1. Created table

```
1   -- Create the employee table
2   CREATE TABLE public.employee (
3       id SERIAL PRIMARY KEY,
4       name VARCHAR(15),
5       status VARCHAR(15)
6   );
```

Data Output | Messages | Notifications

```
CREATE TABLE

Query returned successfully in 60 msec.
```

2. Executing transaction 1:

.

```
9   BEGIN;
10  SELECT txid_current();
11
12  INSERT INTO public.employee (name, s
13  VALUES ('Alice', 'Not fired');
14
15  SELECT *, xmin, xmax
16  FROM public.employee e;
17
18  COMMIT;
19
```

Data Output | Messages | Notifications

```
BEGIN

Query returned successfully in 60 msec.
```

```
10  SELECT txid_current();
11
12  INSERT INTO public.employee (name, status)
13  VALUES ('Alice', 'Not fired');
```

Data Output | Messages | Notifications

	txid_current bigint
1	2185

This is current transaction_id

```
12  INSERT INTO public.employee (name, status)
13  VALUES ('Alice', 'Not fired');
14
15  SELECT *, xmin, xmax
16  FROM public.employee e;
17
18  COMMIT;
19
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 57 msec.
```

```
15  SELECT *, xmin, xmax
16  FROM public.employee e;
17
18  COMMIT;
19
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	1	Alice	Not fired	2184	0

After executing insert and select statements on the 1 option, doing select on the 2 option does not show anything , transaction is not committed.

```
17
18  COMMIT;
19
```

Data Output | Messages | Notifications

```
COMMIT

Query returned successfully in 78 msec.
```

All queries executed as in lecture

```
8   -- First transaction
9   BEGIN;
10
11  SELECT *, xmin, xmax
12  FROM public.employee e;
13
14  COMMIT;
15
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	1	Alice	Not fired	2184	0

Now it is visible as well because we commit our transaction

```
11  SELECT *, xmin, xmax
12  FROM public.employee e;
13
14  COMMIT;
15
```

Data Output | Messages | Notifications

```
COMMIT

Query returned successfully in 63 msec.
```

Transaction 2.

Beginning each transaction, and seeing transaction ids on both transactions

```
21  begin;
22  select *, xmin, xmax
23  from public.employee e;
24  select *, xmin, xmax
25  from public.employee e;
26  commit;
27
28
```

Data Output | Messages | Notifications

SQL

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	1	Alice	Not fired	2184	0

```
29
30  begin;
31  select txid_current();
32  delete from public.employee
33  where id = 1;
34  select *, xmin, xmax
35  from public.employee e;
```

Data Output | Messages | Notifications

SQL

	txid_current bigint
1	2187

```
20
21  begin;
22  select *, xmin, xmax
23  from public.employee e;
24  select *, xmin, xmax
25  from public.employee e;
26  commit;
```

Data Output | Messages | Notifications

SQL

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	1	Alice	Not fired	2190	0

```
32  delete from public.employee
33  where id = 1;
34  select *, xmin, xmax
35  from public.employee e;
36  commit;
37
```

Data Output | Messages | Notifications

```
DELETE 1

Query returned successfully in 58 msec.
```

```
29
30  begin;
31  select txid_current();
32  delete from public.employee
33  where id = 1;
34  select *, xmin, xmax
35  from public.employee e;
36  commit;
37
```

Data Output | Messages | Notifications

SQL

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid

Deleting row from the 2 option and seeing on the 1 option, that xmax(deleting id) is updated to the transaction id from the right side. So in the 2 option, it is still able to see the row, even though it is deleted. After commiting 2 side, the row is deleted

Transaction 3.

In the third transaction, when on the 2 option I updated the row and it showed the updated version, Alice- Fired, but on the 1 option, since it is not committed transaction, i can see that xmax is updated to transaction id from the right side, means that this column is deleted, even its updated, it is deleted and rewrited.

```
40  begin;
41  select *, xmin, xmax
42  from public.employee e;
43  select *, xmin, xmax
44  from public.employee e;
45  commit;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	2	Alice	Not fired	2192	0

```
48  begin;
49  select txid_current();
50  update public.employee
51  set status = 'Fired'
52  where id = 2;
53  select *, xmin, xmax
54  from public.employee e;
55
```

Data Output | Messages | Notifications

```
UPDATE 1

Query returned successfully in 48 msec.
```

After committing, it shows that alice fired updated on the 1 option as well.

```
54  from public.employee e;
55
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid
1	2	Alice	Fired	2193	0

3.Set isolation level to repeatable read

```
56  SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
57  SHOW TRANSACTION ISOLATION LEVEL;
58
```

Data Output | Messages | Notifications

	transaction_isolation text
1	repeatable read

While inserting with Repeatable Read, I can see what's happening with cmin and cmax; they are not changing

```
43  select *, xmin, xmax, cmin, cmax
44  from public.employee e;
45  commit;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid	cmin cid	cmax cid
1	2	Alice	Fired	2193	0	0	0

```
55
56  SET TRANSACTION ISOLATION LEVEL serializable;
57  SHOW TRANSACTION ISOLATION LEVEL;
58
59
60
```

Data Output | Messages | Notifications

	transaction_isolation text
1	serializable

```
43  select *, xmin, xmax, cmin, cmax
44  from public.employee e;
45  commit;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid	cmin cid	cmax cid
1	2	Alice	Fired	2193	0	0	0
2	4	Mark	Not fired	2196	0	0	0
3	5	Maria	Not fired	2197	0	0	0
4	3	Sofia	Fired	2199	0	0	0

Here I tried read committed level and run some updates, nothing change in xmax

```
56  SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
57  SHOW TRANSACTION ISOLATION LEVEL;
58
59
60
```

Data Output | Messages | Notifications

	transaction_isolation text
1	read committed

```
42  from public.employee e;
43  select *, xmin, xmax, cmin, cmax
44  from public.employee e;
45  commit;
```

Data Output | Messages | Notifications

	id [PK] integer	name character varying (15)	status character varying (15)	xmin xid	xmax xid	cmin cid	cmax cid
1	2	Alice	Fired	2193	0	0	0
2	5	Maria	Not fired	2197	0	0	0
3	3	Sofia	Fired	2199	0	0	0
4	4	Mark	Fired	2200	0	0	0

With read committed, it is not prevented to lost update anomalies because it only ensures that each statement sees a consistent snapshot of the database.

concurrent transactions can overwrite each other's changes without awareness. In this case, last one that updated was fired and Mark status was set to Fired.