1 Создание

CREATE TABLE FULL\_JOBS

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NN NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NN NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NN NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NN NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NN NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVAL CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PK PRIMARY KEY (EMPLOYEE\_ID, START\_DATE));

Insert into full\_jobs(EMPLOYEE\_ID, START\_DATE, END\_DATE, JOB\_ID, JOB\_TITLE, MIN\_SALARY, MAX\_SALARY, DEPARTMENT\_ID)

Select jh.employee\_id, jh.start\_date, jh.end\_date, jh.job\_id, j.job\_title, j.min\_salary, j.max\_salary, jh.department\_id from job\_history jh join jobs j on jh.job\_id = j.job\_id;

Select \* from full\_jobs;

Create Cluster jobsjob\_history1 (EMPLOYEE\_ID NUMBER(6,0), START\_DATE DATE) size 512;

Create index idx\_jobsjob\_history1 on cluster jobsjob\_history1;

CREATE TABLE full\_jobs1

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NN1 NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NN1 NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NN1 NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NN1 NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NN1 NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVAL1 CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PK1 PRIMARY KEY (EMPLOYEE\_ID, START\_DATE)) cluster jobsjob\_history1(EMPLOYEE\_ID, START\_DATE);

begin

for xx in (select \*from FULL\_JOBS)

loop

insert into full\_jobs1 values (xx.employee\_id, xx.start\_date, xx.end\_date, xx.job\_id, xx.job\_title, xx.min\_salary, xx.max\_salary, xx.department\_id);

end loop;

end;

Create Cluster jobsjob\_history2 (EMPLOYEE\_ID NUMBER(6,0), START\_DATE DATE) size 1024;

Create index idx\_jobsjob\_history2 on cluster jobsjob\_history2;

CREATE TABLE full\_jobs2

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NN2 NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NN2 NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NN2 NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NN2 NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NN2 NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVAL2 CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PK2 PRIMARY KEY (EMPLOYEE\_ID, START\_DATE)) cluster jobsjob\_history2(EMPLOYEE\_ID, START\_DATE);

begin

for xx in (select \*from FULL\_JOBS)

loop

insert into full\_jobs2 values (xx.employee\_id, xx.start\_date, xx.end\_date, xx.job\_id, xx.job\_title, xx.min\_salary, xx.max\_salary, xx.department\_id);

end loop;

end;

Create Cluster jobsjob\_history3 (EMPLOYEE\_ID NUMBER(6,0), START\_DATE DATE) size 2048;

Create index idx\_jobsjob\_history3 on cluster jobsjob\_history3;

CREATE TABLE full\_jobs3

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NN3 NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NN3 NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NN3 NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NN3 NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NN3 NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVAL3 CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PK3 PRIMARY KEY (EMPLOYEE\_ID, START\_DATE)) cluster jobsjob\_history3(EMPLOYEE\_ID, START\_DATE);

begin

for xx in (select \*from FULL\_JOBS)

loop

insert into full\_jobs3 values (xx.employee\_id, xx.start\_date, xx.end\_date, xx.job\_id, xx.job\_title, xx.min\_salary, xx.max\_salary, xx.department\_id);

end loop;

end;

Create Cluster jobsjob\_history4 (EMPLOYEE\_ID NUMBER(6,0), START\_DATE DATE) size 4096;

Create index idx\_jobsjob\_history4 on cluster jobsjob\_history4;

CREATE TABLE full\_jobs4

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NN4 NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NN4 NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NN4 NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NN4 NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NN4 NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVAL4 CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PK4 PRIMARY KEY (EMPLOYEE\_ID, START\_DATE)) cluster jobsjob\_history4(EMPLOYEE\_ID, START\_DATE);

begin

for xx in (select \*from FULL\_JOBS)

loop

insert into full\_jobs4 values (xx.employee\_id, xx.start\_date, xx.end\_date, xx.job\_id, xx.job\_title, xx.min\_salary, xx.max\_salary, xx.department\_id);

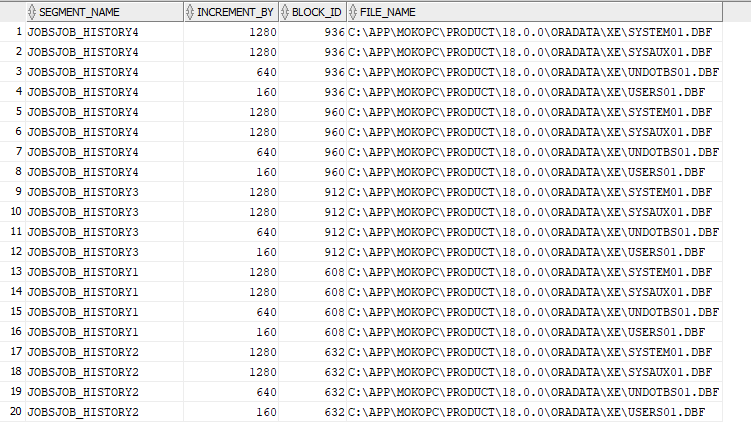
end loop;

end;

Анализ

select ext.segment\_name, df.increment\_by, ext.block\_id, DF.FILE\_NAME

from dba\_extents ext, dba\_data\_files df where SEGMENT\_NAME like 'JOBSJOB\_HISTORY%';



2. Создание

Create Cluster jobsjob\_history\_hash (EMPLOYEE\_ID NUMBER(6,0), START\_DATE DATE) size 2048 HASHKEYS 50;

CREATE TABLE full\_jobsHash

( EMPLOYEE\_ID NUMBER(6,0) CONSTRAINT FULLHIST\_EMPLOYEE\_NNHash NOT NULL ENABLE,

START\_DATE DATE CONSTRAINT FULLHIST\_START\_DATE\_NNHash NOT NULL ENABLE,

END\_DATE DATE CONSTRAINT FULLHIST\_END\_DATE\_NNHash NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT FULLHIST\_JOB\_NNHash NOT NULL ENABLE,

JOB\_TITLE VARCHAR2(35 BYTE) CONSTRAINT FULLOB\_TITLE\_NNHash NOT NULL ENABLE,

MIN\_SALARY NUMBER(6,0),

MAX\_SALARY NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

CONSTRAINT FULLHIST\_DATE\_INTERVALHash CHECK (end\_date > start\_date) ENABLE,

CONSTRAINT FULLHIST\_EMP\_ID\_ST\_DATE\_PKHash PRIMARY KEY (EMPLOYEE\_ID, START\_DATE)) cluster jobsjob\_history\_hash(EMPLOYEE\_ID, START\_DATE);

begin

for xx in (select \*from FULL\_JOBS)

loop

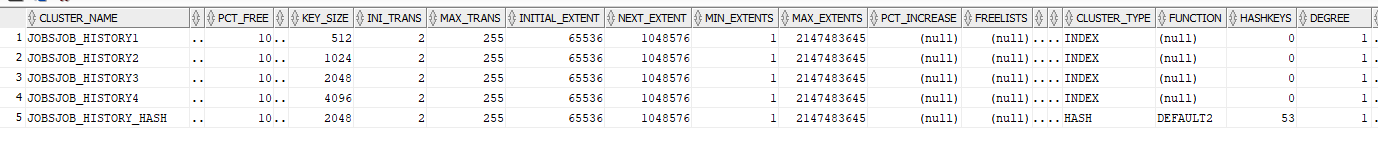
insert into full\_jobsHash values (xx.employee\_id, xx.start\_date, xx.end\_date, xx.job\_id, xx.job\_title, xx.min\_salary, xx.max\_salary, xx.department\_id);

end loop;

end;

Анализ

select \* from USER\_CLUSTERS



3.

CREATE OR REPLACE TYPE nest AS TABLE OF VARCHAR2(30);

CREATE TABLE EMPLOYEES\_COPY

( "EMPLOYEE\_ID" NUMBER(6,0),

"FIRST\_NAME" VARCHAR2(20 BYTE),

"LAST\_NAME" VARCHAR2(25 BYTE) CONSTRAINT "EMP\_LAST\_NAME\_NN\_COPY" NOT NULL ENABLE,

"EMAIL" VARCHAR2(25 BYTE) CONSTRAINT "EMP\_EMAIL\_NN\_COPY" NOT NULL ENABLE,

"PHONE\_NUMBER" VARCHAR2(20 BYTE),

"HIRE\_DATE" DATE CONSTRAINT "EMP\_HIRE\_DATE\_NN\_COPY" NOT NULL ENABLE,

"JOB\_ID" VARCHAR2(10 BYTE) CONSTRAINT "EMP\_JOB\_NN\_COPY" NOT NULL ENABLE,

"SALARY" NUMBER(8,2),

"COMMISSION\_PCT" NUMBER(2,2),

"MANAGER\_ID" NUMBER(6,0),

"DEPARTMENT\_ID" NUMBER(4,0)

CONSTRAINT "EMP\_SALARY\_MIN\_COPY" CHECK (salary > 0) ENABLE,

CONSTRAINT "EMP\_EMP\_ID\_PK\_COPY" PRIMARY KEY ("EMPLOYEE\_ID"),

CONSTRAINT "EMP\_EMAIL\_UK\_COPY" UNIQUE ("EMAIL"),

CONSTRAINT "EMP\_DEPT\_FK\_COPY" FOREIGN KEY ("DEPARTMENT\_ID")

REFERENCES "A"."DEPARTMENTS" ("DEPARTMENT\_ID") ENABLE,

CONSTRAINT "EMP\_JOB\_FK\_COPY" FOREIGN KEY ("JOB\_ID")

REFERENCES "A"."JOBS" ("JOB\_ID") ENABLE,

CONSTRAINT "EMP\_MANAGER\_FK\_COPY" FOREIGN KEY ("MANAGER\_ID")

REFERENCES "A"."EMPLOYEES" ("EMPLOYEE\_ID") ENABLE

);

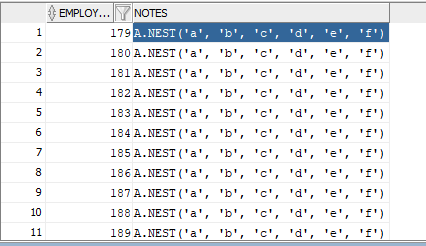
ALTER TABLE employees\_copy ADD notes nest NESTED TABLE notes STORE AS nest\_tab;

Insert into employees\_copy(EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID, NOTES)

Select EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, EMAIL, PHONE\_NUMBER, HIRE\_DATE, JOB\_ID, SALARY, COMMISSION\_PCT, MANAGER\_ID, DEPARTMENT\_ID, nest('a', 'b', 'c', 'd', 'e', 'f') from employees;

Select employee\_id, notes from employees\_copy;

Select employee\_id, /\*+ NESTED\_TABLE\_GET\_REFS +\*/ notes from employees\_copy;



4.

CREATE OR REPLACE TYPE nest AS TABLE OF VARCHAR2(30);

CREATE OR REPLACE TYPE job\_type AS OBJECT (

EMPLOYEE\_ID NUMBER(6,0),

START\_DATE DATE,

END\_DATE DATE,

JOB\_ID VARCHAR2(10 BYTE),

DEPARTMENT\_ID NUMBER(4,0)

);

CREATE TYPE all\_jobs IS TABLE OF job\_type;

CREATE TABLE EMPLOYEES\_OBJ

( EMPLOYEE\_ID NUMBER(6,0),

FIRST\_NAME VARCHAR2(20 BYTE),

LAST\_NAME VARCHAR2(25 BYTE) CONSTRAINT EMP\_LAST\_NAME\_NN\_OBJ NOT NULL ENABLE,

EMAIL VARCHAR2(25 BYTE) CONSTRAINT EMP\_EMAIL\_NN\_OBJ NOT NULL ENABLE,

PHONE\_NUMBER VARCHAR2(20 BYTE),

HIRE\_DATE DATE CONSTRAINT EMP\_HIRE\_DATE\_NN\_OBJ NOT NULL ENABLE,

JOB\_ID VARCHAR2(10 BYTE) CONSTRAINT EMP\_JOB\_NN\_OBJ NOT NULL ENABLE,

SALARY NUMBER(8,2),

COMMISSION\_PCT NUMBER(2,2),

MANAGER\_ID NUMBER(6,0),

DEPARTMENT\_ID NUMBER(4,0),

previousJobs all\_jobs,

CONSTRAINT EMP\_SALARY\_MIN\_OBJ CHECK (salary > 0) ENABLE,

CONSTRAINT EMP\_EMP\_ID\_PK\_OBJ PRIMARY KEY (EMPLOYEE\_ID),

CONSTRAINT EMP\_EMAIL\_UK\_OBJ UNIQUE (EMAIL),

CONSTRAINT EMP\_DEPT\_FK\_OBJ FOREIGN KEY (DEPARTMENT\_ID)

REFERENCES A.DEPARTMENTS (DEPARTMENT\_ID) ENABLE,

CONSTRAINT EMP\_JOB\_FK\_OBJ FOREIGN KEY (JOB\_ID)

REFERENCES A.JOBS (JOB\_ID) ENABLE,

CONSTRAINT EMP\_MANAGER\_FK\_OBJ FOREIGN KEY (MANAGER\_ID)

REFERENCES A.EMPLOYEES (EMPLOYEE\_ID) ENABLE

) NESTED TABLE previousJobs STORE AS every\_job;

declare

curHist all\_jobs;

begin

for xx in (select \*from employees) loop

Select job\_type(EMPLOYEE\_ID, START\_DATE, END\_DATE, JOB\_ID, DEPARTMENT\_ID) bulk COLLECT INTO curHist from JOB\_HISTORY where employee\_id = xx.EMPLOYEE\_ID;

Insert into employees\_obj values (xx.EMPLOYEE\_ID, xx.FIRST\_NAME, xx.LAST\_NAME, xx.EMAIL, xx.PHONE\_NUMBER, xx.HIRE\_DATE, xx.JOB\_ID, xx.SALARY, xx.COMMISSION\_PCT, xx.MANAGER\_ID, xx.DEPARTMENT\_ID, curHist);

end loop;

end;

Select \* from employees\_obj;

