Assignment 1 - DB2

Daniel Brand

Andreas Kostecka

1. BEGIN AND COMMIT.

Install your team's DBMS. (At this point locally, on your personal computer. We're working on a server setup.) Answer the following question:

- What are the keywords to manually begin and commit a transaction in your system? Give an example.
  - There is no explicit command for starting a transaction in DB2. The so called unit of work (transaction) begins in DB2 with the first SQL statement and ends with either a COMMIT or ROLLBACK. Between the single statements there can be SAVEPOINTS. A rollback can specify a SAVEPOINT, to which the rollback will occur, or it will roll back to the beginning of the transaction. When a COMMIT is done the entire transaction is committed.

```
INSERT INTO user(Name, Lastname)
values('Daniel', 'Brand');
COMMIT;

INSERT INTO user(Name, Lastname)
values('Daniel', 'Brand');
SAVEPOINT SP1 ON ROLLBACK RETAIN CURSORS;
INSERT INTO user(Name, Lastname)
values('Andres', Kostecka);
ROLLBACK TO SAVEPOINT SP1;
```

  - Depending on the installation of the db2 DBMS Auto-Commit is maybe always on. For executing more statements in the same transaction the Auto-Commit function has to be turned off.
    - Turning Auto-Commit off for a single statement

```
db2 +c "sql-command-to-run"
db2 +c -td "sql-script-to-run.sql"
```

    - Turning Auto-Commit off for all sessions

```
db2set DB2OPTIONS=+c;
```

2. ISOLATION LEVELS.

The ISO standard specifies certain isolation levels which should be implemented in a DBMS. Answer the following questions:

- Are all isolation levels specified in the ISO standard implemented in your DBMS?
  - Yes, DB2 (Express-C) supports all 4 types as described in the ISO standard
    - UR - Uncommitted Read (ISO-Name: Read uncommitted)
    - CS - Cursor Stability (ISO-Name: Read committed)
    - RS - Read Stability (ISO-Name: Repeatable read)
    - RR - Repeatable Read (ISO-Name: Serializable)
- What is the default isolation level in your DBMS?
  - The default isolation level used in DB2 is CS - Cursor Stability (ISO-Name: Read committed)
- How to manually specify a certain isolation level in your DBMS?
  - It is possible to define the isolation level using the WITH { ISOLATION LEVEL } clause in a SQL statement:

```
SELECT ... WITH {UR | CS | RS | RR}
SELECT COUNT(*) FROM tab1 WITH RS
```

  - Another way to change the isolation level for a current session is via the CHANGE command in de DB2 terminal:

```
CHANGE SQLISL TO {UR | CS | RS | RR} or
CHANGE ISOLATION TO {UR | CS | RS | RR}
```

  - Also possible is to set the isolation level variable for the current session via the db2 terminal:

```
SET CURRENT ISOLATION {UR | CS | RS | RR}
```

3. PHENOMENA.

Consider the three undesired phenomena defined by the ISO standard regarding concurrency. For each phenomenon, provide a minimal example that triggers the phenomenon if the isolation level is too low (1 point for each phenomenon).

The phenomena description was cited from the IBM DB2 documentation. It has to be pointed out, that in DB2 the command BEGIN is not explicitly needed, meaning that DB2 does start a transaction with the first SQL statement being issued. Afterwards, everything works as learned in the lecture, so that any example works for DB2 too. For convenience we included BEGIN anyways.

- **dirty read:** This occurs if one transaction can see the results of the actions of another transaction before it commits.

```
T1:
BEGIN
SELECT * FROM table WHERE wherever
T2:
BEGIN
```

```
UPDATE table SET whatever WHERE wherever
/* No commit here */
T1:
SELECT * FROM table WHERE wherever
T2:
ROLLBACK
```

If Transaction 2 reads the value written by Transaction 1, then a DIRTY READ has occurred.

- **Non-Repeatable Read (also called Fuzzy Read):** This occurs if the results of one transaction can be modified or deleted by another transaction before it commits.

```
T1:
BEGIN
SELECT * FROM table WHERE wherever
T2:
BEGIN
UPDATE table SET whatever WHERE wherever
COMMIT
T1:
SELECT * FROM table WHERE wherever
COMMIT
```

If Transaction 1 gets a different result from the each Read, then a NON-REPEATABLE READ has occurred

- **phantom read:** This occurs if the results of a query in one transaction can be changed by another transaction before it commits.

```
T1:
BEGIN
SELECT * FROM table WHERE wherever
T2:
BEGIN
INSERT INTO table(columns) VALUES(values)
COMMIT
T1:
SELECT * FROM table WHERE wherever
COMMIT
```

If Transaction 1 gets a different result from each Select, then a PHANTOM READ has occurred