Deadlock

- Két vagy több folyamat között alakulhat ki
- Ha van A erőforrás és B, és 1es program/szál lelockolja A erőforrást, 2es a B-t, de utána az 1es használná Bt és 2es At, akkor deadlock alakul ki.

Alapértelmezett prioritás

- 5
- min: 1 (MIN_PRIORITY)
- max: 10 (MAX_PRIORITY)

Atomi művelet

- Referencia írása és olvasása
- Primitív változók írása és olvasás (kivéve: long és double)
- értékadás volatile változóknak

Immutable

- Egy objektum immutable, ha annak állapota nem változthatható meg a konstruktor lefutása után.
- Nem tartalmaznak setter metódusokat.
- Minden adattagjuk private és final.
- Leszármazott osztályok nem írhatnak felül metódusokat.
  – Vagy final class deklaráció.
  – Vagy private konstruktor és a példányokat egy factory metódus állítja elő.
- Ha az adattagok között van referencia típus:
  – Az osztály nem tartalmazhat metódust, amely ezt módosítja.
  – A referencia nem osztható meg. A konstruktorban kapott külső referencia nem tárolható, csak a kapott objektum másolata. (defensive copy)
  – Metódusból nem adható vissza az eltárolt referencia, csak a másolata. (defensive copy)

Concurrent api

- ExecutorService execute metódusa Runnable objektumot fogad csak el. (és nincs visszatérési értéke)
- Exceptionök:
  – RejectedExecutionException - if this task cannot be accepted for execution.
  – NullPointerException - if command is null
- A Future objektum fogja tartalmazni a Callable objektumnként definált szál futásának az eredményét.
- A Lock típusú objektumot – hasonlóan a szonkronizációkor használt monitor lock-hoz – egy időben csak egy szál birtokolhatja. DE, néhány lock megengedi hogy egy erőforrást több szál használjon. (Pl. ReadWriteLock read lockja)

SQL

- Adatdefiníciós utasítások (Data Definition Language – DDL), amelyek objektumok létrehozására, módosítására, törlésére valók.
- Adatmanipulációs utasítások (Data Manipulation Language – DML), amelyek rekordok felvitelére, módosítására és törlésére alkalmazhatók.
- Adatkezelő utasítások (Data Query Language – DQL), amelyekkel a letárolt adatokat tudjuk visszakeresni.
- Adatvezérlő utasítások (Data Control Language – DCL), amelyekkel az adatvédelmi és a tranzakció-kezelő műveletek végrehajthatóak.

JDBC

- The DataSource interface, new in the JDBC 2.0 API, provides another way to connect to a data source. The use of a DataSource object is the preferred means of connecting to a data source.
- DriverManager.getConnection(String url[, String user, String password])
- DataSource - poolozott kapcsolat
- Each SQLException provides several kinds of information:
  – a string describing the error. This is used as the Java Exception message, available via the method getMesasge.
  – a “SQLstate” string, which follows either the XOPEN SQLstate conventions or the SQL:2003 conventions. The values of the SQLState string are described in

the appropriate spec. The DatabaseMetaData method getSQLStateType can be used to discover whether the driver returns the XOPEN type or the SQL:2003 type.
– an integer error code that is specific to each vendor. Normally this will be the actual error code returned by the underlying database.
– a chain to a next Exception. This can be used to provide additional error information.
– the causal relationship, if any for this SQLException.

CallableStatement

- tárolt queryket hajt végre
- resultset(ek)et ad vissza
- kiterjeszti: AutoCloseable, PreparedStatement, Statement, Wrapper

Statement

- kiterjeszti: AutoCloseable, Wrapper
- belőle származnak: CallableStatement, PreparedStatement
- The object used for executing a static SQL statement and returning the results it produces.

PreparedStatement

- An object that represents a precompiled SQL statement.
- A SQL statement is precompiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times.

```
PreparedStatement pstmt =
  con.prepareStatement("UPDATE EMPLOYEES SET SALARY = ? WHERE ID = ?");
pstmt.setBigDecimal(1, 153833.00);
```

Tranzakciók

- TRANSACTION_READ_COMMITTED is the default isolation level for

transactions.

- TRANSACTION_READ_UNCOMMITTED - dirty read possible (nem commitolt adatok kiolvasása)
- TRANSACTION_READ_COMMITTED - nonrepeatable read possible (időközben módosult adatok)
- TRANSACTION_REPEATABLE_READ - sorokon még mindig lehetséges phantom read
- TRANSACTION_SERIALIZABLE - mindentől véd

Dirty read

> (Transaction B sees data updated by transaction A. Those updates have not yet been committed.)

Non-Repeatable Reads

> (Transaction B updates rows viewed by transaction A before transaction A commits.) If Transaction A issues the same SELECT statement, the results will be different.

Phantom Reads

> Transaction B inserts a row that would satisfy the query in Transaction A if it were issued again.

Transaction

- JDBC connections start out with auto-commit mode enabled, where each SQL statement is implicitly demarcated with a transaction.
- Users who wish to execute multiple statements per transaction must turn auto-commit off.
- Changing the auto-commit mode triggers a commit of the current transaction (if one is active).
- Connection.setTransactionIsolation() may be invoked anytime if auto-commit is enabled.

- If auto-commit is disabled, Connection.setTransactionIsolation() may only be invoked before or after a transaction. Invoking it in the middle of a transaction leads to undefined behavior.

Mínőségi mutatók

- Karbantarthatóság, megbízhatóság, biztonság, hatéko nyság, használhatóság.

User story

- GIVEN környezet WHEN tevékenység THEN hatás

Követelményelemzés

- megvalósíthatósági elemzés, követelmény feltárás, követelmény specifikáció, követelmény validáció

Deployment

- Ábrázolja a szoftver összetevőinek fizikai (különböző gépeken történő) elhelyezését, a szükséges szoftverkörnyezettel.