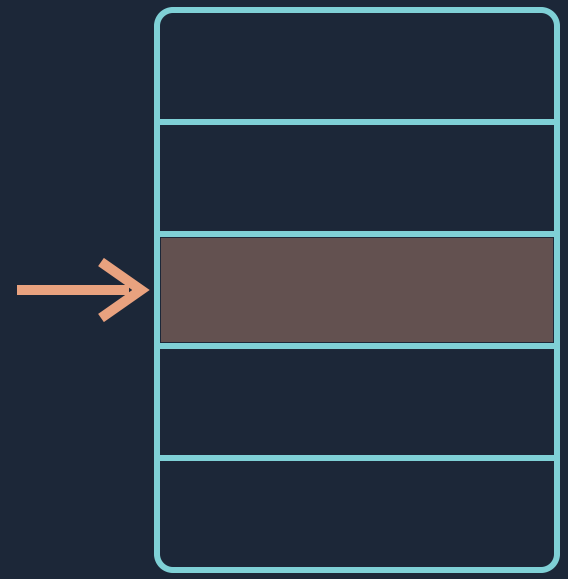
МӘЛІМЕТТЕР ҚОРЫ · ДӘРІС 12

Курсорлармен жұмыс

Курсор түрлері · DECLARE · OPEN · FETCH · CLOSE



Оқытушы: Айдана Карибаева

01

Курсор дегеніміз не?

Курсор — сервер құрған нәтижелер жиыны, онда әрекеттерді **жеке жолдар** үшін орындауға болады.

SQL жиындармен жұмыс істейді; курсор жолдарды **бір-бірлеп** өңдеуге мүмкіндік береді.

Курсор **жадты** алады және баяу — қажет болмаса жиындық операцияларды қолданыңыз.



нәтижелер жиыны

ID	АТЫ	ҚАЛАСЫ
1	Әлі	Алматы
2	Дана	Астана
3	Ерлан	Шымкент
4	Мира	Тараз

02

SQL Server-дегі курсорлар

T-SQL курсорлары

триггерлерде, сақталған процедураларда және сценарийлерде

Сервер (API) курсорлары

серверде орындалады; ADO, OLE DB, **ODBC**, DB-Library арқылы

Клиенттік курсорлар

нәтижелер жиыны клиентте кэштеледі

ODBC (Open Database Connectivity) — SQL CLI стандартын іске асыратын Microsoft API спецификациясы.

03

T-SQL курсорларының түрлері

Статикалық

`STATIC / snapshot`

деректердің **көшірмесі** tempdb-де; басқалардың өзгерістері **көрінбейді**

Динамикалық

`DYNAMIC`

жолға жүгінгенде таңдалады; **барлық** өзгерістер көрінеді; ресурс көп

Тізбекті

`FORWARD_ONLY`

тек **басынан аяғына**; кері қозғалыс жоқ; ең жылдам

Кілттік

`KEYSET`

tempdb-де тек **кілттер** жиыны; мәндердің өзгерісі көрінеді, жаңа жолдар — жоқ

04

Курсормен жұмыс кезеңдері

1 **DECLARE**
жариялау

→

2 **OPEN**
ашу, толтыру

→

3 **FETCH**
жолды алу

→

4 **UPDATE / DELETE**
өзгерту (қажет болса)

→

5 **CLOSE**
жабу

→

6 **DEALLOCATE**
жадты босату

@@FETCH_STATUS = 0 болғанша қайталау

05

Курсорды жариялау

```
DECLARE cursor_name CURSOR
  [ LOCAL | GLOBAL ]
  [ FORWARD_ONLY | SCROLL ]
  [ STATIC | KEYSET | DYNAMIC
    | FAST_FORWARD ]
  [ READ_ONLY | SCROLL_LOCKS
    | OPTIMISTIC ]
  [ TYPE_WARNING ]
FOR select_statement
  [ FOR UPDATE [ OF column_name [,…n] ] ]
```

LOCAL / GLOBAL	көріну аймағы
SCROLL	кез келген бағытта жылжу
READ_ONLY	өзгертуге болмайды
INSENSITIVE	статикалық, өзгерістер көрінбейді
FOR UPDATE	өзгертуге болатын бағандар

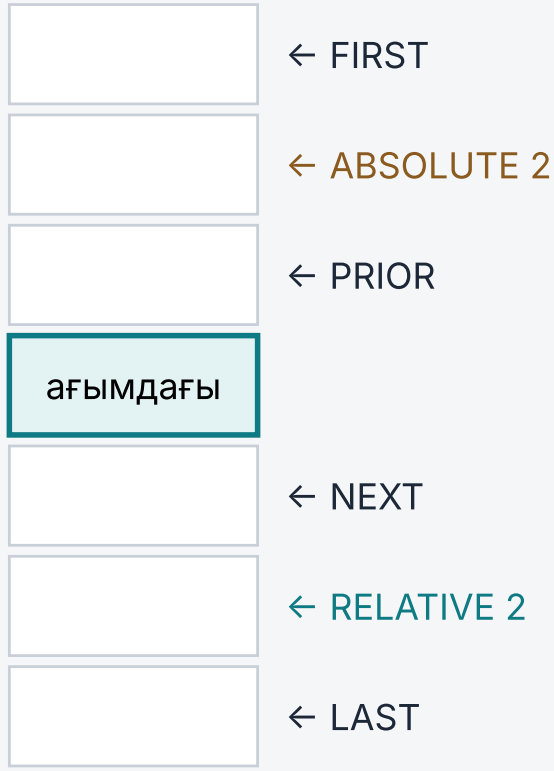
Барлық параметрлер бір-бірімен үйлесе бермейді, мыс. **SCROLL_LOCKS** мен **FAST_FORWARD**.

06 OPEN және FETCH

```
OPEN { [GLOBAL] cursor_name | @cursor_var }
```

Жолдар саны — **@@CURSOR_ROWS** (n — жолдар саны; −n — жүктелуде; 0 — жол жоқ)

```
FETCH [ [ NEXT | PRIOR | FIRST | LAST
  | ABSOLUTE n | RELATIVE n ] FROM ]
  cursor_name
  [ INTO @var1 [, …n] ]
```



INTO — баған мәндері **ретімен** айнымалыларға жазылады; типтері сәйкес болуы керек.

07

Курсор арқылы өзгерту, жабу, жою

```
UPDATE table_name
SET column_name = expression
WHERE CURRENT OF cursor_name;
```

```
DELETE FROM table_name
WHERE CURRENT OF cursor_name;
```

> Курсор тұрған **ағымдағы жол** өзгереді немесе жойылады.

Жабу — қайта ашуға болады

```
CLOSE cursor_name;
```

Жою — жад босатылады

```
DEALLOCATE cursor_name;
```

08 Толық мысал

```
DECLARE @id INT, @name VARCHAR(50);

DECLARE emp_cursor CURSOR FOR
  SELECT employee_id, employee_name
  FROM Employee;

OPEN emp_cursor;
FETCH NEXT FROM emp_cursor INTO @id, @name;

WHILE @@FETCH_STATUS = 0
BEGIN
  PRINT CAST(@id AS VARCHAR) + ' ' + @name;
  FETCH NEXT FROM emp_cursor INTO @id, @name;
END;

CLOSE emp_cursor;
DEALLOCATE emp_cursor;
```

@@FETCH_STATUS	МАҒЫНАСЫ
0	жол сәтті алынды
−1	жолдар бітті / қате
−2	жол жоқ (жойылған)

```
1 Әлі
2 Дана
3 Ерлан
```

09

Мысал: курсор функция ішінде

```
CREATE FUNCTION ТапсырысҚұны (@ТапсырысID int)
RETURNS float AS
BEGIN
  DECLARE @Сома float = 0, @Құны float;
  DECLARE cur CURSOR LOCAL FORWARD_ONLY STATIC FOR
    SELECT Саны * Бағасы
    FROM ТапсырысТауарлары
    WHERE ТапсырысID = @ТапсырысID;
  OPEN cur;
  FETCH NEXT FROM cur INTO @Құны;
  WHILE @@FETCH_STATUS = 0
  BEGIN
    SET @Сома = @Сома + @Құны;
    FETCH NEXT FROM cur INTO @Құны;
  END
  CLOSE cur; DEALLOCATE cur;
  RETURN @Сома;
END
```

ТапсырысТауарлары

ТАПСЫРЫСID	САНЫ	БАҒАСЫ	САНЫ×БАҒАСЫ
7	2	1500	3000
7	1	4000	4000
7	5	200	1000

Сома = 8000

> Бұл есептеу курсорсыз да болады: SELECT SUM(Саны*Бағасы) — әлдеқайда жылдам.

10 Қорытынды

ҚАДАМ	КОМАНДА	МАҒЫНАСЫ
1	`DECLARE c CURSOR FOR SELECT …`	курсорды жариялау
2	`OPEN c`	нәтижелер жиынын толтыру
3	`FETCH NEXT FROM c INTO @v`	келесі жолды алу
4	`WHILE @@FETCH_STATUS = 0`	барлық жолдарды өту
5	`UPDATE … WHERE CURRENT OF c`	ағымдағы жолды өзгерту
6	`CLOSE c / DEALLOCATE c`	жабу және жадты босату