Part 13: Datenschutz: Zugriffsrechte in SQL

Literatur:

- 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.

Lernziele

Nach diesem Kapitel sollten Sie Folgendes können:

- Einige Vorgehensweisen von Hackern erklären.
- Sicherheitsrelevante DBMS Funktionen erklären.
- Die SQL Befehle `GRANT` und `REVOKE` benutzen.
- Einige Grundzüge des deutschen Datenschutzrechtes beim Aufbau von Datenbanken berücksichtigen.

  Oder zumindest erkennen, wann Sie weitere Informationen einholen müssen.

Inhalt

1. Anforderungen
2. Deutsches Datenschutzrecht
3. Die SQL-Befehle GRANT und REVOKE
4. Oracle
5. DB2
6. SQL Server

DB-Sicherheit: Motivation (1)

- Information: wichtiger Aktivposten von Firmen. Darf nicht in die Hände der Konkurrenz gelangen.

  Z.B. Kundendaten, Einkaufspreise, Produkt-Zusammensetzungen.

- Es gibt Gesetze zum Schutz vertraulicher Daten.

  Wenn man z.B. zuläßt, daß ein Hacker Zugriff auf Kreditkartennummern bekommt und diese Daten dann mißbraucht, können hohe Schadenersatzforderungen entstehen. Außerdem: schlechte Reputation.

- Die Vertraulichkeit bestimmter Daten muß auch innerhalb der Firma geschützt werden (z.B. Gehälter).

  Man muß auch verhindern, daß Angestellte einfach alle Daten mitnehmen können, wenn sie die Firma verlassen.

DB-Sicherheit: Motivation (2)

- Falls es einem Hacker gelingen sollte, alle Daten zu löschen, oder vollständig durcheinander zu bringen, entsteht ein gewaltiger Schaden für die Firma.

  Man sollte mindestens noch Sicherungskopien haben, auch offline, und an einem anderen Ort untergebracht (wegen einem möglichen Brand im Rechnerraum). Aber die Wiederherstellung dauert, und die letzten Änderungen sind schwer zu rekonstruieren.

- Unautorisierte Änderungen / Verfälschungen der Daten müssen verhindert werden, z.B. sollten Angestellte nicht ihr eigenes Gehalt ändern können.

  Bestimmte Geschäftsregeln — wer darf was tun — sollten durch das Informationssystem automatisch sichergestellt werden.

Sicherheits-Anforderungen (1)

Es sollte möglich sein ...

- sicher zu stellen, daß nur die tatsächlich legitimierten Benutzer Zugriff auf die Datenbank haben.

  → Benutzer Identifikation/Authentifikation

- festzulegen, welcher Nutzer welche Operationen auf welchen DB-Objekten durchführen darf.

  → Benutzer-Autorisierung

- die Aktionen der Benutzer zu protokollieren.

  → Auditing

Sicherheits-Anforderungen (2)

Es sollte auch möglich sein ...

- die Benutzung von Resourcen (Platten-Platz, CPU-Zeit) für jeden Nutzer zu begrenzen (Quotas).

  Sonst kann ein einzelner Nutzer die ganze Datenbank zum Stillstand bringen.

- Gruppen von Benutzern mit den gleichen Rechten zu verwalten.

  Umgekehrt kann ein Nutzer auch unterschiedliche Rollen haben.

- mehrere Administratoren mit unterschiedlichen Befugnissen zu haben (nicht nur einen einzelnen Nutzer "root", der alles darf).

Sicherheits-Anforderungen (3)

Es sollte auch möglich sein ...

- die Vertraulichkeit und Integrität der Daten auch in einer Netzwerk-Umgebung zu garantieren.

  Oft geschieht die Client-Server Kommunikation unverschlüsselt.

- sicher zu stellen, daß niemand direkten Zugriff auf die Daten hat (und damit die Zugriffskontrolle der Datenbank umgeht).

- darauf zu vertrauen, daß nicht durch Programmierfehler im DBMS Sicherheitslücken bestehen (oder sie wenigstens schnell korrigiert werden).

Benutzer-Authentifikation (1)

- Üblich ist, daß sich Benutzer mit Passworten gegenüber dem System ausweisen.
- Zum Teil führen die DBMS selbst eine Benutzer-Identifikation durch, zum Teil verlassen sie sich auf das Betriebssystem.

  Beide Möglichkeiten haben Vor- und Nachteile: Es ist bequem, nicht mehrfach Passworte eingeben zu müssen, und es ist besser, wenn Anwendungsprogramme keine Passworte enthalten. Aber in einer Client-Server-Umgebung kann man nicht so sicher sein, daß der Benutzer auf dem Client tatsächlich die Person ist, die er/sie behauptet, zu sein (und daß der Computer wirklich der richtige Computer ist). Manche Client-Betriebssysteme haben ein schwächeres Sicherheitssystem als der Server (bei manchen PCs kann jeder eine Boot-CD einlegen).

Benutzer-Authentifikation (2)

- Es ist gefährlich, wenn des gleiche Passwort immer wieder genutzt wird.

  Wenn ein Hacker das Passwort irgendwie beobachten kann, kann er es auch nutzen. Banken senden ihren Kunden typischerweise Listen mit Einmal-Passworten (TANs).
- Einige Administratoren zwingen die Nutzer, ihre Passworte in regelmäßigen Abständen zu ändern.

  Z.B. jeden Monat. Dies kann aber zu schwächeren Passworten führen, (und zu häufigeren Problemen mit vergessenen Passworten), als wenn der Benutzer ein neues Passwort nur wählt, wenn er dazu bereit ist. Wenn ein System den Benutzer zwingt, sein Passwort zu ändern, prüft es üblicherweise auch, daß neues und altes Passwort sich deutlich unterscheiden, und daß der Benutzer nicht zu schnell zurück wechselt.

Benutzer-Authentifikation (3)

- Trojaner sind Programme, die neben der Hauptfunktion, für die sie eingesetzt werden, im Verborgenen noch andere (schlechte) Dinge tun.
- Speichern Sie nie ein Passwort für einen anderen Computer auf der Festplatte Ihres Computers. Das gilt auch für Cookies, die Passworte ersetzen.

  Wenn Sie sich auf diese Art bei dem anderen Rechner einloggen können, ohne ein Passwort einzugeben, kann das auch jemand, der sich (z.B. mit Trojaner) die entsprechenden Daten von Ihrem System besorgt. Wenn das Passwort in einer Datei steht, ist es besonders einfach, und betrifft auch Passworte, die Sie nicht nutzen, wenn der Trojaner aktiv ist. Je nach Betriebssystem könnte er aber auch Tastendrücke mitloggen oder den Hauptspeicher anschauen.

Benutzer-Authentifikation (4)

- Passworte sollten nicht leicht zu raten sein.

  Der Name Ihrer Freundin/Ihres Freundes, Ihrer Lieblings-Popgruppe (Autor, Schauspieler, Film, Urlaubsland, etc.), Ihre Telefonnummer, Adresse, Geburtsdatum wären besonders leicht. Falls es einem Hacker gelingt, an eine verschlüsselte Version Ihres Passwortes zu kommen, kann er mit einer schnellen Verschlüsselungsroutine viele Worte probieren (selbst wenn eine direkte Entschlüsselung nicht möglich ist). Ein Hacker würde in so einem Fall den Duden durchprobieren, alle Namen von Personen, Popgruppen, relativ lange Folgen von Ziffern, zumindest alle kurzen Folgen von Kleinbuchstaben, etc. Außerdem alles auch rückwärts und auf verschiedene andere Arten leicht verändert. Deswegen wird empfohlen, daß ein gutes Passwort nicht zu kurz sein sollte, und Buchstaben, Ziffern, und Sonderzeichen enthalten sollte. Um eine möglichst zufällige Buchstabenfolge zu bekommen, kann man sich einen Satz denken und davon die Anfangsbuchstaben nehmen.

Benutzer-Authentifikation (5)

- Wenn ein System mehrere Anmeldeversuche mit falschem Passwort entdeckt, sollte es einen Alarm auslösen und eventuell den Account sperren.

  Außerdem gibt es oft eine künstliche Verzögerung, bevor das Programm dem Benutzer mitteilt, daß die Anmeldedaten falsch waren. So kann ein Hackerprogramm nicht viele Worte in kurzer Zeit probieren. Das Sperren des Accounts bestraft aber den korrekten Nutzer.

- Viele DBMS haben Default-Passworte für den Administrator. Diese müssen sofort geändert werden.

  Z.B. hat in Oracle der automatisch angelegte Administrator-Account `SYSTEM` das Passwort `MANAGER`, und das Passwort für `SYS` (kann alles) ist `CHANGE_ON_INSTALL`. Jeder Hacker weiß das. In SQL Server hat der mächtigste Account `sa` direkt nach der Installation kein Passwort.

Benutzer-Authentifikation (6)

- Man sollte Gast-Accounts löschen oder sperren.

  Oracle hat einen Account `SCOTT` mit Passwort `TIGER`. SQL Server hat einen Account `guest`, für Windows-Nutzer, die der Datenbank unbekannt sind. Prüfen Sie die Accounts im System in regelmäßigen Abständen und sperren Sie alle, die nicht wirklich benötigt werden.

- Das die meisten DBMS Client-Server Systeme sind, ist der DB-Server automatisch am Netz, sobald Ihr Rechner mit dem Internet verbunden ist.

  In dieser Hinsicht verhält sich der DB-Server ähnlich wie ein Web-Server. Selbst wenn der Hacker sich nicht beim Betriebssystem anmelden kann, kann er sich vielleicht bei der Datenbank anmelden. Oracle wartet normalerweise auf Port 1521 auf Verbindungen, die Default-SID ist "`ORCL`".

Benutzer-Authentifikation (7)

- Wird ein Passwort unverschlüsselt über das Internet verschickt, können es Hacker eventuell lesen.
  - ◇ Beim klassischen Ethernet kann man alle Pakete mitlesen, die über das Netz geschickt werden.

    Mit modernen Switches ist das nicht mehr möglich, aber erst, nachdem der Switch die Adressen der angeschlossenen Rechner gelernt hat. Man kann sich also nicht darauf verlassen, daß andere Rechner, die an das lokale Netz angeschlossen sind, die Datenpakete nicht auch erhalten.
  - ◇ Die Route, die Datenpakete über das globale Internet nehmen, ist kaum vorhersehbar.

    Ein Gateway wird vielleicht von einem Hacker betrieben, oder ist schon von einem Hacker "geknackt".

Benutzer-Authentifikation (8)

- Wenn Sie ein Passwort eingeben, achten Sie darauf, daß Sie auch mit dem richtigen Programm bzw. dem richtigen Computer verbunden sind.

  Früher gab es Programme, die wie ein UNIX Login Prompt aussahen, aber das Passwort an einen Hacker schickten.

  Der Suchpfad für Kommandos darf nur vertrauenswürdige Verzeichnisse enthalten (z.B. nicht "."). Sonst bekommt man vielleicht ein ganz anderes Programm, wenn man z.B. "sqlplus" aufruft.

  Es gibt viele "Phishing" ("password fishing") EMails, die z.B. behaupten, von der eigenen Bank zu kommen, und zur Eingabe von PIN und TAN auffordern. Tatsächlich kommt man mit der URL aber auf eine Hacker-Webseite, die wie die echte Webseite der Bank aussieht.

Benutzer-Authentifikation (9)

- Man sollte möglichst nicht das gleiche Passwort für verschiedene Systeme (auch Webshops) nutzen.

  Angestellte des Webshops können Ihr Passwort möglicherweise im Klartext lesen.

- Sagen Sie Ihr Passwort (auch PIN etc.) nie offiziell klingenden Unbekannten am Telefon!

  Z.B. dem technischen Support des Rechenzentrums, das gerade auf LDAP umstellt, und Ihnen die Mühe ersparen möchte, ins Rechenzentrum zu kommen, und dort Ihr Passwort noch einmal einzutippen.

Zugriffsrechte (1)

- Oft haben verschiedene Nutzer einer Datenbank auch verschiedene Rechte.
- Kommandos bestehen aus
  - ◇ "Subject" (der Benutzer, der es ausführt),
  - ◇ "Verb" (die Operation, z.B. "`INSERT`), und
  - ◇ "Objekt" (typischerweise eine Tabelle).
- In SQL kann man mit dem `GRANT`-Kommando (s.u.) definieren, welche dieser Tripel erlaubt sind.

  Das DBMS speichert also eine Menge von Tripeln (u, o, d), aus Benutzer u, Operation o (im wesentlichen `SELECT`, `INSERT`, `UPDATE`, `DELETE`), Objekt/Tabelle o. Eigentlich Quadrupel: Wer hat das Recht erteilt?

Zugriffsrechte (2)

- Das Subjekt-Verb-Objekt Modell für die Zugriffsrechte hat aber Einschränkungen:
  - ◇ Kommandos wie "`CREATE TABLE`" beziehen sich nicht auf existierende Objekte, aber man muß ihre Benutzung einschränken können.

    Prinzip: Jeder sollte nur das tun können, was er tun muß.
  - ◇ Für DB-Administratoren sollen die normalen Zugriffsbeschränkungen nicht gelten. Es soll aber auch nicht jeder Administrator alles können.

    Große Firmen haben mehrere Administratoren.

Zugriffsrechte (3)

- Während das Subjekt-Verb-Objekt Modell sehr portabel ist (schon im SQL-86 Standard enthalten), sind die Erweiterungen zur Lösung der obigen Probleme meist vorhanden, aber sehr systemabhängig:
  - ◇ Oracle hat neben Objektrechten nach dem obigen Modell noch eine große Anzahl von Systemrechten (wer darf welches Kommando/Verb benutzen, ggf. auch für beliebige Objekte).
  - ◇ Andere Systeme unterscheiden oft nur wenige Benutzerklassen, die verschieden privilegiert sind.

Zugriffsrechte (4)

- Sichten und serverseitige Prozeduren ("stored procedures") erlauben es, Objekte zu verkapseln:
  - ◇ Z.B. ist es möglich, daß ein Benutzer für eine Tabelle selbst kein `SELECT`-Recht hat, aber über eine Sicht bestimmte Zeilen/Spalten oder aggregierte Daten sehen darf.
  - ◇ Es ist auch möglich, daß ein Benutzer kein direktes `INSERT`-Recht für eine Tabelle hat, aber eine Prozedur benutzen kann, die Daten in diese Tabelle nach zusätzlichen Prüfungen einfügt.

Sicherheitsmodelle

- Es gibt zwei grundlegend verschiedene Sicherheitsmodelle. Normale Datenbanksysteme implementieren nur das erste:
  - ◇ "Discretionary Access Control" erlaubt den Administratoren, die Zugriffsrechte nach Belieben zu vergeben.
  - ◇ "Mandatory Access Control" basiert auf einer Klassifizierung von Benutzern und Daten.

    Z.B. "confidential", "secret", "top secret". Benutzer können nur Daten auf der eigenen oder einer niedrigeren Sicherheitsebene lesen, und nur Daten auf der eigenen oder einer höheren Sicherheitsebene schreiben.

Auditing

- In manchen Systemen kann man die Aktionen der Benutzer mitprotokollieren lassen.

  Das gibt typischerweise Ärger mit dem Betriebsrat, eventuell auch mit dem Datenschutz. Man beachte, daß nicht nur erfolgreich ausgeführte Kommandos protokolliert werden müssen, sondern auch Kommandos, die z.B. wegen nicht ausreichender Zugriffsrechte abgelehnt wurden.
- So kann man wenigstens hinterher herausfinden, wer für ein Problem verantwortlich ist.
- Eventuell fallen bei einer Kontrolle auch merkwürdige Muster auf, bevor wirklich ein Schaden eintritt.

  Man muß sehr selektiv mitprotokollieren, wenn man im Protokoll wirklich noch etwas Interessantes finden will.

Datensicherheit

- Niemand (der nicht ohnehin DBA Rechte hat) sollte direkten Zugriff auf die Daten haben (und damit die Zugriffskontrolle der Datenbank umgehen).

  Aus Performance-Gründen werden die Daten normalerweise unverschlüsselt in Betriebssystem-Dateien gespeichert. Wer auf diese Dateien Zugriff hat, kann auch ohne DB-Account die Daten darin lesen.
- Backup Bänder müssen weggeschlossen werden.
- In Deutschland wurde ein PC gestohlen, der ein AIDS-Register enthielt. In den USA sind mehrere Notebooks der CIA mit geheimen Unterlagen verschwunden.

Inhalt

1. Anforderungen
2. Deutsches Datenschutzrecht
3. Die SQL-Befehle GRANT und REVOKE
4. Oracle
5. DB2
6. SQL Server

Datenschutzrecht (1)

- In diesem Abschnitt sollen die Vorschriften des Bundesdatenschutzgesetzes erklärt werden.

  Ich bin kein Jurist, und der zur Verfügung stehende Platz ist nicht ausreichend für eine vollständige Erklärung. Alle Angaben sind ohne Gewähr. Wenn es wirklich wichtig ist, informieren Sie sich bitte bei einem Juristen.

- Das Gesetz soll davor schützen, daß man durch Umgang mit seinen personenbezogenen Daten in seinem verfassungsmäßig garantierten Persönlichkeitsrecht beeinträchtigt wird (freie Selbstbestimmung bei der Entfaltung der Persönlichkeit).

Datenschutzrecht (2)

- Es soll vermieden werden, daß "Bürger nicht mehr wissen können, wer was wann bei welcher Gelegenheit über sie weiß."
- Man soll normalerweise selbst über die Preisgabe und Verwendung seiner personenbezogenen Daten bestimmen können ("Informationelle Selbstbestimmung").
- Personenbezogene Daten sind "Einzelangaben über persönliche oder sächliche Verhältnisse einer bestimmten oder bestimmbaren natürlichen Person".

Datenschutzrecht (3)

- Das Gesetz bezieht sich nicht auf aggregierte oder anonymisierte Daten.
- Ebenfalls ausgenommen sind Daten, die ausschließlich für persönliche oder familiäre Tätigkeiten erhoben/verarbeitet werden.
- Das Gesetz gilt auch für "nicht-automatisierte Dateien".

  Es ist also nicht richtig, daß das Gesetz grundsätzlich erst dann anwendbar ist, wenn die Daten in einem Computer gespeichert werden. Die Daten müssen aber gleichartig aufgebaut sein und nach bestimmten Merkmalen zugänglich sein (Kartei).

Datenschutzrecht (4)

Besonders schützungswerte, sensitive Daten:

- "Besondere Arten personenbezogener Daten" sind:
  - ◇ rassische und ethnische Herkunft
  - ◇ politische Meinungen
  - ◇ religiöse oder philosophische Überzeugungen
  - ◇ Gewerkschaftszugehörigkeit
  - ◇ Gesundheit
  - ◇ Sexualleben
- Für diese Daten gelten teilweise verschärfte Vorschriften (s.u.). Sie müssen in jeder Einverständniserklärung explizit genannt werden.

Datenschutzrecht (5)

Verbot mit Ausnahmen:

- Personenbezogene Daten dürfen nur abgespeichert werden, wenn
  - ◇ der Betroffene zugestimmt hat,
  - ◇ die Daten aufgrund eines Gesetzes gespeichert werden müssen,
  - ◇ die Daten zur Erfüllung eines Vertrages (o.ä.) mit dem Betroffenen gespeichert werden müssen,
  - ◇ . . . (Fortsetzung siehe nächste Folie)

Datenschutzrecht (6)

Verbot mit Ausnahmen, Forts.:

- Zulässige Gründe für die Speicherung personenbezogender Daten, Forts.:
  - ◇ wenn es zur Wahrung berechtigter Interessen der Firma nötig ist, und es keinen Grund zur Annahme gibt, daß schutzwürdige Interessen des Betroffenen überwiegen,
  - ◇ die Daten allgemein zugänglich sind oder veröffentlicht werden dürften.

Sofern keine schutzwürdigen Interessen entgegenstehen.

Datenschutzrecht (7)

Zweckbindung der Daten:

- Daten müssen für einen speziellen Zweck gesammelt werden und dürfen später nur unter bestimmten Voraussetzungen für andere Zwecke verwendet werden:
  - ◇ Wahrung berechtigter Interessen (wie oben)
  - ◇ Daten allgemein zugänglich (wie oben)
  - ◇ Wahrung berechtigter Interessen eines Dritten,

Mit Einschränkungen, z.B. bei strafbaren Handlungen, oder Angaben des Arbeitgebers auf arbeitsrechtliche Verhältnisse.

Datenschutzrecht (8)

Zweckbindung der Daten, Forts.:

- Gründe für Nutzung von Daten zu anderen Zwecken als bei der ursprünglichen Erhebung, Forts.:
  - ◇ Abwehr von Gefahren für die staatliche Sicherheit, Verfolgung von Straftaten
  - ◇ Werbung/Marktforschung mit Einschränkungen

    Wenn es sich eine Liste von Angehörigen einer Personengruppe handelt, die nur die Zugehörigkeit zu der Gruppe, Berufsbezeichnung, Namen, akademische Titel, Anschrift und Geburtsjahr enthält, und kein Grund zur Annahme besteht, daß der Betroffene ein schutzwürdiges Interesse hat, das dem entgegensteht.
  - ◇ Für Forschungszwecke (unter Bedingungen).

Datenschutzrecht (9)

Benachrichtigungspflicht:

- Personen müssen darüber informiert werden,
  - ◇ daß Daten über sie gespeichert werden,
  - ◇ welche Arten von Daten das sind,
  - ◇ was der Zweck der Datenverarbeitung ist,
  - ◇ wer die Daten speichert (Stelle/Firma)

  (falls sie es nicht ohnehin schon wissen).
- Es gibt aber viele Ausnahmen von der Benachrichtigungspflicht, siehe nächste Folie.

Datenschutzrecht (10)

Ausnahmen von der Benachrichtigungspflicht:

- Daten müssen per Gesetz gespeichert werden
- Daten entstammen einer öffentlichen Quelle, unverhältnismäßiger Aufwand
- Daten müssen geheim gehalten werden (Gesetz oder überwiegendes Interesse eines Dritten)
- Würde Geschäftszweck erheblich gefährden (und kein überwiegendes Interesse des Betroffenen)
- Werbezwecke mit eingeschränkten Daten (s.o.), unverhältnismäßiger Aufwand.

Datenschutzrecht (11)

Datenschutzbeauftragter:

- Firmen oder öffentliche Stellen, die personenbezogende Daten verarbeiten, müssen einen Datenschutzbeauftragten bestellen.

  Bei Firmen gilt das nur, wenn mindestens 10 Personen ständig mit der Verarbeitung dieser Daten beschäftigt sind. Handelt es sich aber um besonders schützenswerte Daten, oder handelt die Firma mit den Daten, ist ein Datenschutzbeauftragter unbedingt nötig.
- Der Datenschutzbeauftragte muß die erforderliche Fachkunde und Zuverlässigkeit besitzen.
- Er ist in Ausübung dieser Tätigkeit weisungsfrei.

Datenschutzrecht (12)

Datenschutzbeauftragter, Forts.:

- Der Datenschutzbeauftragte
  - ◇ überwacht die ordnungsgemäße Anwendung der Datenverarbeitungsprogramme,
  - ◇ schult die Benutzer in den Vorschriften des Datenschutzes,
  - ◇ ist rechtzeitig über Vorhaben zur Verarbeitung personenbezogener Daten zu informieren.
- Es gibt auch externe Berater als Datenschutzbeauftragte.

Datenschutzrecht (13)

Meldepflicht:

- Die Verarbeitung personenbezogener Daten ist den zuständigen Aufsichtsbehörden zu melden, sofern man keinen Datenschutzbeauftragten hat.

  Die Meldepflicht entfällt aber, wenn unter 10 Personen mit der Verarbeitung der Daten beschäftigt sind, und die Betroffenen ihr Einverständnis erklärt haben, oder es der Abwicklung eines Vertrages oder ähnlichen Vertrauensverhältnisses dient.
- Hat die Firma einen Datenschutzbeauftragten, muß der natürlich entsprechend informiert werden.
- Falls man mit den Daten handelt, gilt die Meldepflicht an die Behörden auf jeden Fall.

Datenschutzrecht (14)

Inhalte der Meldepflicht:

- Verantwortliche Stelle (Firma, DV-Leiter)
- Zweck der Datenerhebung und -Verarbeitung
- Betroffene Personengruppen
- Daten oder Datenkategorien
- Empfänger, denen Daten mitgeteilt werden könnten
- Regelfristen für die Löschung der Daten
- geplante Datenübermittlung in Drittstaaten
- Maßnahmen zur Datensicherheit

Datenschutzrecht (15)

Vorabkontrolle:

- Während normalerweise diese Meldung ausreicht, ist bei besonders kritischen Fällen eine Vorabkontrolle notwendig:
  - ◇ bei besonders sensitiven Daten (Folie 13-29)
  - ◇ wenn die Verarbeitung der Daten dazu bestimmt ist, Persönlichkeit, Fähigkeiten, Leistung des Betroffenen zu bewerten.
- Zuständig ist der Datenschutzbeauftragte

  Kann sich in Zweifelsfällen an die Behörde wenden.

Datenschutzrecht (16)

Verbot von rein automatischen Entscheidungen:

- Es ist verboten, Entscheidungen über Menschen, die besonders negative Konsequenzen für sie haben können, rein automatisch (durch ein Computerprogramm) zu fällen.

  Mindestens müssen die Betroffenen darüber informiert werden, und die Möglichkeit bekommen, ihren Standpunkt darzustellen. Der Fall muß dann anschließend erneut betrachtet werden (durch einen Menschen).

- Wenn der Betroffene fragt, muß die grundlegende Logik der Entscheidung erklärt werden.

Datenschutzrecht (17)

Recht auf Information:

- Personen, über die Daten gespeichert werden, können die speichernde Stelle fragen,
  - ◇ welche Daten genau über sie gespeichert werden,
  - ◇ was die Quelle für diese Daten war,
  - ◇ zu welchem Zweck die Daten gespeichert sind,
  - ◇ an wen die Daten übermittelt wurden.
- Normalerweise müssen diese Fragen beantwortet werden, und in den meisten Fällen ohne Gebühr.

Datenschutzrecht (18)

Recht auf Information, Forts.:

- Die Polizei und ähnliche Behörden sind vom Recht auf Information natürlich ausgenommen.
- Man kann aber die Bundesdatenschutzbeauftragten bitten, die Daten zu prüfen.

  Weitere Ausnahmen: Wenn eine Firma mit Daten handelt, braucht sie die Frage, an wen die Daten übermittelt wurden, nicht zu beantworten (Geschäftsgeheimnis). Ggf. hat ein Richter zu entscheiden, was wichtiger ist. Firmen, die mit Daten handeln, können eine Gebühr für die Beantwortung der Fragen verlangen, wenn mit der Antwort Geld verdient werden kann. Die Gebühr kann aber niemals höher als die tatsächlichen Kosten sein. Gibt es Grund zur Annahme, daß die gespeicherten Daten inkorrekt sind, kann keine Gebühr verlangt werden.

Datenschutzrecht (19)

Recht auf Korrektur:

- Falsche Daten müssen berichtigt werden.

  Wenn die betroffene Person behauptet, die Daten seien falsch, und es läßt sich nicht feststellen, ob die Daten tatsächlich richtig oder falsch sind, müssen die Daten blockiert werden (können nicht mehr benutzt werden, sind als gelöscht markiert). Dies gilt nicht für Firmen, die mit Daten handeln, aber sie müssen den Daten eine Gegendarstellung der betroffenen Person beifügen und dürfen die Daten nicht ohne diese Gegendarstellung weitergeben.

- Falls die falschen Daten bereits weitergegeben wurden, muß der Empfänger informiert werden.

  Es sei denn, der Aufwand dafür wäre unverhältnismäßig hoch im Vergleich zum Schaden für die betroffene Person.

Datenschutzrecht (20)

Pflicht zur Löschung:

- Daten müssen gelöscht werden, wenn sie
  - ◇ illegal gespeichtert wurden,
  - ◇ besonders sensitiv sind, und nicht bewiesen werden kann, daß sie korrekt sind,

    Siehe Folie 13-29, hier auch Information über illegale Aktivitäten.
  - ◇ nicht mehr für den ursprünglichen Zweck nötig sind.
- U.u. reicht es, sie als gelöscht zu markieren.

  Sie dürfen dann nicht mehr zugreifbar sein, können aber für im Gesetz beschriebene Zwecke wiederhergestellt werden.

Datenschutzrecht (21)

Technische Anforderungen:

- Um Mißbrauch der Daten zu verhindern, ist (in vernüftigen Grenzen) sicherzustellen, daß
  - ◇ nur zulässige Nutzer Zugriff auf die Rechner haben (Schutz vor Hackern),
  - ◇ diese Nutzer nur in den Grenzen ihrer Zugriffsrechte damit arbeiten können,
  - ◇ festgestellt werden kann, an wen (welche Firmen oder Behörden) Daten übermittelt wurden,

Datenschutzrecht (22)

Technische Anforderungen, Forts.:

- Es ist weiter sicherzustellen, daß
  - ◇ Daten bei der Übermittlung geschützt sind,

    Z.B. sollten Daten nur verschlüsselt über öffentliche Netze übertragen werden.
  - ◇ es möglich ist, festzustellen, wer die Daten eingegeben, geändert, gelöscht (?) hat,
  - ◇ die Daten vor Beschädigung/Verlust geschützt sind.

    Wenn z.B. festgehalten werden muß, an wen Daten übermittelt wurden, darf die Firma nicht einfach sagen können, "leider ist uns die Platte kaputt gegangen, auf der diese Daten standen."

Datenschutzrecht (23)

Schadenersatz/Strafe:

- Wenn eine Firma/Behörde jemanden durch illegale oder inkorrekte Verarbeitung seiner/ihrer Daten schädigt, muß sie Schadenersatz zahlen.
- Dies gilt aber nicht, wenn sie einen angemessenen Standard an Sorgfalt erfüllt hat.

  Für Behörden gilt diese Einschränkung nicht, dafür ist die maximale Entschädigung begrenzt.
- Wenn dieses Gesetz vorsätzlich verletzt wurde, um damit Geld zu verdienen oder jemanden zu schädigen, ist die Maximalstrafe 2 Jahre Gefängnis.

Inhalt

1. Anforderungen
2. Deutsches Datenschutzrecht
3. Die SQL-Befehle GRANT und REVOKE
4. Oracle
5. DB2
6. SQL Server

GRANT Kommando (1)

- In SQL können Zugriffsrechte für Datenbankobjekte (Tabellen, Sichten, etc.) an andere Benutzer mit dem `GRANT` Kommando gegeben werden.
- `GRANT` war schon im SQL-86 Standard enthalten. Es hat die Form

  `GRANT ⟨Rechte⟩ ON ⟨Objekt⟩ TO ⟨Nutzer⟩`
- Das DBMS speichert diese Tripel im Systemkatalog ab, und prüft bei jedem Zugriff, ob der aktuelle Nutzer das entsprechende Recht hat.

  Es wird noch zusätzlich vermerkt, wer das Recht vergeben hat.

GRANT Kommando (2)

- Angenommen, die Tabelle `AUFGABEN` gehört dem Benutzer `BRASS`.
- Er kann Lese- und Einfüge-Rechte ("privileges") für diese Tabelle den Benutzern `MEIER` und `JUNG` mit folgendem Kommando geben:

```
GRANT SELECT, INSERT ON AUFGABEN TO MEIER, JUNG
```

- Anschließend können `MEIER` und `JUNG` z.B.:
  - `SELECT * FROM BRASS.AUFGABEN`
  - `INSERT INTO BRASS.AUFGABEN VALUES (...)`

GRANT Kommando (3)

- `MEIER` und `JUNG` können die einmal eingegebenen Tabellenzeilen aber nicht ändern oder löschen.

  Auch nicht die Zeilen, die sie selbst eingegeben haben. Die Zeilen werden Bestandteil der Tabelle, die `BRASS` gehört. Wenn man nicht selbst etwas programmiert (z.B. eine Spalte mit `DEFAULT USER` (in Oracle), für die nicht explizit ein Wert angegeben werden kann: s.u.), ist nicht nachzuvollziehen, wer eine bestimmte Zeile eingegeben hat. Manche DBMS haben Erweiterungen für diesen Zweck.
- Man kann später z.B. dem Benutzer `MEIER` noch zusätzlich die fehlenden Rechte geben:

```
GRANT UPDATE, DELETE ON AUFGABEN TO MEIER
```

  Die bereits vorher vergebenen Rechte bleiben dabei bestehen.

GRANT Kommando (4)

Rechte für Tabellen/Sichten in SQL-92:

- `SELECT`: Lesezugriff (Anfragen an die Tabelle).

  In SQL Server kann man `SELECT` für einzelne Spalten vergeben. Das ist nicht im SQL-92 Standard und geht nicht in Oracle und DB2. Aber Sichten haben den gleichen Effekt.

- `INSERT`: Einfügen neuer Zeilen.
- `INSERT`$(A_1, \ldots, A_n)$: Nur für die Spalten A_i können Werte angegeben werden, die übrigen werden mit dem jeweiligen Defaultwert gefüllt ($\rightarrow$ `CREATE TABLE`).

  `INSERT` nur für bestimmte Spalten ist Teil des SQL-92 Standards, aber nur in Oracle möglich (nicht in SQL Server und DB2). Mit Sichten kann man den gleichen Effekt erreichen.

GRANT Kommando (5)

Rechte für Tabellen/Sichten in SQL-92, Forts.:

- `UPDATE`: Änderung von Tabelleneinträgen.
- `UPDATE`$(A_1, \ldots, A_n)$: Nur Daten in den Spalten A_i können geändert werden.
- `DELETE`: Löschung von Tabellenzeilen.
- `REFERENCES`: Anlegen von Fremdschlüsseln, die auf diese Tabelle verweisen.

  Wenn Benutzer A in seiner Tabelle R auf die Tabelle S des Benutzers B verweist, aber B keine `DELETE`-Rechte an R bekommt, kann B referenzierte Zeilen aus seiner eigenen Tabelle S nicht mehr löschen. A kann auch prüfen, ob ein Schlüsselwert in S existiert.

GRANT Kommando (6)

Rechte für Tabellen/Sichten in SQL-92, Forts.:

- `REFERENCES`$(A_1, \ldots, A_n)$: Nur dieser Schlüssel darf in Fremdschlüsseln referenziert werden.

  Das ist nur wichtig, wenn eine Tabelle mehrere Schlüssel hat. Unterstützt in Oracle und DB2 (nicht in SQL Server).

Rechte für Domains/Zeichensätze, etc. in SQL-92:

- `USAGE`: Verwendung z.B. der Domain (benutzerdefinierter Datentyp) in `CREATE TABLE` Anweisungen.

  In keinem der drei DBMS (Oracle, DB2, SQL Server) unterstützt.

GRANT Kommando (7)

Weitere Rechte für Tabellen (nicht im Standard):

- `ALTER`: Recht, die Tabellendefinition zu ändern.

  Unterstützt in Oracle und DB2, nicht in SQL Server.

- `INDEX`: Recht, einen Index für de Tabelle anzulegen.

  Unterstützt in Oracle und DB2, nicht in SQL Server.

Rechte für Prozeduren/Packages (nicht im Standard):

- `EXECUTE`: Recht, die Prozedur auszuführen.

  Unterstützt in allen drei DBMS.

- `BIND`: Neuoptimierung von SQL-Anweisungen.

  Nur in DB2 für "Packages" (SQL-Anweisungen eines Programms).

GRANT Kommando (8)

Rechte für Schema Objekte (nur DB2):

- `ALTERIN`: Recht, jedes Objekt (z.B. Tabelle) des Schemas zu ändern.
- `CREATEIN`: Recht, Objekte im Schema anzulegen.
- `DROPIN`: Recht, Objekte im Schema zu löschen.

Rechte für Directory Objekte (nur Oracle):

- `READ`: Recht, Dateien im Directory zu lesen.

Alle Rechte auf dieser Folie sind nicht im SQL Standard enthalten.

GRANT Kommando (9)

`GRANT ALL PRIVILEGES`:

- Anstatt einzelne Rechte aufzulisten, geht auch:

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

- Dies sind alle Rechte, die der Benutzer, der das `GRANT`-Kommando ausführt, selber für ⟨`Object`⟩ hat.

`TO PUBLIC`:

- Um Rechte an alle Benutzer der Datenbank zu geben (auch an zukünftige Nutzer), schreibt man:

  `GRANT SELECT ON COURSES TO PUBLIC`

GRANT Kommando (10)

`WITH GRANT OPTION`:

- Man kann einem Benutzer ein Recht auch mit der Möglichkeit geben, das Recht weiterzugeben.
- Dazu hängt man die Klausel "`WITH GRANT OPTION`" an das `GRANT`-Kommando an:

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

- Dies erlaubt dem Benutzer `BRASS` auch, die `GRANT`-Option an andere Benutzer weiterzugeben (die es dann selbst weitergeben können, u.s.w.).

GRANT Command (11)

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 (12)

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.

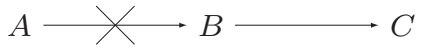
- It is possible to grant "ALL PRIVILEGES" and to revoke them later selectively.

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

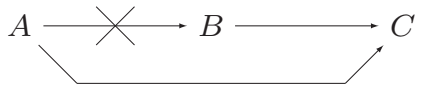
- When a table is deleted, all GRANTs for it are deleted, too. If it is later re-created, only the owner can access it.

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)

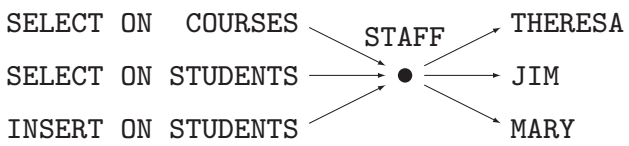
- When a user A is granted a role R, A receives all privileges that were or will be granted to R.

  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.