ORACLE

Locking

Mag. Thomas Griesmayer

concurrency consistency

- Data concurrency, which ensures that users can access data at the same time.
- Data consistency, which ensures that each user sees a consistent view of the data, including visible changes made by the user's own transactions and committed transactions of other users.
- In a single-user database, locks are not necessary because only one user is modifying information. However, when multiple users are accessing and modifying data, the database must provide a way to prevent concurrent modification of the same data.

https://docs.oracle.com/cd/B28359_01/server.111/b28318/consist.htm#CNCPT1313 (29.9.2019)

concurrency consistency

- Locks achieve the following important database requirements:
  - Consistency - the data a session is viewing or changing must not be changed by other sessions until the user is finished.
  - Integrity - the data and structures must reflect all changes made to them in the correct sequence.
- In general, multiuser databases use some form of data locking to solve the problems associated with data concurrency, consistency, and integrity. Locks are mechanisms that prevent destructive interaction between transactions accessing the same resource.

https://docs.oracle.com/cd/B28359_01/server.111/b28318/consist.htm#CNCPT1313 (29.9.2019)

Locking

- Locked objects:
  - Table lock (**TM**)
  - Row lock (**TX**)
- Locked data:
  - User data
  - Datadictonary
- Lock duration:
  - **COMMIT**
  - **ROLLBACK**

Consistent state

```
INSERT INTO CUSTOMER
VALUES (50, 'Fritz', 400);

SELECT *
FROM   CUSTOMER
FOR UPDATE;

UPDATE CUSTOMER
SET BALANCE = 500
WHERE  CUSTOMER_ID = 3;

DELETE FROM CUSTOMER
WHERE  CUSTOMER_ID = 6;

COMMIT;
```

Consistent state

lock duration

- Statements (`UPDATE, DELETE, ...`) generates a lock and are hold for the duration of the transaction.
- Oracle releases all locks by an explicit or implied `COMMIT` or `ROLLBACK` - end of a transaction.
- Lock mode:
  - exclusive lock
  - share lock
- Locks affect the interaction of readers and writers. A reader is a query of a resource, whereas a writer is a statement modifying a resource.

https://docs.oracle.com/cd/B19306_01/server.102/b14220/consist.htm (29.9.2019)

lock behavior

- A row is locked only when modified by a writer - When a statement updates one row, the transaction acquires a lock for this row only.
- Locking level:
  - row
  - block
  - table
- A writer of a row blocks a concurrent writer of the same row - If one transaction is modifying a row, then a row lock prevents a different transaction from modifying the same row simultaneously.
- A reader never blocks a writer - Because a reader of a row does not lock it, a writer can modify this row.
- A writer never blocks a reader - When a row is being changed by a writer, the database uses undo data data to provide readers with a consistent view of the row.

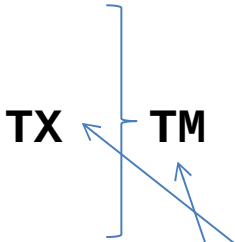
http://docs.oracle.com/cd/E11882_01/server.112/e40540/consist.htm (29.9.2019)

lock behavior

- A row is locked only when modified by a writer - When a statement updates one row, the transaction acquires a lock for this row only.
- Locking level:
  - row
  - block
  - table
- A writer of a row blocks a concurrent writer of the same row - If one transaction is modifying a row, then a row lock prevents a different transaction from modifying the same row simultaneously.
- A reader never blocks a writer - Because a reader of a row does not lock it, a writer can modify this row.
- A writer never blocks a reader - When a row is being changed by a writer, the database uses undo data data to provide readers with a consistent view of the row.

http://docs.oracle.com/cd/E11882_01/server.112/e40540/consist.htm (29.9.2019)

C_ID	NAME	BALANCE
1	Fritz	€ 800
2	Susi	€ 1000
5	Alex	€ 400




```
SELECT *
FROM   CUSTOMER
WHERE  CUSTOMER_ID = 1;


UPDATE CUSTOMER
SET    BALANCE = 1200
WHERE  CUSTOMER_ID=2;


UPDATE CUSTOMER
SET    BALANCE = BALANCE*1.01;


COMMIT;
```

busy wait

C_ID	NAME	BALANCE
1	Fritz	€ 800
2	Susi	€ 1000
5	Alex	€ 400

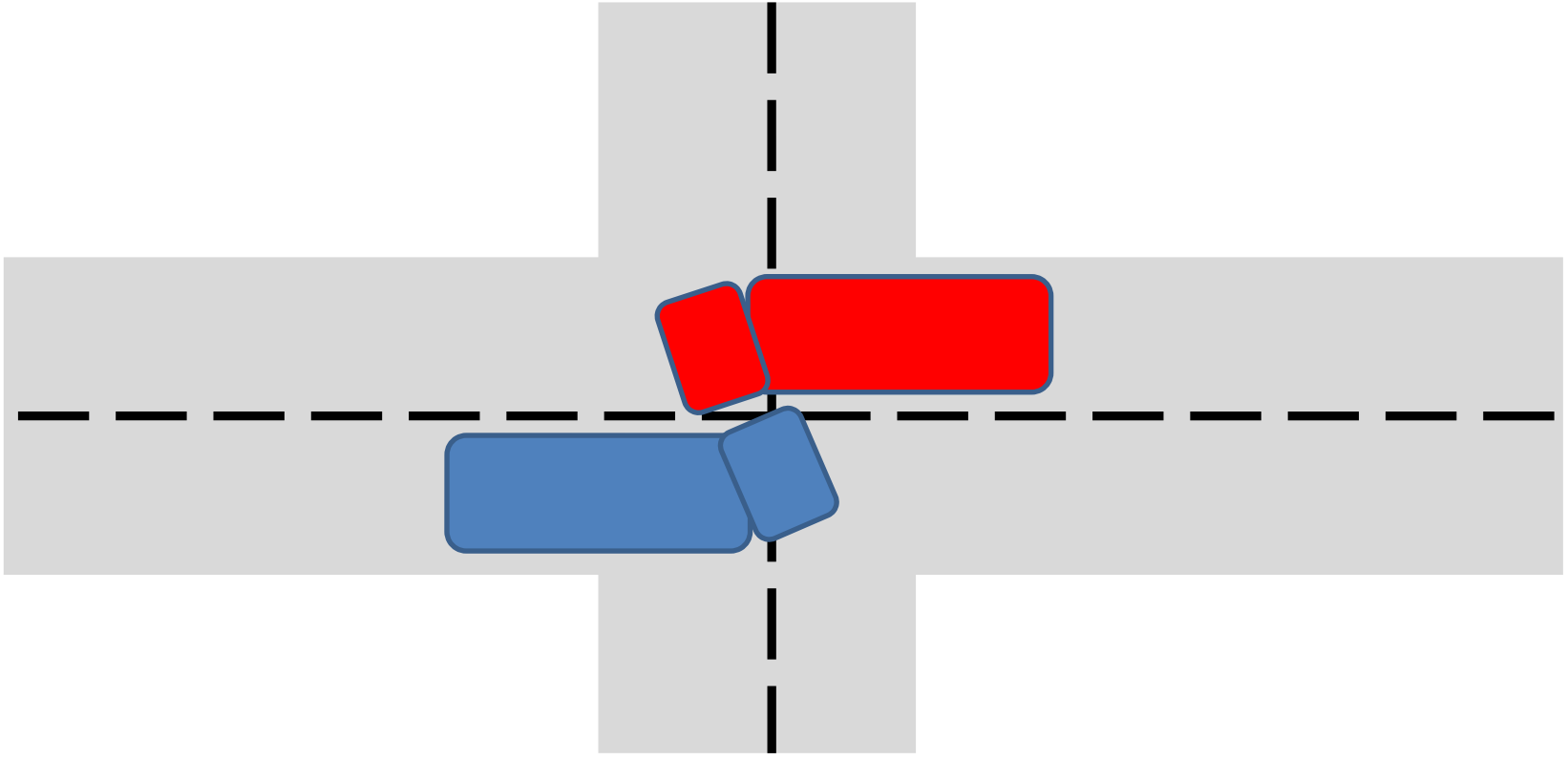
TX

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE+500
WHERE  CUSTOMER_ID=2;

...BUSY WAIT...


1 ROW UPDATED
```

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE+100
WHERE  CUSTOMER_ID=2;


COMMIT;
```

deadlock

- A deadlock occurs when two or more session are waiting for data locked by each other, resulting in all the sessions being blocked.



deadlock

C_ID	NAME	BALANCE
1	Fritz	€ 800
2	Susi	€ 1000
5	Alex	€ 400

TX

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE-500
WHERE  CUSTOMER_ID=1;
```

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE+500
WHERE  CUSTOMER_ID=5;
... BUSY WAIT ...
```

TX

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE+100
WHERE  CUSTOMER_ID=5;
```

```
UPDATE CUSTOMER
SET    BALANCE = BALANCE-100
WHERE  CUSTOMER_ID=1;
ORA-00060: deadlock detected
```