the interest of the affected person to get this notice is not more important than this danger.
  (5) The data are stored for selling them for advertising or market research purposes, they are limited as explained above, and because of the number of persons it would be unreasonable to inform all.

Data Privacy Law (5)

Duty of Registration, Data Privacy Commissioner:

- Companies or agencies who process personal data must appoint a data privacy commissioner who ensures that the laws are fulfilled.

  ("Datenschutzbeauftragter") This does not apply to agencies with at most four employees that work with these data. However, it does apply to all companies who sell personal data.

- Processing of personal data must be registered with the state authorities/data privacy commissioner.

  This does not apply to the case of at most four employees. If the data is collected for being sold, the processing must always be registered with the state authorities.

Data Privacy Law (6)

Duty of Registration, Continued:

- The report must contain:
  - ◇ name and address of the company/agency,
  - ◇ the persons responsible for the data processing,
  - ◇ the purpose of the data processing,
  - ◇ which data about which persons will be stored,
  - ◇ to whom the data will be transmitted,

    Especially in which countries (applicable data privacy standards?).
  - ◇ after what time the data will be deleted,
  - ◇ information about data security measures.

Data Privacy Law (7)

Sensitive Data:

- If the data is especially sensitive, the data processing must be checked by the state authories or data privacy commissioner before it starts.

  Sensitive data is defined as data about race, ethical origin, political opinions, religious or philosophical convictions, membership in a work's union, health, or sexual life. In addition, any data processing that is used to evaluate the personality, performance, or behaviour of persons must first be checked.

- Sensitive data must be explicitly mentioned in any agreement to store one's personal data.

  There are special restrictions who may store sensitive data.

Data Privacy Law (8)

Exclusion of "Computer Only" Decisions:

- It is forbidden to use only computer programs for making decisions about people that have important negative consequences for them.

  At least, the affected people must be told about this and given the chance to explain their point of view, in which case the decision must be considered again. If a person asks, the basic logic of the automized decision must be explained.

Data Privacy Law (9)

Right of Information:

- Persons about which data is stored can ask the company/agency
  - ◇ what data is exactly stored about them,
  - ◇ what was the source for these data,
  - ◇ for which purpose the data is stored,
  - ◇ to whom the data was transmitted.
- This question usually must be answered (and in most cases without payment).

Data Privacy Law (10)

Right of Information, Continued:

- Of course, police and similar state authories are exceptions for the right of information.

  If a company sells the data, it might not have to answer the question to whom the data was transmitted because this might be a business secret. However, a judge may have to decide what is more important. Companies who sell data might be able to require a payment for the answer if the answer can be used for earning money (seems unusual). The required payment may never be more than the actual cost. If there is reason to assume that the stored data is incorrect, no payment can be required for the answer.

- However, the federal data privacy commissioner can be asked to check the data.

Data Privacy Law (11)

Right of Correction:

- Incorrect data must be corrected.

  If the affected person claims that the data is incorrect and it is not possible to determine whether the data is indeed correct or incorrect, the data must be blocked (can no longer be used, marked as deleted). This does not apply to companies who sell data but they must attach a statement of the affected person (counterrepresentation) to the data and are not allowed to transmit the data without this statement.

- If the data was already transmitted, the recipient must be informed.

  Unless the effort for this would be too big compared to the problem for the affected person.

Data Privacy Law (12)

Requirement of Deletion:

- Data must be deleted if they
  - ◇ were stored illegally,
  - ◇ are especially sensitive and their correctness cannot be proven,

    Especially sensitive data is defined in the law (see above). In this case it also contains information about illegal activities.
  - ◇ are no longer needed for the original purpose.
- Instead of deletion, it might suffice to block them.

  I.e. they are no longer normally accessible ("marked as deleted"), but they can still be recovered if required for purposes defined in the law.

Data Privacy Law (13)

Technical Requirements (for Preventing Misuse):

- Within reasonable limits, it must be ensured that
  - ◇ only valid users have access to the computers,

    and they can only work within the limits of their access rights,
  - ◇ it can be determined to whom (which companies/agencies) data is transferred,
  - ◇ data is protected during transfer (e.g. encrypted),
  - ◇ it is possible to determine who entered, modified, or deleted (?) the data,
  - ◇ data is protected against damage or loss.

Data Privacy Law (14)

Compensation for Damage:

- If a company/agency does somebody damage by illegal or incorrect processing of his/her data, it must pay a compensation for this damage.
- However, this holds only if the company/agency did violate a reasonable standard of carefulness.

  For public authorities, this exception does not apply, however the maximal payment is restricted.
- If the law is violated by purpose for money or in order to harm somebody, the maximum punishment are two years in prison.

Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

GRANT Command (1)

- In SQL, access rights on database objects (tables, views, etc.) can be given to other users by means of the `GRANT` command.
- The `GRANT` command was already contained in the SQL-86 standard. It has the form

  `GRANT` ⟨`Rights`⟩ `ON` ⟨`Object`⟩ `TO` ⟨`Users`⟩
- E.g. give read and insert rights ("privileges") on the table `COURSES` to the users `BRASS` and `SPRING`.

```
GRANT SELECT, INSERT ON COURSES TO BRASS, SPRING
```


GRANT Command (2)

- This will give the users `BRASS` and `SPRING` read access to the table `COURSES` and the possibility to append data (add new rows).
- They will not be able to delete or modify rows in the table (unless they had these rights before).
- It is possible to later `GRANT` them the `UPDATE` and `DELETE` rights, too.

  Then `SELECT` and `INSERT` do not have to be repeated.
- The DBMS probably stores in a system table the user-command-object triples.

GRANT Command (3)

Possible Rights ("Privileges") in SQL-92:

- `SELECT`: Read access (use of the table in queries).

  In SQL Server, `SELECT` rights can apply to specified columns. This is not part of the SQL-92 standard and not supported in Oracle and DB2. But views give the same effect.
- `INSERT`: Appending new data (insertion of rows).
- `INSERT`$(A_1, \ldots, A_n)$: Only values of the A_i may be specified, other columns will be filled with default values (declared in the `CREATE TABLE` command).

  Allowing `INSERT` only for specific columns is part of the SQL-92 standard, but only supported in Oracle (not in SQL Server and DB2). But views serve the same purpose.

GRANT Command (4)

SQL-92 Rights, continued:

- `UPDATE`: Changing column values of existing rows.
- `UPDATE`$(A_1, \ldots, A_n)$: Only data in the columns A_i may be changed.
- `DELETE`: Deleting data (rows from the table).
- `REFERENCES`: Creating integrity constraints which reference this table.

  Referencing someone else's table in a foreign key constraint and not giving him/her `DELETE` rights on the new table can in effect limit his/her `DELETE` rights on his/her own table. Also `REFERENCES` allows checking whether a key value is there or not.

GRANT Command (5)

SQL-92 Rights, continued:

- `REFERENCES`($A_1, \ldots, A_n$): Only the A_i may be referenced.

  This is interesting if the table has two or more keys. It is supported in Oracle and DB2 (not in SQL Server).

- In addition, SQL-92 has the right "`USAGE`" on domains, character sets, etc. (not supported in any of the three DBMS).

GRANT Command (6)

Non-Standard Rights for Tables:

- `ALTER`: Right to change the table definition.

  Supported in Oracle and DB2, not SQL Server.

- `INDEX`: Right to create an index on this table.

  Supported in Oracle and DB2, not SQL Server.

Non-Standard Rights for Procedures/Packages:

- `EXECUTE`: Right to execute the procedure etc.

  This is supported in all three DBMS.

- `BIND`: Reoptimize SQL statements in a package.

  This exists in DB2 only.

GRANT Command (7)

Non-Standard Rights for Schema Objects (DB2):

- `ALTERIN`: Alter any object in the schema.
- `CREATEIN`: Create objects in the schema.
- `DROPIN`: Drop any object in the schema.

Non-Standard Rights for Directory Objects (Oracle):

- `READ`: Read files in the directory.

GRANT Command (8)

`GRANT ALL PRIVILEGES`:

- Instead of listing single rights it is possible to say

  `GRANT ALL PRIVILEGES ON` ⟨`Object`⟩ `TO` ⟨`Users`⟩

- This means all rights which the user executing the `GRANT` has on the object.

`TO PUBLIC`:

- Instead of listing all users in the system, the following is possible (this includes future users):

  `GRANT SELECT ON COURSES TO PUBLIC`

GRANT Command (9)

`WITH GRANT OPTION`:

- A user can be given a right with the possibility to pass on the right to other users.
- To do this, add the clause "`WITH GRANT OPTION`" to the `GRANT`-command:

```
GRANT SELECT ON COURSES TO BRASS
WITH GRANT OPTION
```

- This allows the user `BRASS` also to pass the grant option to other users (who then can also pass the right to other users, etc.).

GRANT Command (10)

Owner of an Object:

- The owner of a database object (table etc.), i.e. the user who created the object, has all rights on it including the grant option for these rights.
- By default, only the owner has rights on an object. Other users can get rights only by explicit `GRANT`s.

  Users with system administrator rights might be able to access the object without being granted rights explicitly.

- The owner of an object can also drop the object again (which is not included in the other rights).

GRANT Command (11)

`CONTROL`-Right in DB2:

- This gives all rights on the object with grant option. It also gives the right to drop the object.

  There is no `CONTROL`-right in Oracle and SQL Server. But it is similar to being the owner of the object.

- By default, the object creator has the `CONTROL` right.

  For views, the creator gets the `CONTROL` right only if he/she holds it also for the underlying base tables (and used views).

- But the `CONTROL`-right is always given without `GRANT OPTION`. It can be granted only by an administrator.

  Granting `CONTROL` needs the `SYSADM` or `DBADM` authority.

Revoking Access Rights (1)

- Previously granted access rights can be revoked (taken back) with a command very similar to `GRANT`:

  `REVOKE` ⟨`Rights`⟩ `ON` ⟨`Object`⟩ `FROM` ⟨`Users`⟩

- ⟨`Rights`⟩: comma-separated list of single privileges or "`ALL PRIVILEGES`".
- ⟨`Object`⟩: name of database object (e.g. table, view).
- ⟨`Users`⟩: comma-separated list of user names or "`PUBLIC`".

Revoking Access Rights (2)

- Example:

  ```
  REVOKE INSERT ON COURSES FROM BRASS
  ```

  If BRASS had SELECT and INSERT rights on COURSES before this command, he will have only the SELECT right afterwards.

- Users can only revoke rights which they have granted earlier.

  Therefore, in its internal tables, the DBMS stores not only the triple user-right-object, but the quadruple (A, P, O, B): User A has granted privilege P on object O to user B.

Revoking Access Rights (3)

- It is not possible to grant a right to PUBLIC and revoke it from a specific user.

  Since PUBLIC also refers to future users, the database stores that it was granted to PUBLIC and not simply the right for every existing user.

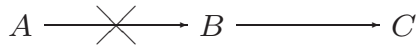
- It is possible to grant "ALL PRIVILEGES" and to revoke them later selectively.

  "ALL PRIVILEGES" refers only to the rights the user currently has. They are stored one by one in the system table.

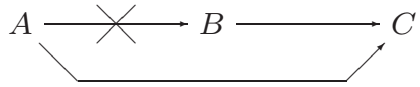
- When a table is deleted, all GRANTs for it are deleted, too. If it is later re-created, only the owner can access it.

Revoking Access Rights (4)

- If A granted a privilege "WITH GRANT OPTION" to B, and B granted it to C, and then A revokes the right from B, it will be recursively revoked from C.



- In SQL-92, this requires CASCADE, see below.
- However, if C got the right in addition on some other path (e.g. directly from A), he/she keeps it.



Revoking Access Rights (5)

- The REVOKE command restores the same situation as if B never had the right.
- But note that when B has used the right to change the tables, these changes are not magically undone.
- SQL-86 did not have a REVOKE command.
- In DB2, only the CONTROL right on a database object gives the possibility to REVOKE rights on it.

  It is a bit strange that the grant option allows users to grant the right, but not revoke it. In this way, DB2 does not have to keep track of the path a user got the right. If a user with CONTROL rights revokes it, it is gone (unless a group/public has the right).

Revoking Access Rights (6)

- In SQL-92 and SQL Server, `CASCADE` must be added if rights are to be revoked recursively, e.g.:

```
REVOKE INSERT ON Course FROM BRASS CASCADE
```

- In SQL Server, `CASCADE` is always required when revoking a right that was granted `WITH GRANT OPTION`.

  In SQL-92 this is only necessary if the user from which the right is revoked has used the grant option to grant the right to somebody else. Also, in SQL-92 `RESTRICT` can be specified instead of `CASCADE` to execute the `REVOKE` only if no other rights of other users depend on it. SQL Server does not understand `RESTRICT`.

- Oracle and DB2 do not support `CASCADE/RESTRICT`.

Revoking Access Rights (7)

- SQL-92 supports "`REVOKE GRANT OPTION FOR ...`".

  This is understood only in SQL Server, not in Oracle or DB2. SQL Server requires `CASCADE` is specified in addition.

- In Oracle, "`CASCADE CONSTRAINTS`" must be added to the `REVOKE` of the `REFERENCES` privilege if the privilege was used to create foreign keys. These foreign keys will then be dropped.

Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

Accessing Other Schemas (1)

- In Oracle, one must normally write "⟨`User`⟩.⟨`Name`⟩" to access database objects (tables, views, etc.) of other users, e.g.

```
SELECT * FROM BRASS.PRESIDENT
```

- Of course, this only works if the user `BRASS` has granted the `SELECT` privilege to the current user or to `PUBLIC`.
- If a user tries to access a table, but does not even have the `SELECT` right on it, Oracle gives the same error message as if the table did not exist.

Accessing Other Schemas (2)

- In Oracle, the user name is a DB schema identifier, so a database-wide unique identification of an object always consists of user name and object name.
- Oracle does not support schemas independently from users. If one wants to separate two sets of tables, one needs two Oracle accounts (user IDs).
- The guest account "SCOTT" with password "TIGER" can be used to check other user's rights.

Accessing Other Schemas (3)

- Since it is a bit difficult to write two-part names, Oracle supports user-defined abbreviations:

```
CREATE SYNONYM PRESIDENT FOR BRASS.PRESIDENT
```

- Such a synonym is valid only for the current user (who created the synonym).
- The DBA can also use

```
CREATE PUBLIC SYNONYM PRESIDENT
                FOR BRASS.PRESIDENT
```

  Then the table will look as if it exists under every account (but there is still only one copy).

Accessing Other Schemas (4)

- Such public synonyms are used for the data dictionary tables/views (system catalog).

  E.g. `CAT` is a synonym for `SYS.USER_CATALOG`.

- Even when a public synonym "X" is defined, users can still define their own table/views "X".

  Of course, then they cannot use the public synonym, but they still can access the table with "⟨`User`⟩.⟨`Table`⟩".

- Synonyms can be deleted with:

```
DROP SYNONYM PRESIDENT
```

- Synonyms are not part of the SQL-92 standard.

System Privileges (1)

- Besides access rights to tables etc. (called "object privileges"), Oracle also has "system privileges".
- These refer to the execution of specific commands, not to database objects.
- E.g. one needs the system privilege "`CREATE TABLE`" in order to be able to execute this command.

  A user who is only supposed to enter data does not need to create new tables. For a secure system, every user should have only the privileges he/she needs. I.e. the user should only be able to execute the commands he/she is supposed to execute. Object privileges alone could not restrict the use of the `CREATE TABLE` command.

System Privileges (2)

- In order to log into Oracle, one needs the system privilege "CREATE SESSION".

  An account can be locked by not granting (or revoking) this privilege. It is still possible to access tables, views, etc. under this account via synonyms or "⟨User⟩.⟨Table⟩" (if one has the necessary access rights).

- Many system privileges are only for DBAs, e.g.:
  - "SELECT ANY TABLE" (read access to all tables),
  - "DROP ANY TABLE" (delete data of arbitrary users),
  - "CREATE USER" (create a new user).

System Privileges (3)

- Since the usual privileges of a DBA are separated into different system privileges, it is possible to have several DBAs with different responsibilities.

  Of course, one can still have one DBA with all privileges.

- There are currently more than 90 different system privileges.

  Basically, every administration command corresponds to a system privilege. Different kinds of CREATE commands also correspond to system privileges (since these commands could not be restricted otherwise). Most commands also have an ANY-version as a system privilege (allows one to apply the command to objects of any user). CREATE ANY TABLE: create tables in any schema.

System Privileges (4)

- If a user has a system privilege "WITH ADMIN OPTION", he/she can give it to other users:

  GRANT CREATE TABLE TO SCOTT

  Adding "WITH ADMIN OPTION" gives SCOTT the right to grant "CREATE TABLE", too.

- When a system privilege is revoked from a user A who had it "WITH ADMIN OPTION", privileges are not recursively revoked from users B who got it from A.

  This might be the reason why it was not called "GRANT OPTION". But it is very similar ("GRANT OPTION" can be used only for object privileges).

Roles (1)

- It is difficult to grant privileges to many users one by one. In one way or another, all modern DBMS support groups of users with similar privileges.
- Oracle has the concept of "roles", which are sets of privileges that can be granted to users:

  CREATE ROLE ⟨Name⟩

- Only those with the system privilege "CREATE ROLE" can execute this command (e.g. only the DBA).

Roles (2)

- Access rights are granted to a role (e.g. "STAFF") in the same way as they are granted to a user:

```
GRANT SELECT ON COURSES TO STAFF;
```

- Granting access rights to a role is treated in the same way as granting it to specific users (DBA rights are not needed).
- Roles can be granted to users:

```
GRANT STAFF TO JIM, MARY
```


Roles (3)

- When a user A is granted a role R, A receives all privileges that were or will be granted to R.

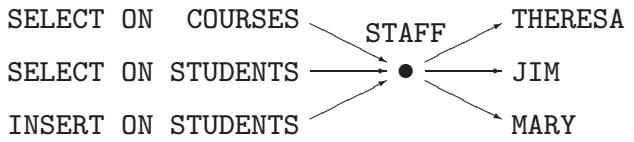
  But if "STAFF" is not one of the default roles of these users, which are automatically activated when they log in, they must explicity execute "SET ROLE STAFF" in every session in which they want to use these privileges. (It seems that this is not enforced in Oracle 8.0.)

- Only the owner of the role or a user who has received it with admin option can grant the role to another user.

```
GRANT STAFF TO THERESA WITH ADMIN OPTION
```


Roles (4)

- Roles are a level of indirection between privileges and users, intended to simplify the administration of a group of users with the same privileges:



- When further privileges are granted to "STAFF", these become automatically available to THERESA, JIM, and MARY.

Roles (5)

- A role can be granted to another role, e.g.

```
GRANT STAFF TO REG_OFFICE
```

- Then users with the role "REG_OFFICE" also have all the privileges granted to "STAFF".

  I.e. "REG_OFFICE" is more powerful, it implies staff.

- Roles can be protected by passwords. Then the SET ROLE command requires a password.

Roles (6)

- Several roles are predefined in Oracle 8, e.g.
  - ◇ `CONNECT`: Basic usage rights.

    This corresponds to the system privileges: `CREATE SESSION`, `ALTER SESSION`, `CREATE DATABASE LINK`, `CREATE SYNONYM`, `CREATE TABLE`, `CREATE CLUSTER`, `CREATE VIEW`, `CREATE SEQUENCE`.
  - ◇ `RESOURCE`: Rights for advanced users.

    This includes e.g. `CREATE TABLE`, `CREATE PROCEDURE`, `CREATE TRIGGER`. Students in this course were granted `CONNECT` and `RESOURCE` (but `UNLIMITED TABLESPACE` was revoked).
  - ◇ `DBA`: Right to do everything.
- In older Oracle versions, users were classified into these three types.

Creating Users (1)

User Authentication:

- Oracle can perform the user authentication itself. One must specify a user name and a password:

```
CREATE USER BRASS IDENTIFIED BY ABC_78
```

  Passwords have the same syntax as table names: They are not case-sensitive and `"..."` is needed to include special characters.
- Oracle can also rely on the authentication done by the operating system or a network service:

```
CREATE USER OPS$BRASS IDENTIFIED EXTERNALLY
```

  So when the UNIX user `BRASS` logs into Oracle (with empty username/password), he becomes the Oracle user `OPS$BRASS`.

Creating Users (2)

- A user created as explained above has no rights, not even the system privilege to connect to the DB.
- The necessary privileges can be given e.g. with:

```
GRANT CONNECT, RESOURCE TO BRASS
```

- After the `GRANT`, these roles can be made default roles, so that they are automatically activated when the user logs in:

```
ALTER USER BRASS DEFAULT ROLE ALL
```

  It seems that roles without a password automatically become default roles (?). So this command might not be necessary.

Tablespaces and Quotas (1)

- A tablespace is a database file or a collection of DB files (storage space, container for tables).
- All tablespaces are listed in the system catalog table `DBA_TABLESPACES`.

  E.g. use "`SELECT TABLESPACE_NAME FROM DBA_TABLESPACES`" to list all tablespaces. This query must be executed by a DBA. All users have read access to `USER_TABLESPACES` (tablespaces that are accessible by the current user). The files for each tablespace are listed in `DBA_DATA_FILES`. It has e.g. the columns `FILE_NAME`, `FILE_ID`, `TABLESPACE_NAME`, `BYTES`. See also `DBA_FREE_SPACE/USER_FREE_SPACE` and `DBA_FREE_SPACE_COALESCED`.
- The tablespace "`SYSTEM`" contains e.g. the data dictionary (collection of system tables).

Tablespaces and Quotas (2)

- `CREATE USER BRASS IDENTIFIED BY MY_PASSWORD`
  `DEFAULT TABLESPACE USER_DATA`
  `TEMPORARY TABLESPACE TEMPORARY_DATA`
  `QUOTA 2M ON USER_DATA`
  `QUOTA UNLIMITED ON TEMPORARY_DATA`
- A tablespace can be defined when a table is created.

  Otherwise it is stored in the user's `DEFAULT TABLESPACE` (which is `SYSTEM` if it is not set in the `CREATE USER`).
- Without quota (and "`UNLIMITED TABLESPACE`"), the user cannot create tables on the tablespace.

  Use: `REVOKE UNLIMITED TABLESPACE FROM BRASS`

Changing and Deleting Users

- If a user has forgotten his/her password:

  `ALTER USER BRASS IDENTIFIED BY NEW_PASSWORD`
- A user without tables can be deleted in this way:

  `DROP USER BRASS`
- To delete the user including all his/her data, use:

  `DROP USER BRASS CASCADE`
- The following command ensures that the user can no longer log in, but leaves his/her data untouched:

  `ALTER USER BRASS ACCOUNT LOCK`

Some Predefined Users (1)

- `SYS`: Owner of the system tables (data dictionary).

  Most powerful account. Default password: `CHANGE_ON_INSTALL`.
- `SYSTEM`: The default database administrator.

  For most administration tasks. Default password: `MANAGER`.
- `SCOTT`: Guest and demonstration account.

  Default password: `TIGER`. Sometimes there are additional accounts used in tutorials: `ADAMS`, `BLAKE`, `CLARK`, `JONES`.
- `OUTLN`: Schema contains information for optimizer.

  Default password: `OUTLN`.

Some Predefined Users (2)

- `DBSNMP`: Information for the "intelligent agent".

  It is used for remote administration via the "enterprise manager". Default password: `DBSNMP`.
- One should check the list of users in the system table `ALL_USERS` and lock all users that are currently not needed (or change their passwords).

  There is also a table `DBA_USERS` with more information. The list of users created during the installation can change with new versions. Also, when one installs additional software (e.g. the Oracle application manager), more accounts are created.
- Hackers know all the default passwords!

External Password File

- Whereas the above passwords are stored in the database (encrypted), there usually is an additional file that contains passwords of administrators who need e.g. to start up the database.

  When the database is not running, passwords stored in the database cannot be accessed. If you use `CONNECT INTERNAL` in the server manager (`svrmgrl`) or `CONNECT SYS AS SYSDBA`, the default password is `ORACLE`. Actually, the `SYS` password in the password file and in the database can be different. The password file is generated by the `orapwd` utility program. Later, every user granted `SYSDBA/SYSOPER` rights is also stored in the password file. Instead of using a password file, you can use OS authentication. This depends on the parameter `REMOTE_LOGIN_PASSWORDFILE`.

Other Security Features (1)

- The resource usage of DB users can be restricted by creating a "profile" for them. This defines e.g.
  - ◇ How many concurrent sessions the user can have (number of windows with DB applications).
  - ◇ After what idle time he/she is logged off.
  - ◇ How much CPU time and how many logical reads (disk accesses) is allowed per session/per call.
  - ◇ After what time a password must be changed.
  - ◇ Which function is used to check the password complexity.

Other Security Features (2)

- Oracle also has an `AUDIT` command for defining which user actions are logged in system tables, so that one can later find out who did what.
  - ◇ E.g. all insertions should be logged that were executed (not refused):

```
AUDIT INSERT ON SCOTT.EMP
BY SESSION WHENEVER SUCCESSFUL;
```

    "`BY SESSION`" means that only one record is written for an entire session that did this operation (default). Alternative: "`BY ACCESS`".
  - ◇ E.g. log all unsuccessful login attempts:

```
AUDIT CONNECT WHENEVER NOT SUCCESSFUL;
```


Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

Users in DB2 (1)

- DB2 uses the authentication mechanism of the underlying operating system (DB2 itself does not manage passwords).
- DB2 has something similar to the "system privileges" of Oracle, they are called "Database Authorities" and "Instance-Level Authorities" in DB2.
- A DB2 instance can manage several databases.

  Instance: server process. Database: Collection of DB schemas (and their table data). E.g. use "`CONNECT TO SAMPLE`" (a specific DB) after logging into DB2.

Users in DB2 (2)

- An operating system user can connect to a database if he/she has the "`CONNECT`" authority.
- E.g. an administrator can "create" a user in the database by issuing this command:

```
GRANT CONNECT ON DATABASE TO BRASS
```

- This right can also be granted to "`PUBLIC`", in which case every OS user can connect to the database.

Database Authorities in DB2

- `CONNECT`: Right to log into the database.
- `CREATETAB`: Right to create tables.

  No right is required for view creation.
- `BINDADD`: Right to create packages.

  I.e. to compile application programs with embedded SQL.
- `IMPLICIT_SCHEMA`: Create tables in new schemas.
- `CREATE_NOT_FENCED`: Extend DBMS (add functions).
- `DBADM`: Access and modify all DB objects, grant any object privilege or DB authority except `DBADM`.

Groups in DB2

- DB2 has no roles, but it understands the concept of groups of the underlying operating system.

  E.g. UNIX and Windows NT have groups of users.
- Privileges can be granted not only to users, but also to groups.

  Privileges granted to groups are not used when binding a package.
- So there are three ways a user might get a privilege:
  - ◇ It can be granted to this user personally,
  - ◇ to a group in which he/she is member,
  - ◇ or to `PUBLIC`.

Instance-Level Authorities

- The most powerful right in DB2 is the `SYSADM` authority.

  The account under which DB2 is installed plus all members of its group get `SYSADM` authority. It cannot be granted or revoked, but the group membership can be changed in the OS.

- There are also two other groups with fewer rights:
  - `SYSCTRL`: Can e.g. create or delete databases and tablespaces (plus all `SYSMAINT` privileges).
  - `SYSMAINT`: Can e.g. start or stop the database, do backups and restore the database after a crash.

Overview

1. Requirements
2. German Data Privacy Law
3. GRANT and REVOKE in SQL
4. Oracle
5. DB2
6. SQL Server

Creating New Users (1)

- SQL Server has two authentication mechanisms:
  - Windows NT Users or Groups can be mapped to SQL server logins.

    Then Windows NT does the authentication, there is no need to enter again a password. Users from trusted clients (which must run Windows NT) can also be mapped to SQL Server logins.
  - SQL Server Authentication (with password).

    In this case, SQL Server requires login name and password. This is the only possibility if SQL Server runs on Windows 95/98.
- One SQL Server Instance can manage several databases. The databases can have different users.

Creating New Users (2)

- It is necessary to distinguish between a login into the server, and the user in a specific database.

  Each login has a default DB to which it will be connected. The username in a DB may be different from the server login, and in different DBs different usernames can be used.

- Logins / users can be created with the Enterprise Manager (graphical interface).

  Alternatively, one can use the system procedures `sp_addlogin` (SQL Server authentication) and `sp_grantlogin` (Windows NT authentication) to create a login and `sp_grantdbaccess` to allow the login to access a specific database (also needed for the default DB of the login).

Special Users

- There is a login "sa" (system/server administrator). Everything can be done under this login.

  It uses SQL Server authentication (no password by default!).

- Every database has a user "dbo" (DB owner).

  Any member of the sysadmin sever role (e.g. "sa") is mapped to this user when he/she connects to a database.
  If a table is referenced without specifying a user, SQL Server first tries to find it in the account/schema of the current user, and then in the account/schema of "dbo". This eliminates the need for synonyms as used in Oracle.

- Some databases may have a "guest" user.

  A server login not mapped to a db user is mapped to guest.

Roles

- SQL Server has the concept of roles (user groups) which seems to be basically the same as in Oracle.

  In addition SQL Server also uses Windows NT groups.

- However, some roles are special and contain administration rights which cannot explicitly be granted.
- Fixed server roles allow administration of the server. The most powerful is "sysadmin" (e.g. "sa").
- Fixed database roles allow administration of a DB. The most powerful one is "db_owner" (e.g. "dbo").

System Privileges

- In addition, SQL Server has system privileges basically as in Oracle (managed via GRANT/REVOKE):
  - ◇ CREATE DATABASE
  - ◇ CREATE DEFAULT
  - ◇ CREATE PROCEDURE
  - ◇ CREATE RULE
  - ◇ CREATE TABLE
  - ◇ CREATE VIEW
  - ◇ BACKUP DATABASE
  - ◇ BACKUP LOG

Fixed Server Roles

- sysadmin: Perform any activity in SQL Server.
- securityadmin: Manage logins.
- serveradmin: Configure server-wide settings.
- setupadmin: Execute some system stored procedures, e.g. sp_serveroption.
- processadmin: Kill processes of other users.
- diskadmin: Manage the disk files.
- dbcreator: Create and alter databases.

Fixed Database Roles

- `db_owner`: Perform any activity in the database.
- `db_accessadmin`: Manage users of the database.
- `db_securityadmin`: Manage roles and permissions.
- `db_ddladmin`: Create, modify, drop any DB object.
- `db_backupoperator`: Make a backup copy of the DB.
- `db_datareader`: Read all tables in the database.
- `db_datawriter`: Modify all tables in the database.
- `denydatareader`, `denydatawriter`: These roles forbids to read/modify any table in the database.

DENY Statement

- SQL Server has a statement which is the opposite of `GRANT`, e.g.:

```
DENY SELECT ON COURSES TO BRASS
```

- The idea is that `BRASS` might be member of a group or role, which has the right, and you might want to declare `BRASS` as an exception.

  In an older version of SQL Server, `REVOKE` worked like `DENY` now. When an SQL92-conforming `REVOKE` was introduced, the old was called `DENY`.

- So the `DENY` on the user level takes precedence over the `GRANT` on the group/role/public level.