In Oracle Database, global temporary tables are **permanent** objects whose data are stored on disk and automatically deleted at the end of a session or transaction. In addition, global temporary tables are **visible** to all sessions currently connected to the database.

Oracle 18c introduced private temporary tables whose both table definition and data are temporary and are dropped at the end of a transaction or session. On top of that, Oracle stores private temporary tables in memory and each temporary table is only visible to the session which created it.

X 'Local temporary Table' isn't a thing in Oracle RDBMS

> *전역 임시 테이블 (Global Temporary TABLE)*

[정의 및 특징]

- **SESSION**(또는 **TRANSACTION**) 레벨의 임시 데이터를 저장하는 용도

  (오라클의 실행 계획을 저장하기 위한 Plan 테이블이 전역 임시 테이블의 대표적인 사례)

- **SESSION**(또는 **TRANSACTION**) 레벨의 임시 데이타 저장하기 때문에 RAC 시스템에서 글로벌 동기화 불필요

- DML LOCK 필요없음 (**세션** DATA 간 경합이 발생하지 않기 때문)

- REDO LOG 발생 안함 (임시 DATA 이므로 DML문이라도 redo log 발생 하지 않음)

3. ORACLE VS MS-SQL

- 세션이 종료됨과 동시에 임시 테이블 스키마가 날아가지만 오라클은 테이블 스키마는 남아있다.

- MS-SQL에서 전역 임시테이블(##)은 타 세션 공유 가능

As you may already know, the data in a global temporary table is private to a session. You can only see the data populated in your own session, you cannot see the data inserted by other sessions. So, what happens if you populate the temporary table in your session and then run a parallel query on it? As parallel queries use multiple PX servers and multiple sessions, can PX servers see the data in the temporary table?

```
create global temporary table ttemp
(col1 number) on commit delete rows;
insert into ttemp select rownum from
dual connect by level<=10000;
select /*+ parallel(2) */ count(*) from
ttemp;
COUNT(*)
----------
10000
```

이때. 임시테이블을 생성하는 `WITH` 내부 쿼리에도 병렬처리가 발생할 수 있음!!
실행계획을 보고 대강 문제 발생여부를 확인해야 함.

In this case we had two PX servers scanning the temporary table and the reported count is correct. This indicates individual PX servers were able to see the data populated before by the user session. Parallel queries are different in the sense that parallel sessions working on a temporary table can see the data populated by the QC before. When running a query against the temporary table, the QC is aware of the temporary table and sends the segment information to the PX servers so that they can read the data.