第十三章

select 以外的内容

13.1 insert

13.1.1 直接路径插入

eg: append 和 append_values

```
insert /*+ append */ into kso.bit_emp select * from hr.employees nologging;
insert /*+ append_values */ into dual (dummy) values ('Y');
```

13.1.2 多表插入

语法：

insert all + 多个 into 子句

eg: 基本的插入到单表中的多表插入

```
insert all
into people (person_id, first_name, last_name)
values (person_id, first_name, last_name)
into people (person_id, last_name, parent_id)
values (child1, last_name, person_id)
into people (person_id, last_name, parent_id)
values (child2, last_name, person_id)
into people (person_id, last_name, parent_id)
values (child2, last_name, person_id)
into people (person_id, last_name, parent_id)
values (child4, last_name, person_id)
into people (person_id, last_name, parent_id)
values (child5, last_name, person_id)
into people (person_id, last_name, parent_id)
values (child6, last_name, person_id)
select person_id, first_name, last_name, child1, child2, child3, child4, child5, child6
from denormalized_people;
```

eg: 基本的多表插入

```
insert all
into parents (person_id, first_name, last_name)
values (person_id, first_name, last_name)
into children (first_name, last_name, parent_id)
values (child1, last_name, person_id)
into children (first_name, last_name, parent_id)
values (child2, last_name, person_id)
into children (first_name, last_name, parent_id)
values (child3, last_name, person_id)
into children (first_name, last_name, parent_id)
values (child4, last_name, person_id)
into children (first_name, last_name, parent_id)
values (child5, last_name, person_id)
into children (first_name, last_name, parent_id)
values (child6, last_name, person_id)
select person_id, first_name, last_name, child1, child2, child3, child4, child5, child6
from denormalized_people;
```

13.1.3 条件插入

eg:

```
insert all
when 1=1 then  -- always insert the parent
into people (person_id, first_name, last_name)
values (person_id, first_name, last_name)
```

```
when child1 is not null then  -- only insert non-null children
into people (first_name, last_name, parent_id)
values (child1, last_name, person_id)
when child2 is not null then
into people (first_name, last_name, parent_id)
values  (child2, last_name, person_id)
when child3 is not null then
into people (first_name, last_name, parent_id)
values  (child3, last_name, person_id)
when child4 is not null then
into people (first_name, last_name, parent_id)
values  (child4, last_name, person_id)
when child5 is not null then
into people (first_name, last_name, parent_id)
values  (child5, last_name, person_id)
when child6 is not null then
into people (first_name, last_name, parent_id)
values  (child6, last_name, person_id)
select person_id, first_name, last_name, child1, child2, child3, child4, child5, child6
from denormalized_people;
```

13.1.4 DML 错误日志

1）使用 DBMS_ERRLOG.CREATE_ERROR_LOG 来创建错误日志表。

2）在 INSERT 语句中声明 LOG ERRORS 子句。

eg:

```
execute  dbms_errlog.create_error_log('big_emp', 'big_emp_bad');
desc big_emp
desc big_emp_bad
```

Note: ORA_ERR_TAG$列，允许放入用户自定义的数据，以便进行调试（及ETL过程所处的步骤，或其他性质类似的内容）。

eg: 插入错误日志

```
insert into big_emp (employee_id, first_name, last_name, hire_date, email, department_id)
values (300, 'Bob', 'Loblaw', '01-jan-10', 'bob@yourfavoritelawyer.com', 12345)
log errors into big_emp_bad;
```

eg:

```
insert into big_emp (employee_id, first_name, last_name, hire_date, email, department_id)
values (301, 'Bob', 'Loblaw', '01-jan-10', 'bob@yflawyer.com', 12345)
log errors into big_emp_bad;
```

eg:

```
insert into big_emp (employee_id, first_name, last_name, hire_date, email, department_id)
values (302, 'Bob', 'Loblaw', '01-jan-10', 'bob@yflawyer.com', 12345, 1)
log errors into big_emp_bad;
```

eg:

```
insert into big_emp (employee_id, first_name, last_name, hire_date, email, department_id)
values (303, 'Bob', 'Loblaw', '01-jan-10', 'bob@yflawyer.com', '2A45', 1)
log errors into big_emp_bad;
```

Note: 以上insert语句均有报错，执行失败。失败的记录，不管什么原因导致的，都自动插入到了Errors表中。

eg:

```
select ora_err_mesg$, ora_err_tag$, employee_id from big_emp_bad;
```

eg: 更好的插入错误日志

```
set echo on
create table test_big_insert as select * from dba_objects where 1=2;
desc test_big_insert
alter table test_big_insert modify object_id number(2);
insert into test_big_insert
select * from dba_objects
where object_id is not null;
```

eg: 更好地插入错误日志

```
execute dbms_errlog.create_error_log('test_big_insert', 'tbi_errors');
insert into test_big_insert
select * from dba_objects
where object_id is not null
log errors into tbi_errors
reject limit unlimited;
select count(*) from dba_objects where object_id is not null and length(object_id) < 3;
select count(*) from test_big_insert;
select count(*) from dba_objects
where object_id is not null
and length(object_id)  >2;
select count(*) from tbi_errors;
rollback;
select count(*) from test_big_insert;
select count(*) from tbi_errors;

eg: dbms_errlog.create_error_log 参数
exec dbms_errlog.create_error_log(err_log_table_owner => '&owner',
                                  dml_table_name => '&table_name',
                                  err_log_table_name => '&err_log_table_name',
                                  err_log_table_space => null,
                                  skip_unsupported => true);

13.2 update
eg: update 与 CTAS 之间的性能三角形
set autotrace on
set timing on
update skew2 set col1 = col1*1;
create table skew_temp as
select pk_col, col1*1, col1, col2, col3, col4
from kso.skew2;
set timing off
select count(*) from skew_temp;
@find_sql_stats

eg: update 与 CTAS 性能三角形 —— 10%
select count(*) from skew2 where col1 = 1;
Note: about 10% of the rows col1=1
update skew2 set col1=col1*1 where col1=1;
drop table skew_temp;
create table skew_temp as
select pk_col, case when col1 = 1 then col1 * 1 end col1, col2, col3, col4
from skew2;
alter table skew2 rename to skew_old;
alter table skew_temp rename to skew2;

eg: insert append 而不是大量更新
@recreate_table
@recreate_skew2.sql
set timing on
alter index ks0.sys_c0029558 rename to sys_c0029558_old;
alter index kso.skew2_col1 rename to skew2_col1_old;
alte index kso.skew2_col4 rename to skew2_col4_old;
create table kso.skew2_temp
(pk_dol number,
col1 number,
 col2 varchar2(30),
col3 date,
```

```
col4 varchar2(1)
) segment creation immediate
pctfree 10 pctused 40 initrans 1 maxtrans 255 nocompress logging
storage (initial 1483735040 next 1048576
        minextents 1 maxextents 2147483645
pctincrease 0 freelist 1 freelist groups 1 buffer_pool
default flash_cache default cell_flash_cache default)
tablespace users;
insert /*+ append */ into kso.skew2_temp select /*+ parallel(a,4) 8?
pk_col, col1, case when col1 = 2 then 'ABC' else col2 end, col3, col4
from kso.skew2a;
create index kso.skew2_col1 on kso.skew2_temp(col1)
pctfree 10 initrans 2 maxtrans 255 nologging compute statistics
storage (initial 595591168 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
parallel 8;
create index kso.skew2_col1 on kso.skew2_temp(col4)
pctfree 10 initrans 2 maxtrans 255 compute statistics
storage (initial 65536 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
parallel 8;
create unique index kso.sys_c0029558
on kso.skew2_temp (pk_col)
pctfree 10 initrans 2 maxtrans 255 nologging compute statistics
storage (initial 865075200 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
parallel 8;
alter table 'kso.skew2_temp add primary key (pk_col)
using index pctfree 10 initrans 2 maxtrans 255 nologging
compute statistics
storage (initial 865075200 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users enable;
alter table kso.skew2 rename to skew2_orig;
alter table kso.skew2_temp rename to skew2;

eg: 作为比较的大量更新计时
select my_rows, total_rows, 100*my_rows/total_rows row_percent
from (
select sum(decode(col1,1,1,0)) my_rows, count(*) total_rows
from kso.skew2
);
update /*+ parallel 4 */ kso.skew2 set col2 = 'ABC' where col1 = 2;

13.3 delete
eg: 批量删除
delete from kso.skew2 where col1=1;
@recreate_skew3.sql
set timing on
alter index kso.sys_c002958 rename to sys_c0029558_old;
alter index kso.skew2_col1 rename to skew2_col1_old;
```

```
alter index kso.skew2_col4 rename to skew2_col4_old;
create table kso.skew2_temp
(pk_col number,
col1 number,
col2 varchar2(30),
col3 date,
col4 varchar2(1)
) segment creation immediate
pctfree 10 pctused 40 initrans 1 maxtrans 255 nocompress logging
storage (initial 1483735040 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelist 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users;
insert /*+ append */ into kso.skew2_temp
select /*+ parallel(a,4) */ pk_col, col1, col2,col3, col4
from kso.skew2 where col1 != 1;
create index kso.skew2_col1 on kso.skew2_temp (col1)
pctfree 10 initrans 2 maxtrans 255 nologging compute statistics
storage (initial 595591168 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
parallel 8;
create index kso.skew_col4 on kso.skew2_temp(col4)
pctfree 10 initrans 2 maxtrans 255 compute statistics
storage (initial 65536 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
parallel 8;
create unique index kso.sys_c0029558 on kso.skew2_temp (pk_col)
pctfree 10 initrans 2 maxtrans 255 nologging compute statistics
storage (initial 865075200 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users
praallel 8;
alter table kso.skew2_temp add primary key (pk_col)
using index pctfree 10 initrans 2 maxtrans 255
nologging compute statistics
storage (initial 865075200 next 1048576 minextents 1 maxextents 2147483645
pctincrease 0 freelists 1 freelist groups 1 buffer_pool default
flash_cache default cell_flash_cache default)
tablespace users enable;
```

13.4 merge

提供了经典的 upsert 功能

已存在记录则更新，不存在记录则插入

10g允许删除

13.4.1 语法和用法

语法：

```
merge into table_name
using (subquery) on (subquery.column = table.column)
when matched then update ...
when not matched then insert ...
```

eg: 具有 UPDATE 子句的 MERGE

```
merge into kso.big_emp t
```

```
using (select * from hr.employees) s
on (t.employee_id = s.employee_id)
when matched then update set
-- t.employee_id = s.employee_id,  -- on clause columns not allowed
t.first_name = s.first_name,
t.last_name = s.last_name,
t.email = s.email,
t.phone_number = s.phone_number
t.hire_date = s.hire_date,
t.job_id = s.job_id,
t.salary = s.salary,
t.commission_pct = s.commission_pct,
t.manager_id = s.manager_id,
t.department_id = s.department_id
where (s.salary < 3000)
delete where (s.job_id = 'fired');
eg: 具有 INSERT 子句的 MERGE
merge into big_emp t
using (select * from hr.employees) s
on (t.employee_id = s.employee_id)
when not matched then insert
(t.employee_id,
t.first_name,
t.last_name,
t.email,
t.phone_number,
t.hire_date,
t.job_id,
t.salary,
t.commission_pct,
t.manager_id,
t.department_id)
values
(s.employee_id,
s.first_name,
s.last_name,
s.email,
s.phone_number,
s.hire_date,
s.job_id,
s.salary,
s.commission_pct,
s.manager_id,
s.department_id
) where (s.job_id != 'FIRED');

merge into big_emp_t
using (select * from hr.employees where job_id != 'FIRED') s
on (t.employee_id = s.employee_id)
when not matched then insert
(t.employee_id,
t.first_name,
t.last_name,
t.email,
t.phone_number,
t.hire_date,
t.job_id,
t.salary,
```

```
t.commission_pct,
t.manager_id,
t.department_id)
values
(s.employee_id,
s.first_name,
s.last_name,
s.email,
s.phone_number,
s.hire_date,
s.job_id,
s.salary,
s.commission_pct,
s.manager_id,
s.department_id);
eg: 完整的 MERGE
delete from big_emp where employee_id > 190;
insert into hr.jobs select 'FIRED', 'Fired', 0, 0 from dual;
update hr.employees set job_id = 'FIRED' where employee_id=197;
merge /*+ append */ into kso.big_emp t
using (select * from hr.employees) s
on (t.employee_id = s.employee_id)
when matched then update set
-- t.employee_id = s.employee_id,
t.first_name = s.first_name,
t.last_name = s.last_name,
t.email = s.email,
t.phone_number = s.phone_number
t.hire_date = s.hire_date,
t.job_id = s.job_id,
t.salary = s.salary,
t.commission_pct = s.commission_pct,
t.manager_id = s.manager_id,
t.department_id = s.department_id
where (s.salary < 3000)
delete when (s.job_id = 'FIRED')
when not matched then insert
(t.employee_id,
t.first_name,
t.last_name,
t.email,
t.phone_number,
t.hire_date,
t.job_id,
t.salary,
t.commission_pct,
t.manager_id,
t.department_id)
values
(s.employee_id,
s.first_name,
s.last_name,
s.email,
s.phone_number,
s.hire_date,
s.job_id,
s.salary,
s.commission_pct,
```

```
s.manager_id,
s.department_id
) where (s.job_id != 'FIRED');
```

13.4.2 性能比较

eg: insert, merge, CTAS 性能比较

```
@compare_insert_merge_ctas.sql
@flush_pool
alter system flush shared_pool;
select name, vlaue from v$mystat s, v$statname n
where n.statistic# = s.statistic# and name = 'physical writes direct';
create /* compare_insert_merge_ctas.sql */ table skew3
as select * from skew;
select name, value from v$mystat s, v$statname n
where n.statistic# = s.statistic# and name = 'physical writes direct';
truncate table skew3 drop storage;
insert /*+ append */ /* compare_insert_merge_ctas.sql */
into skew3 select * from skew;
select name, value from v$mystat s, v$statname n
where n.statistic# = s.statistic# and name = 'physical writes direct';
truncate table skew3 drop storage;
merge /*+ append */ /* compare_insert_merge_ctas.sql */
into skew3 t
using (select * from skew) s
on (t.pk_col = s.pk_col)
when not matched then insert
(t.pk_col, t.col1, t.col2, t.col3, t.col4)
values (s.pk_col, s.col1, s.col2, s.col3, s.col4);
select name, value from v$mystat s, v$statname n
where n.statistic# = s.statistic# and name = 'physical writes direct';
@fss2
```

13.5 小结

Note: 目前已经开发出在进行更新和删除是使用的直接路径插入。