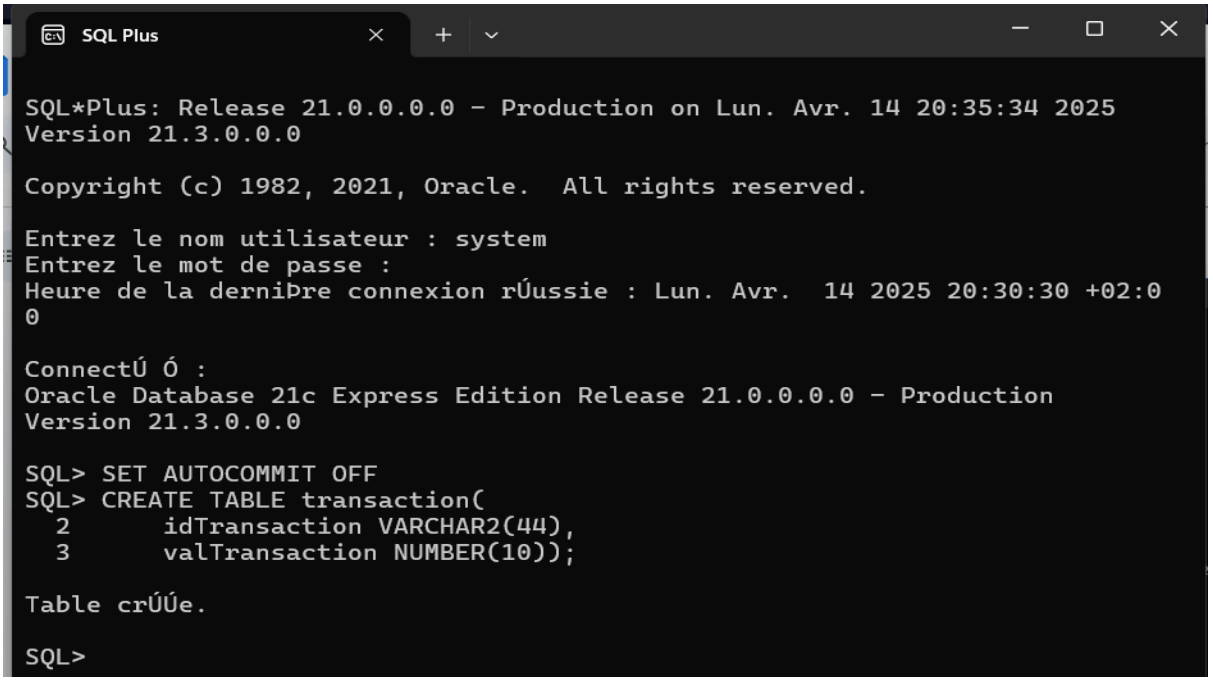
TP n° 5 : Transactions et contrôle de concurrence

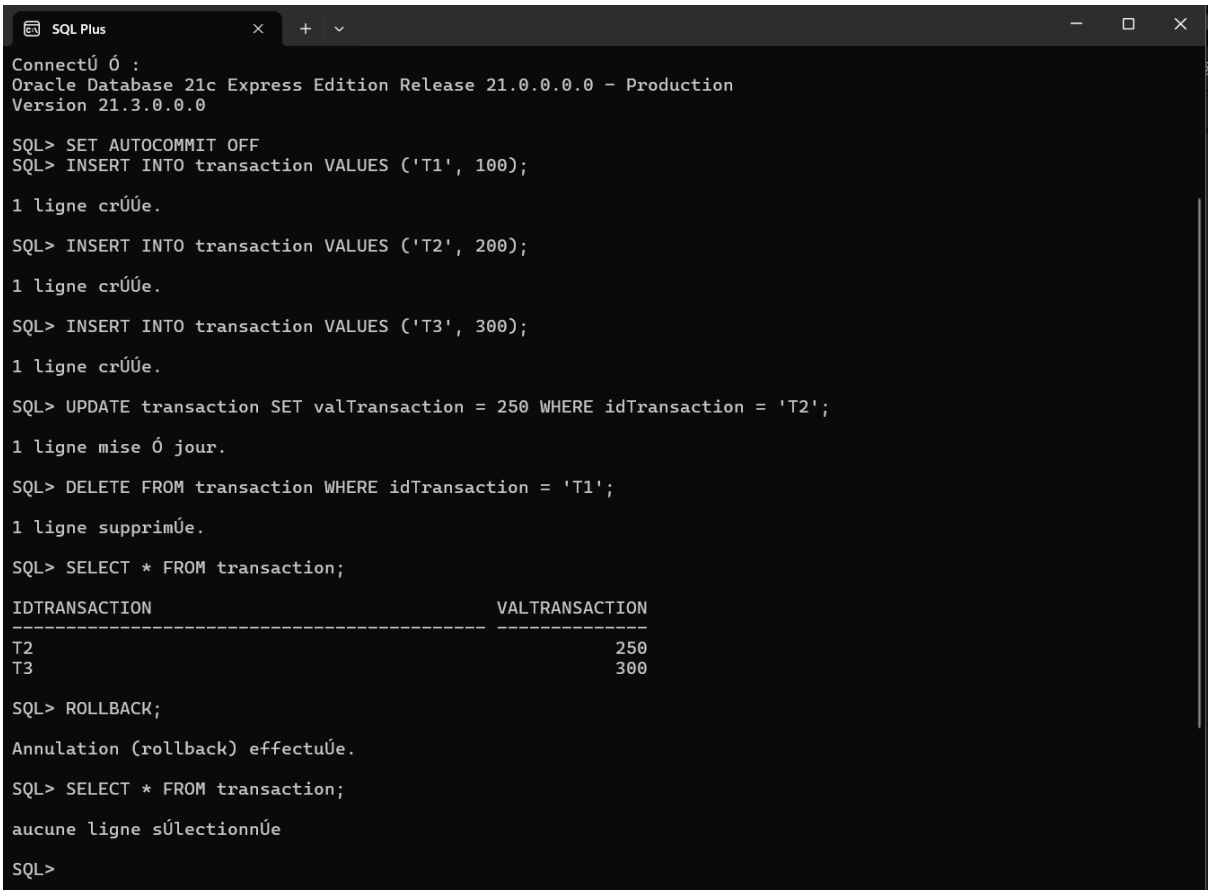
Nombre de pages : 7

Exercice 1 : Atomicité d'une transaction

2) Session S1



Session S2



3)

Session S2

```
SQL> INSERT INTO transaction VALUES ('T4', 400);

1 ligne crÚÚe.

SQL> INSERT INTO transaction VALUES ('T5', 500);

1 ligne crÚÚe.

SQL> quit;
```

Session S1

```
SQL> SELECT * FROM transaction;

aucune ligne sÚlectionnÚe

SQL>
```

⇒ Les données insérées dans S_2 ne sont pas visibles car elles n'ont pas été validées (COMMIT) avant la fermeture de la session.

4)

```
SQL> SELECT * FROM transaction;

aucune ligne sÚlectionnÚe

SQL>
```

⇒Les données ne sont pas préservées car la transaction n'a pas été validée (COMMIT) avant la fermeture brutale.

5)

```
ConnectÚ Ó :
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> INSERT INTO transaction VALUES ('T8', 800);

1 ligne crÚÚe.

SQL> INSERT INTO transaction VALUES ('T9', 900);

1 ligne crÚÚe.

SQL> ALTER TABLE transaction ADD valStransaction NUMBER(10);

Table modifiÚe.

SQL> ROLLBACK;

Annulation (rollback) effectuÚe.
```

```
SQL> SELECT * FROM transaction;

aucune ligne sÚlectionnÚe

SQL>
```

⇒ Les insertions sont annulées (pas de lignes T8/T9)

```
SQL> DESCRIBE transaction;
 Nom                                       NULL ?   Type
 ----------------------------------------- -------- ----------------------------
 IDTRANSACTION                                      VARCHAR2(44)
 VALTRANSACTION                                     NUMBER(10)
 VALSTRANSACTION                                    NUMBER(10)

SQL>
```

⇒ La modification de structure (ajout de colonne) est toujours présente car les DDL (CREATE, ALTER, DROP) provoquent un commit automatique dans Oracle.

6) Conclusion :

- Session : Connexion individuelle à la base de données qui maintient son propre état et contexte.
- Transaction : Suite d'opérations traitée comme une unité indivisible qui doit respecter les propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité).
- Validation :
  - Pour valider une transaction : COMMIT;
  - Pour annuler une transaction : ROLLBACK;
- Atomicité : Une transaction est soit entièrement exécutée, soit entièrement annulée. Si une partie échoue, tout est annulé.

Exercice 2 : Transactions concurrentes

1. Préparation de la base de données

```
ConnectÚ Ó :
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> CREATE TABLE vol(
  2      idVol VARCHAR2(44),
  3      capaciteVol NUMBER(10),
  4      nbrPlacesReservesVol NUMBER(10)
  5  );

Table crÚÚe.

SQL> CREATE TABLE client(
  2      idClient VARCHAR2(44),
  3      prenomClient VARCHAR2(11),
  4      nbrPlacesReservesClient NUMBER(10)
  5  );

Table crÚÚe.

SQL>
```

```
SQL> INSERT INTO vol VALUES ('V1', 100, 0);

1 ligne crÚÚe.

SQL> INSERT INTO client VALUES ('C1', 'Alice', 0);

1 ligne crÚÚe.

SQL> INSERT INTO client VALUES ('C2', 'Bob', 0);

1 ligne crÚÚe.

SQL> COMMIT;

Validation effectuÚe.

SQL>
```

2. Isolation des transactions

- **Dans la session T1 (Transaction 1)**

```
SQL> UPDATE vol SET nbrPlacesReservesVol = nbrPlacesReservesVol + 2 WHERE idVol = 'V1';

1 ligne mise Ó jour.

SQL> UPDATE client SET nbrPlacesReservesClient = nbrPlacesReservesClient + 2 WHERE idClient = 'C1';

1 ligne mise Ó jour.

SQL> SELECT * FROM vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    2

SQL> SELECT * FROM client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             2
C2                                       Bob                               0
```

- **Dans la session T2 (Transaction 2)**

```
ConnectÚ Ó :
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> select * from vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    0

SQL> select * from client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             0
C2                                       Bob                               0

SQL>
```

⇒ **T_1 voit ses propres modifications**
T_2 ne voit pas les modifications non validées de T_1

3. COMMIT et ROLLBACK

- **Dans la session T1 (Transaction 1)**

```
SQL> ROLLBACK;

Annulation (rollback) effectuÚe.

SQL> SELECT * FROM vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    0

SQL> SELECT * FROM client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             0
C2                                       Bob                               0

SQL>
```

- **Dans la session T2 (Transaction 2)**

```
SQL> SELECT * FROM vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    0

SQL> SELECT * FROM client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             0
C2                                       Bob                               0

SQL>
```

⇒ Après ROLLBACK, tout revient à l'état initial
T_2 se comporte comme si T_1 n'avait jamais existé

- **Dans la session T1 (Transaction 1)**

```
SQL> UPDATE vol SET nbrPlacesReservesVol = nbrPlacesReservesVol + 2 WHERE idVol = 'V1';

1 ligne mise Ó jour.

SQL> UPDATE client SET nbrPlacesReservesClient = nbrPlacesReservesClient + 2 WHERE idClient = 'C1';

1 ligne mise Ó jour.

SQL> COMMIT;

Validation effectuÚe.

SQL>
```

- **Dans la session T2 (Transaction 2)**

```
SQL> SELECT * FROM vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    2

SQL> select * from client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             2
C2                                       Bob                               0

SQL>
```

⇒ Après COMMIT, les modifications sont visibles par toutes les sessions.

4. Problème des mises à jour perdues (READ COMMITTED)

Réinitialisation des données :

```
SQL> UPDATE vol SET nbrPlacesReservesVol = 0 WHERE idVol = 'V1';

1 ligne mise Ó jour.

SQL> UPDATE client SET nbrPlacesReservesClient = 0 WHERE idClient IN ('C1', 'C2');

2 lignes mises Ó jour.

SQL> commit;

Validation effectuÚe.

SQL>
```

Exécution concurrente :

- Dans T1 :

```
SQL> SELECT nbrPlacesReservesVol FROM vol WHERE idVol = 'V1';

NBRPLACESRESERVESVOL
--------------------
                   0

SQL> SELECT nbrPlacesReservesClient FROM client WHERE idClient = 'C1';

NBRPLACESRESERVESCLIENT
-----------------------
                      0
```

```
SQL> UPDATE vol SET nbrPlacesReservesVol = nbrPlacesReservesVol + 2 WHERE idVol = 'V1';

1 ligne mise Ó jour.

SQL> UPDATE client SET nbrPlacesReservesClient = nbrPlacesReservesClient + 2 WHERE idClient = 'C1';

1 ligne mise Ó jour.

SQL> COMMIT;

Validation effectuÚe.

SQL>
```

- Dans T2 :

```
SQL> UPDATE vol SET nbrPlacesReservesVol = nbrPlacesReservesVol + 3 WHERE idVol = 'V1';

1 ligne mise Ó jour.

SQL> UPDATE client SET nbrPlacesReservesClient = nbrPlacesReservesClient + 3 WHERE idClient = 'C2';

1 ligne mise Ó jour.

SQL> COMMIT;

Validation effectuÚe.

SQL> SELECT * FROM vol;

IDVOL                                    CAPACITEVOL NBRPLACESRESERVESVOL
---------------------------------------- ----------- --------------------
V1                                               100                    5

SQL> SELECT * FROM client;

IDCLIENT                                 PRENOMCLIEN NBRPLACESRESERVESCLIENT
---------------------------------------- ----------- -----------------------
C1                                       Alice                             2
C2                                       Bob                               3

SQL>
```