DML

Tietojen lisäys, muutos ja poisto

Tapahtumien eli transaktioiden hallinta

(kopioi VMware-paketti koneeseesi:
\\ghost\TEMP\TempHuojo\MySQL)

Tietojen lisääminen

- INSERT INTO taulu VALUES (arvot)
- Esim.

```
INSERT INTO prosessorit(cpu, hinta)
VALUES ('AMD', 700)
```

- Sarakkeiden nimiä ei tarvitse luetella, jos kaikille annetaan arvo tai niillä on oletusarvo tai ne sallivat Null-arvon; on kuitenkin suositeltavaa luetella ne!
- Taulusta toiseen:

```
INSERT INTO taulu1 SELECT s1, s2 FROM taulu2
```

- MySQL:ssä voidaan antaa useita lisättäviä rivejä yhdessä käskyssä

Tietojen muuttaminen eli päivittäminen

UPDATE taulu SET arvo = uusiarvo [WHERE ehto]

- Olemassaolevien tietojen päivittäminen

```
UPDATE tuote SET hinta = 120
WHERE tuoteID = 123    -- tuotteen 123 hinnaksi 120
```

Tai esim. 10 %:n hinnan lisäys:

```
UPDATE tuotteet SET hinta = hinta * 1.1
```

- Useaan tauluun perustuva päivitys

```
UPDATE tuote SET varastosaldo = 10
FROM tuote INNER JOIN myyja
ON tuote.myyjaID = myyja.myyjaID
```

Tietojen poistaminen

- Rivien poistaminen taulusta
  - DELETE {FROM} [taulun nimi] WHERE [ehto poistettaville riveille]:

    DELETE FROM tuote WHERE nimi = 'auto';
- Poisto alikyselyn avulla

  DELETE FROM asiakas WHERE asiakastunnus

  NOT IN (SELECT asiakastunnus FROM TILAUS);

Tapahtumien eli transaktioiden hallinta

- Tapahtuma on yleensä useamman käskyn kokonaisuus, joka suoritetaan joko kokonaan tai ei ollenkaan
- Esim. maalitilaston päivitys (joukkueID:t ja maalien määrä välitetään parametreinä):

```
BEGIN TRANSACTION
    INSERT INTO ottelu (joukkue1ID, joukkue2ID, kotimaalit,
          vierasmaalit) VALUES (:j1id, :j2id, :kotimaalit, :vierasmaalit);
    UPDATE joukkue
          SET maalisaldo = maalisaldo + :kotimaalit - :vierasmaalit
          WHERE joukkueID = :j1id;
    UPDATE joukkue
          SET maalisaldo = maalisaldo + :vierasmaalit - :kotimaalit
          WHERE joukkueID = :j2id;
COMMIT;
```

- COMMIT hyväksyy, ROLLBACK peruuttaa tapahtuman

SQL-92 eristyvyystasot

- Eristyvyystaso ilmaisee transaktion eristyvyyttä eli riippumattomuutta muista samanaikaisista transaktioista
- Tavoitteena ehkäistä samanaikaisuuden ongelmia:
  1. Hukatun päivityksen ongelma (lost update): päällekirjoitus
     => estettävä aina!
  2. Keskeneräisen käsittelyn ongelma (dirty read)
     a) luetaan peruutettava
     b) päivitetään peruutettava
  3. Ristiriitaisen tulkinnan ongelma
     a) eritahtinen päivitys (nonrepeatable read)
     b) uusien rivien ongelma (phantom rows)
- Katso esimerkit erillisestä kalvosetistä (J. Rantanen)

  http://data.hamk.fi/materiaalit/tiedonhallinta2/ccr/eristyvyys.pps

Eristyvyystasot

Komennolla

SET TRANSACTION ISOLATION LEVEL

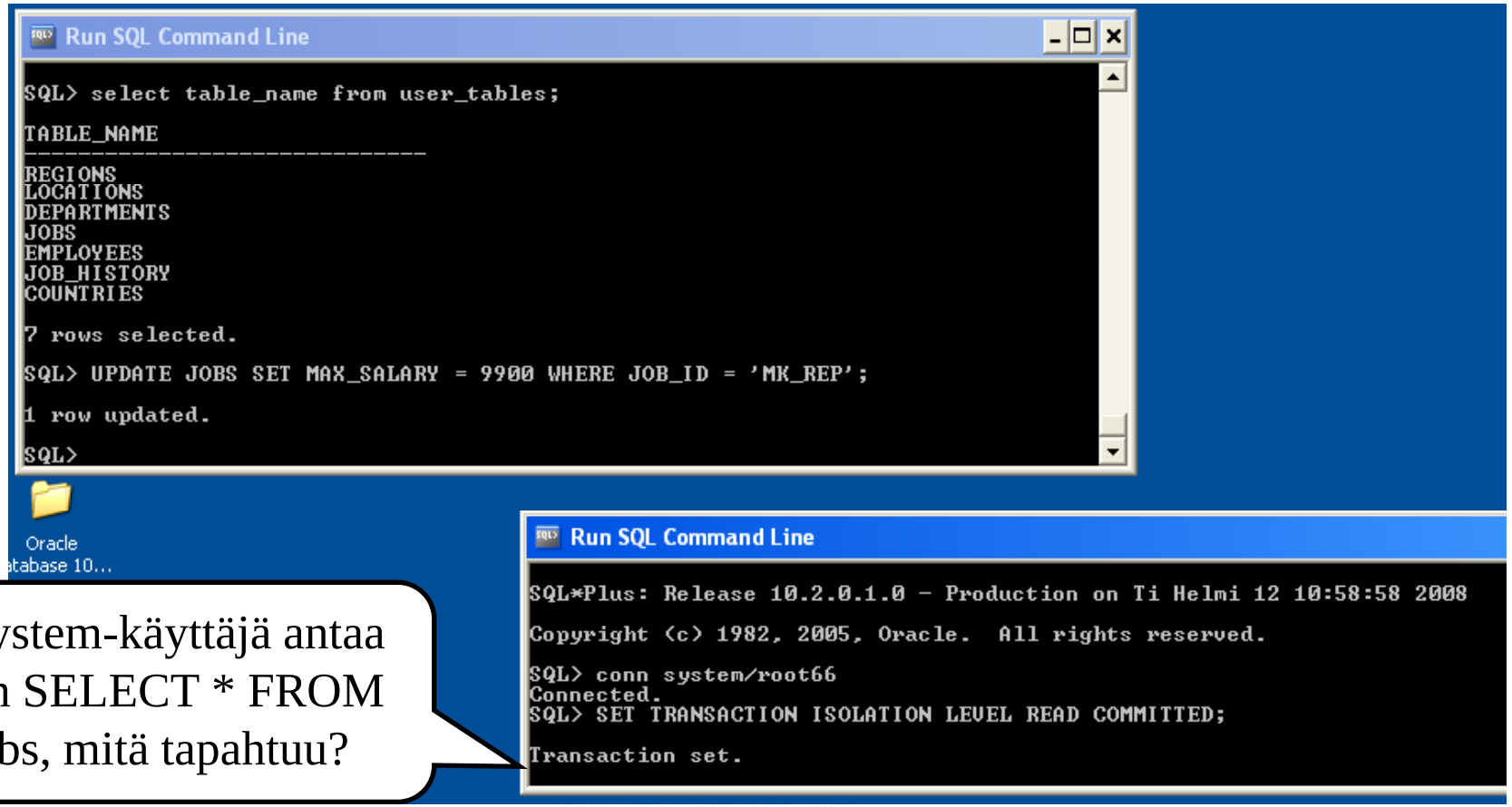
voidaan asettaa neljä eri eristyvyystasoa:

	Dirty Read	Nonrepeatable Read	Phantoms
• **READ UNCOMMITTED**	mahdollinen	mahdollinen	mahdollinen
• **READ COMMITTED**	**Ei** mahdollinen	mahdollinen	mahdollinen
• **REPEATABLE READ**	**Ei** mahdollinen	**Ei** mahdollinen	mahdollinen
• **SERIALIZABLE**	**Ei** mahdollinen	**Ei** mahdollinen	**Ei** mahdollinen

Esim. **SET TRANSACTION ISOLATION LEVEL SERIALIZABLE**

Esimerkki, Oracle

- Kaksi käyttäjää, toinen päivittää taulua, toinen lukee sitä; alkutilanne (ylempi on hr-käyttäjä):



Kun system-käyttäjä antaa käskyn SELECT * FROM hr.jobs, mitä tapahtuu?

Tapahtumien hallintaa muissa tuotteissa

- MySQL: http://www.databasejournal.com/features/mysql/article.php/3393161
- OCELOT: asenna se uudelleen, ohje http://www.ocelot.ca/magazine.htm#CONFIGURE