TOPICS COVERED

- DDL
- DML
- WHERE clause(complete)
- CONDITIONS
- RETRIEVAL
- FUNCTIONS
- ALIASES
- FETCH

DDL

- CREATE TABLE, VIEW and INDEX
- ALTER TABLE
- DROP TABLE,VIEW AND INDEX
- RENAME
- TRUNCATE

General syntax for creating a table

```
Create `table employee
      (empno char(4) not null primary key,
      Empname varchar(10) not null,
      Deptno char(4),
      Salary decimal(7,2) not null,
      Foreign key (deptno) references dept On delete
  CASCADE/SETNULL/RESTRICT)
 In dbmaple.tsmaple;
```

Before specifying a FOREIGN KEY in employee table, DEPT table with deptno as primary key must exist with records.

DDL

Create Unique Index

Create Unique Index inddept on dept(deptno)

Table will use that primary key as an search key while retrieval, so there should be an index for easy and fast searching.

DDL

CREATE TABLE USING LIKE

- A table can also be created with guidelines from a already existing table.
- Primary and Foreign key definitions cannot be inherited.
- The records of the table cannot be inherited.
- Only the table structure(attributes and their Data types and length) will be nherited.
- **Syntax:**

CREATE TABLE table name LIKE [Existing table] name.

DDL

ALTER Statements.

ALTER TABLE allows:

- Addition of new columns to a table.
- Addition and removal of Primary and foreign key specification.

ALTER TABLE does not allow :

- Change of data type
- Change column length
- Change NULL attribute
- Rearrange columns
- Remove a column

DDL

Adding New Column.

Syntax for adding a column to an existing table:

```
ALTER TABLE table name
    ADD column name data type;

ALTER TABLE EMPLOYEE
    ADD  mobile_no char(10) ;
```

DDL

Altering Primary Key for a table

Adding a Primary Key specification for an existing attribute:

Alter Table employee Add primary key (emp_no);

Removing a Primary Key specification from a table:

Alter Table employee Drop primary key.

DDL

Altering Foreign Key for a table

```
Alter table tablename
        Add foreign key <column name>
        References <primary key table>
        On delete <delete rule>

Alter table employee
        Add foreign key dept_no
        References department
        On delete restrict

Alter table employee
        Drop foreign key
```

DDL

DROP TABLE

- Existing table can be deleted
- Specified database entry is removed from system catalog.
- All indexes and views defined for table are also dropped
- The object cannot be referred any longer.

Syntax:

DROP TABLE table name; DROP INDEX index name

DROP TABLE Employee;   DROP INDEX empidx

Scenario 1

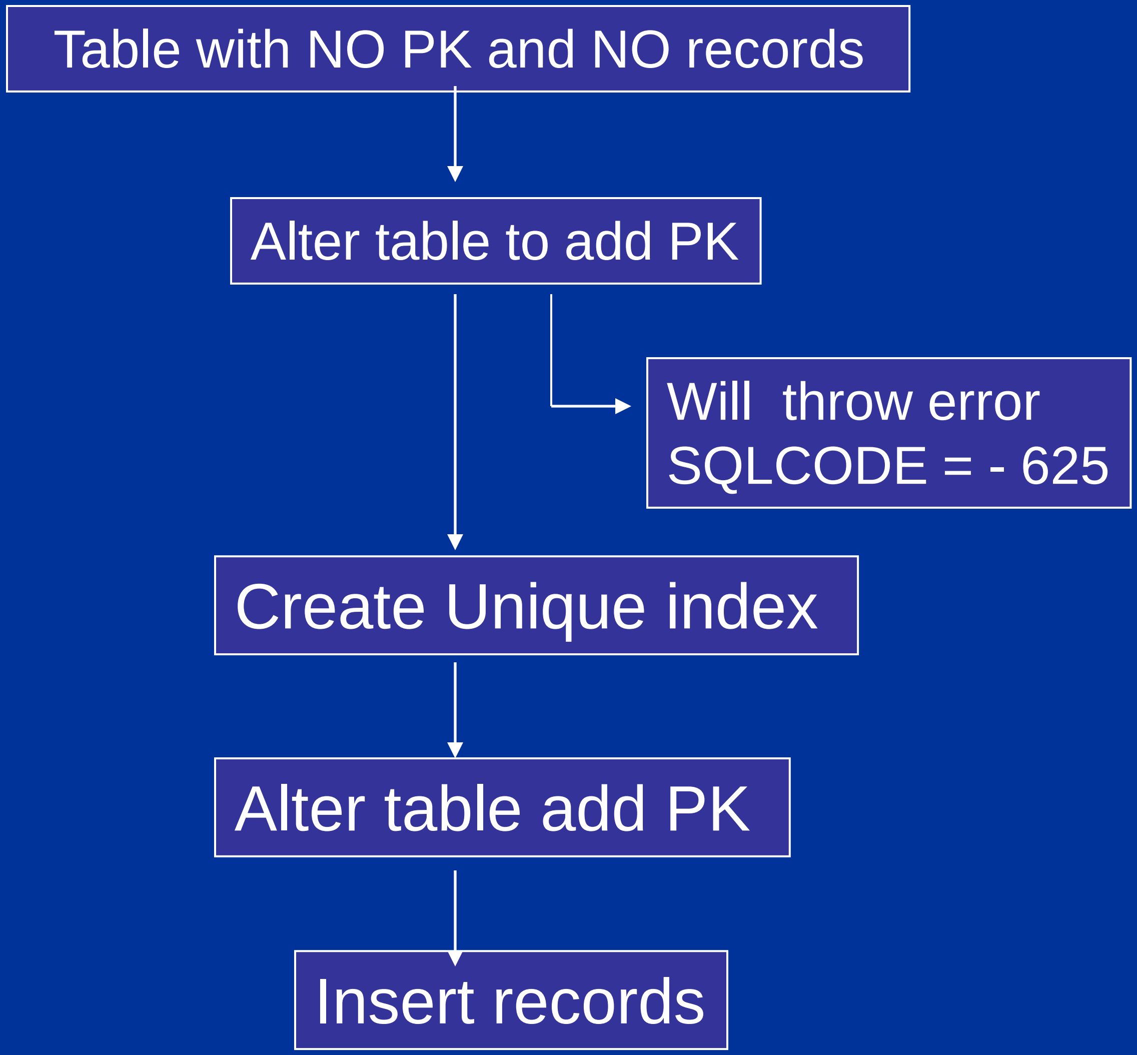


Table with NO PK and NO records
Alter table to add PK
Will throw error SQLCODE = - 625
Create Unique index
Alter table add PK
Insert records

Scenario 2

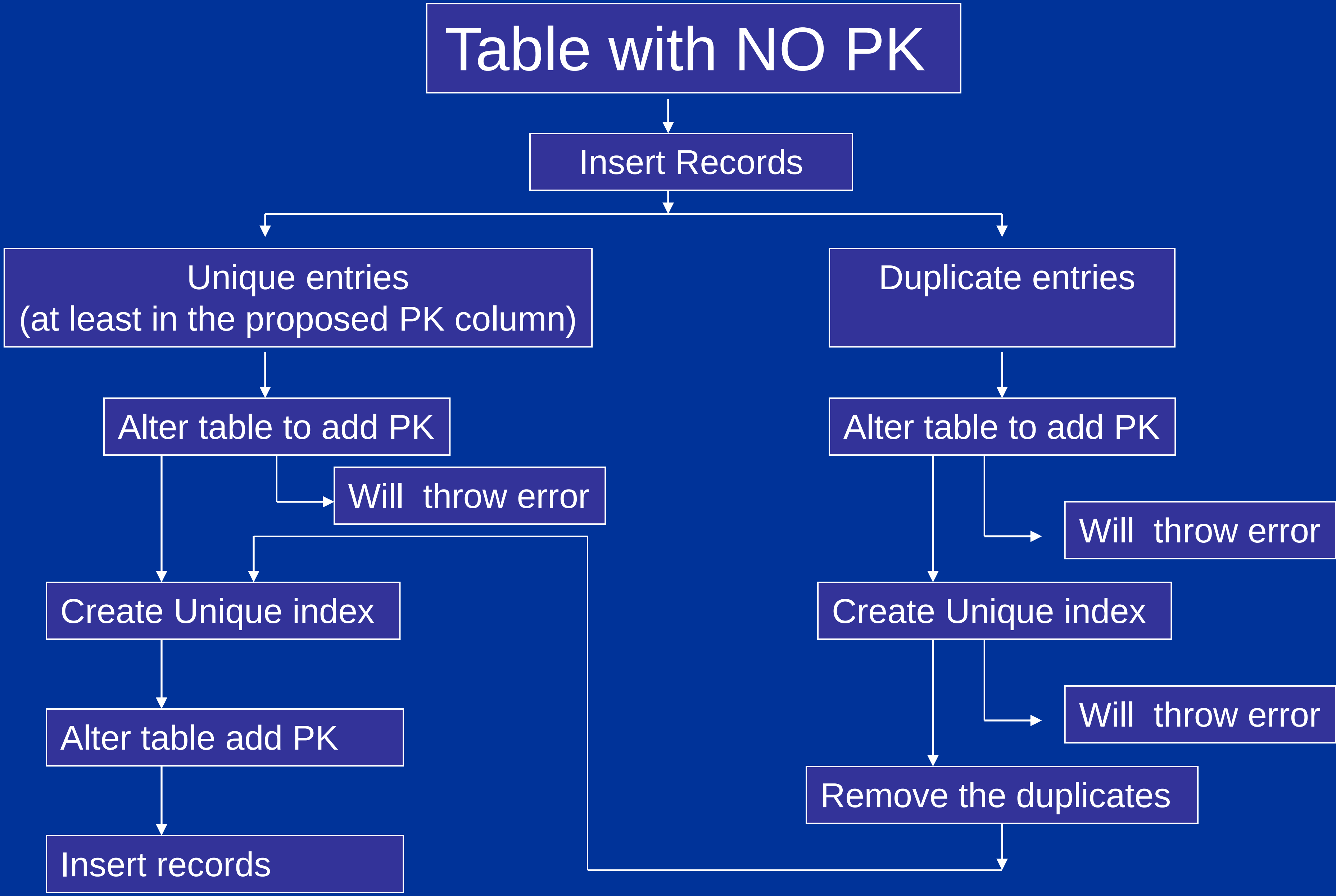


Table with NO PK
Insert Records
Unique entries
(at least in the proposed PK column)
Duplicate entries
Alter table to add PK
Will throw error
Alter table to add PK
Will throw error
Create Unique index
Create Unique index
Will throw error
Alter table add PK
Remove the duplicates
Insert records

Scenario 3

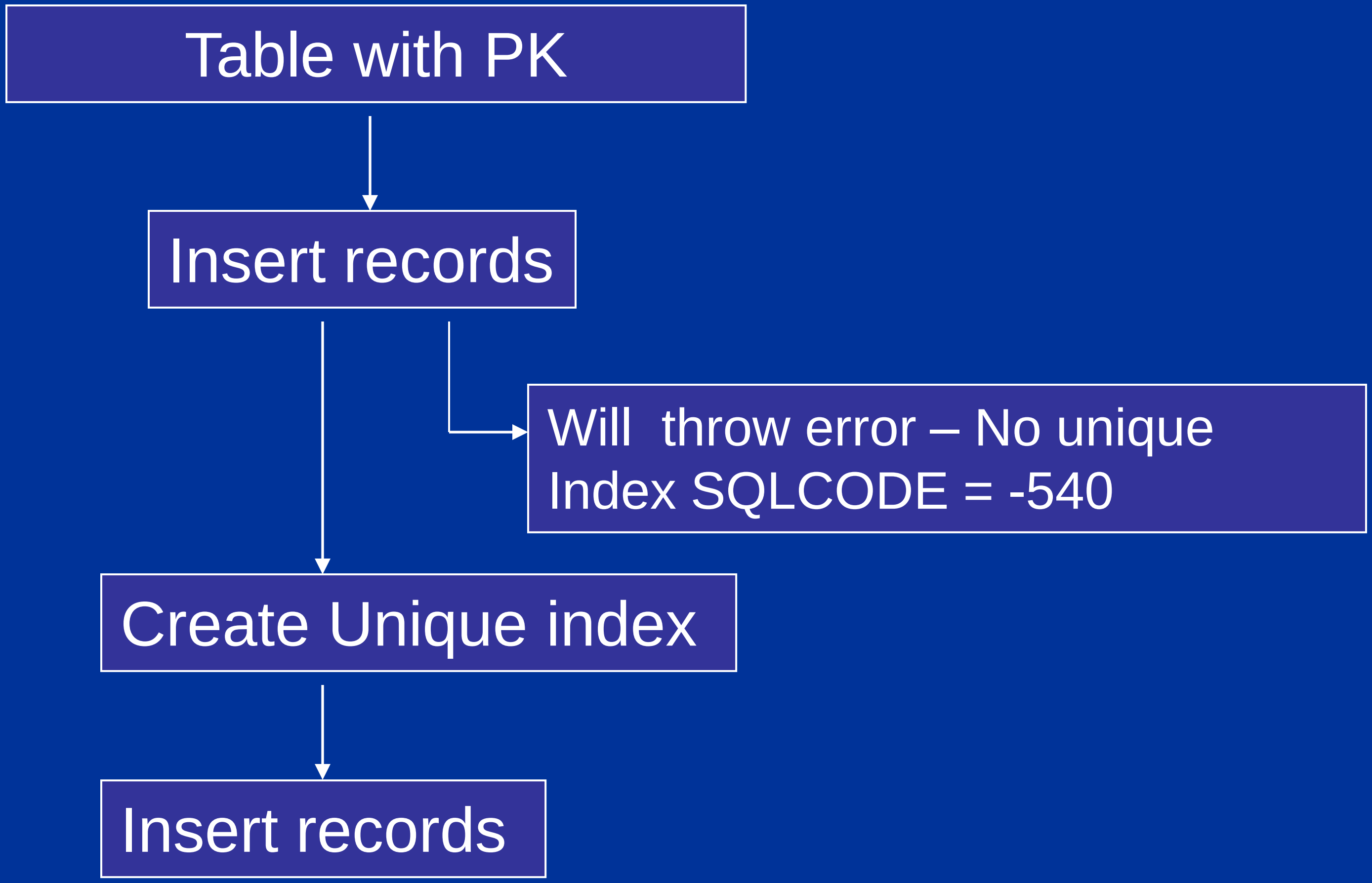
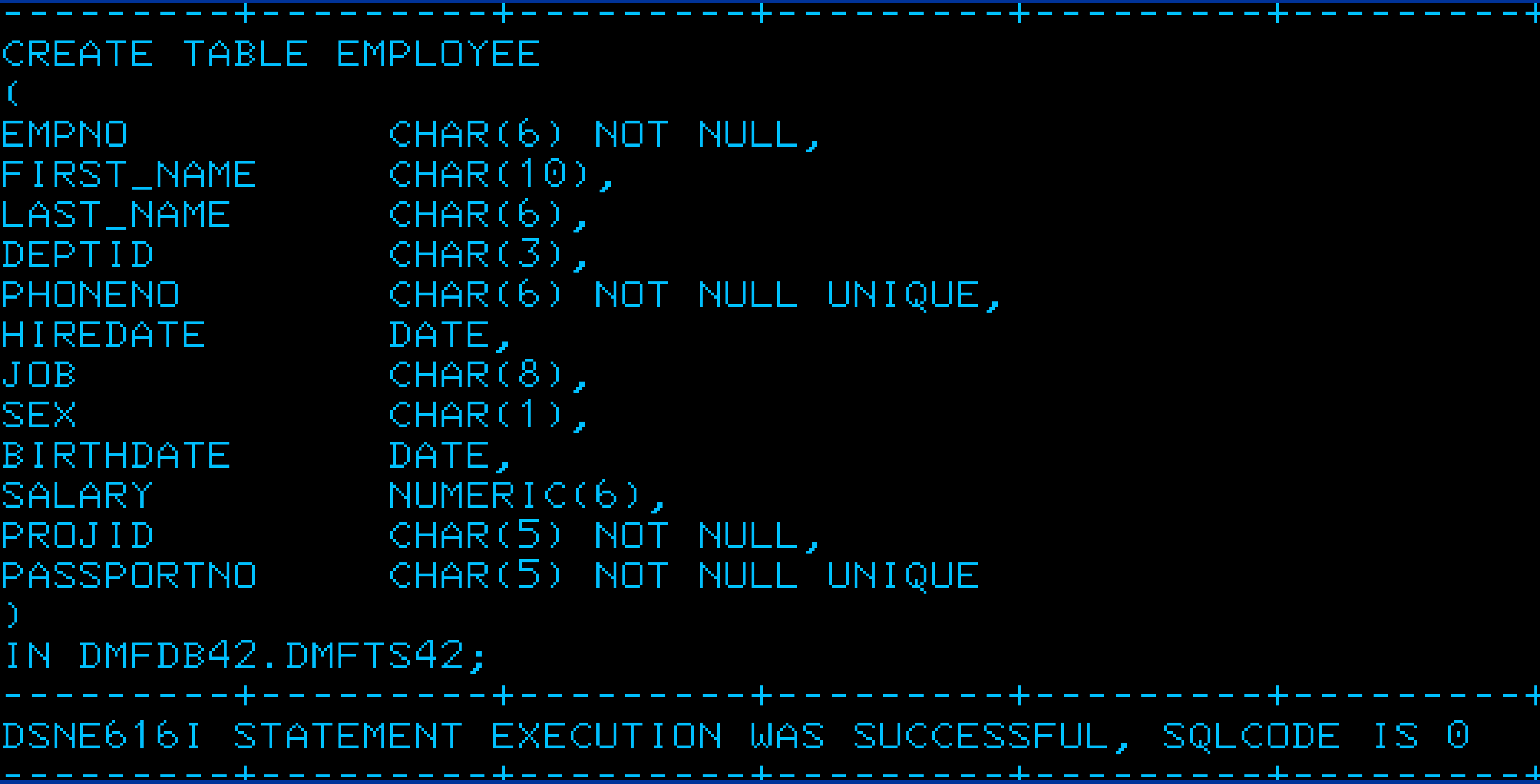


Table with PK
Insert records
Will throw error – No unique Index SQLCODE = -540
Create Unique index
Insert records

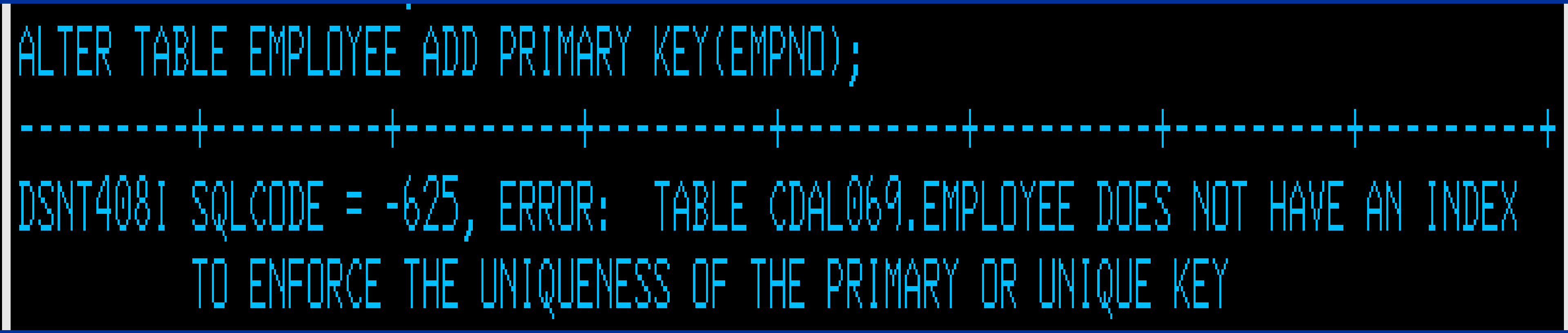
DEMOs

Create table employee without PK

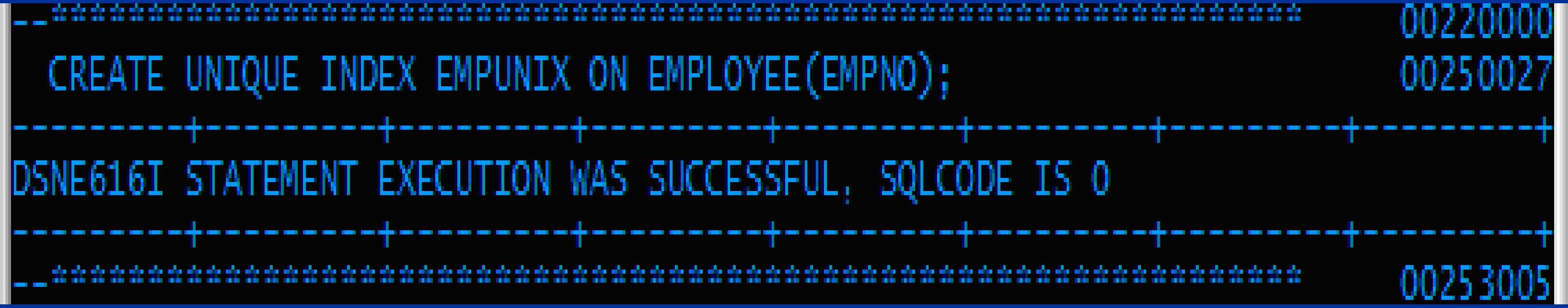


```
---------+---------+---------+---------+---------+---------+
CREATE TABLE EMPLOYEE
(
EMPNO          CHAR(6) NOT NULL,
FIRST_NAME     CHAR(10),
LAST_NAME      CHAR(6),
DEPTID         CHAR(3),
PHONENO        CHAR(6) NOT NULL UNIQUE,
HIREDATE       DATE,
JOB            CHAR(8),
SEX            CHAR(1),
BIRTHDATE      DATE,
SALARY         NUMERIC(6),
PROJID         CHAR(5) NOT NULL,
PASSPORTNO     CHAR(5) NOT NULL UNIQUE
)
IN DMFDB42.DMFTS42;
---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+
```

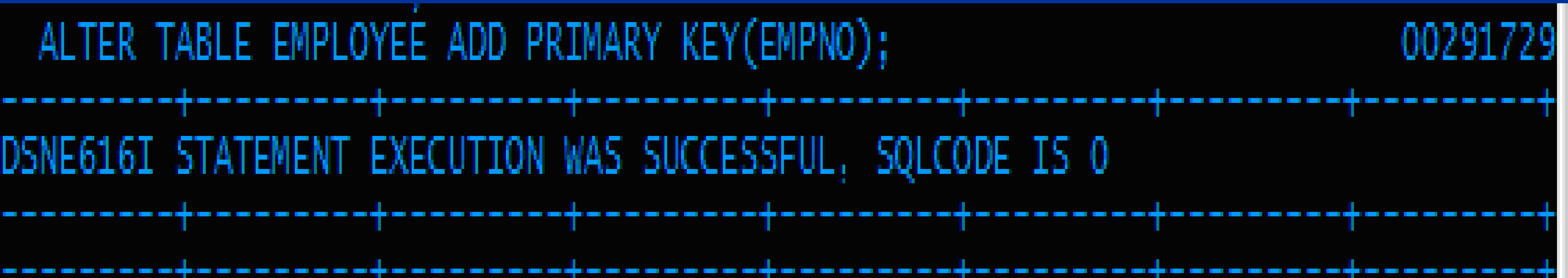
Altering a table to add PK



```
ALTER TABLE EMPLOYEE ADD PRIMARY KEY(EMPNO);
---------+---------+---------+---------+---------+---------+---------+---------+
DSNT408I SQLCODE = -625, ERROR:  TABLE CDAL069.EMPLOYEE DOES NOT HAVE AN INDEX
         TO ENFORCE THE UNIQUENESS OF THE PRIMARY OR UNIQUE KEY
```

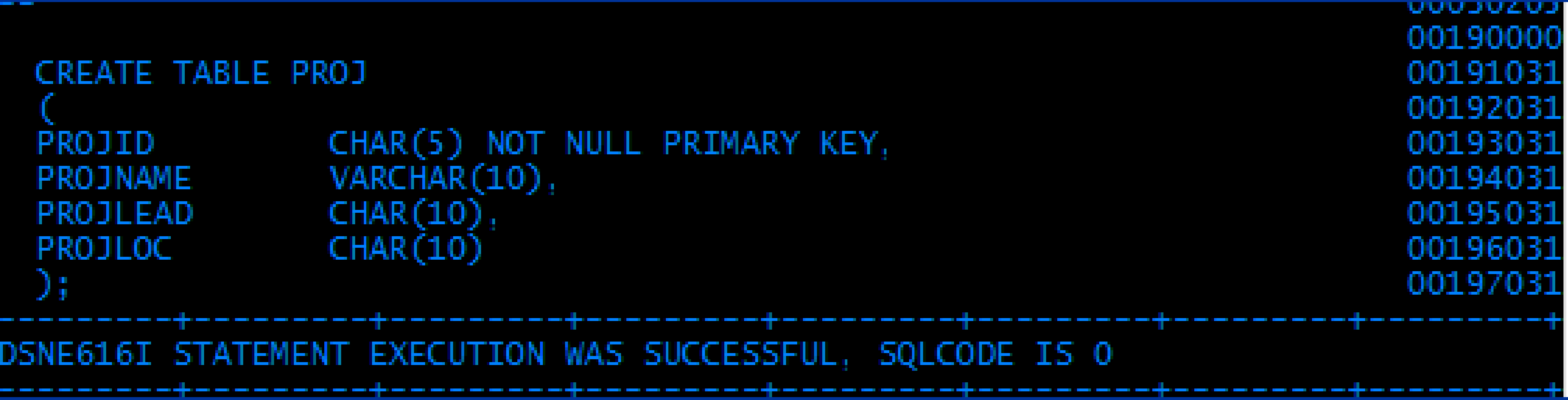


```
--*****************************************************************     00220000
  CREATE UNIQUE INDEX EMPUNIX ON EMPLOYEE(EMPNO);                         00250027
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
--*****************************************************************     00253005
```

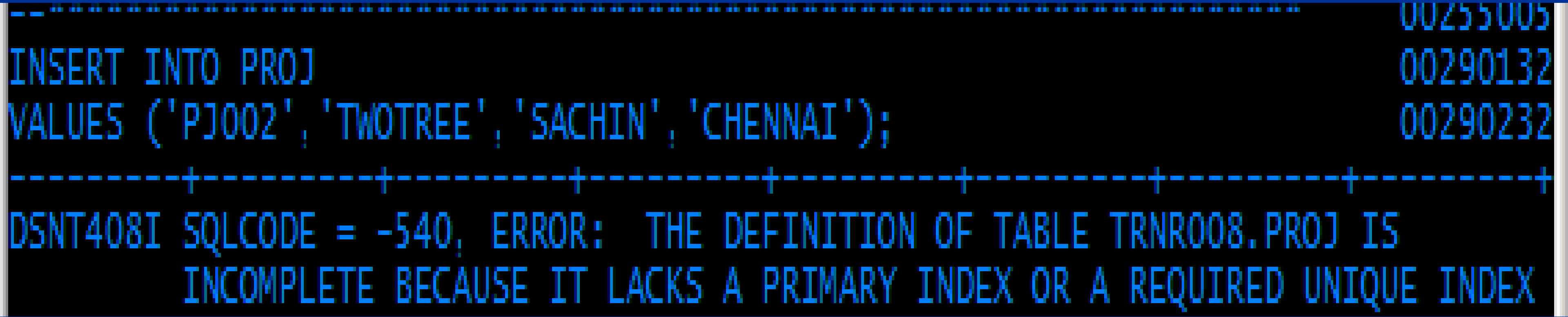


```
  ALTER TABLE EMPLOYEE ADD PRIMARY KEY(EMPNO);                            00291729
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
---------+---------+---------+---------+---------+---------+---------+---------+
```

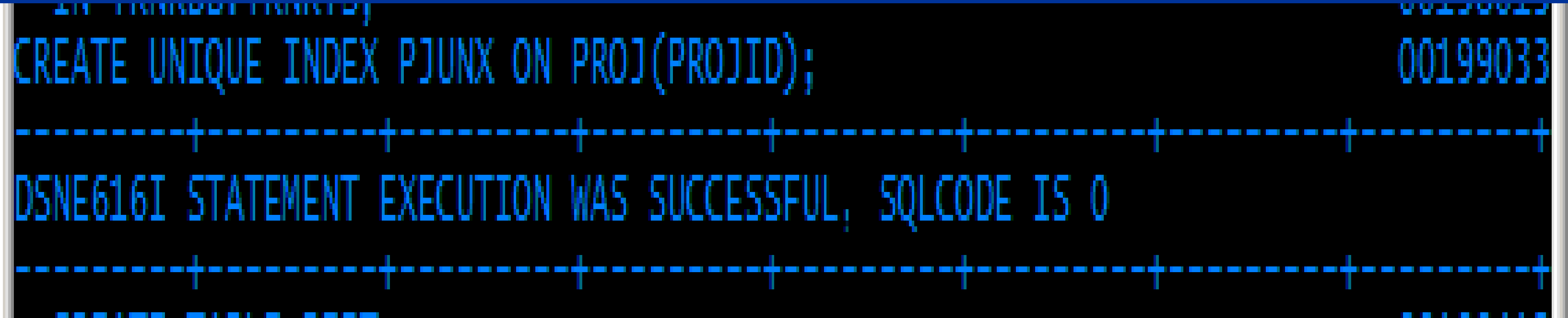
Table with PK



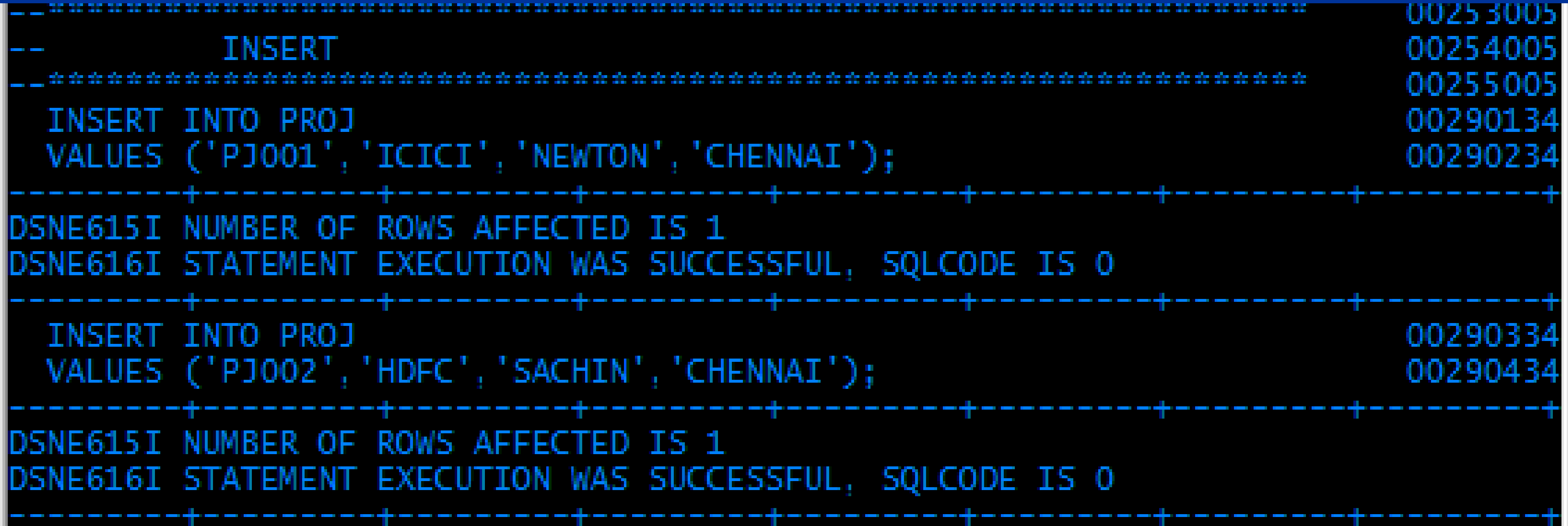
```
                                                                               00190000
 CREATE TABLE PROJ                                                             00191031
 (                                                                             00192031
 PROJID         CHAR(5) NOT NULL PRIMARY KEY,                                  00193031
 PROJNAME       VARCHAR(10),                                                   00194031
 PROJLEAD       CHAR(10),                                                      00195031
 PROJLOC        CHAR(10)                                                       00196031
 );                                                                            00197031
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```



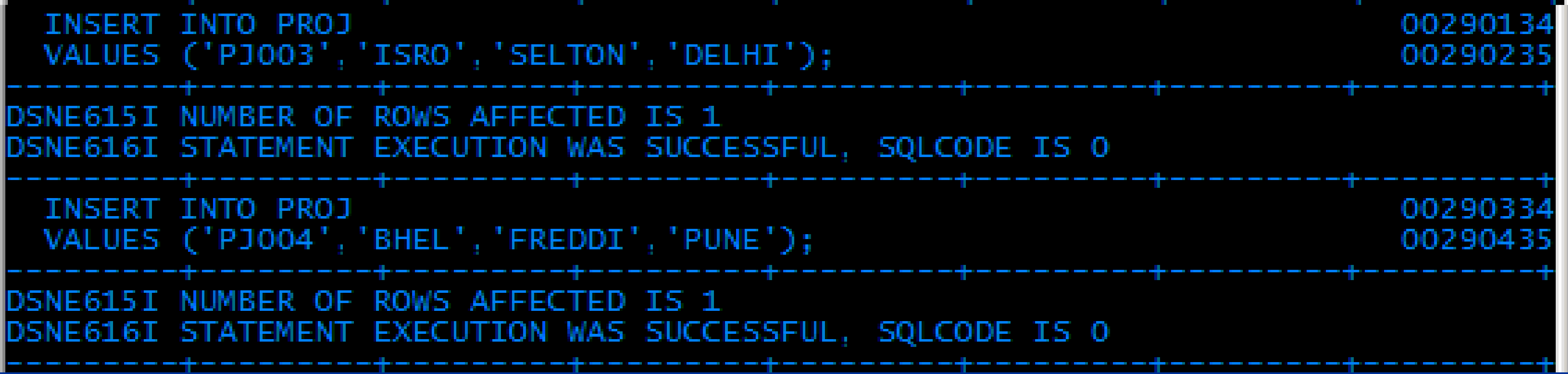
```
INSERT INTO PROJ                                                               00290132
VALUES ('PJ002','TWOTREE','SACHIN','CHENNAI');                                 00290232
---------+---------+---------+---------+---------+---------+---------+---------+
DSNT408I SQLCODE = -540, ERROR:  THE DEFINITION OF TABLE TRNR008.PROJ IS
         INCOMPLETE BECAUSE IT LACKS A PRIMARY INDEX OR A REQUIRED UNIQUE INDEX
```



```
CREATE UNIQUE INDEX PJUNX ON PROJ(PROJID);                                     00199033
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

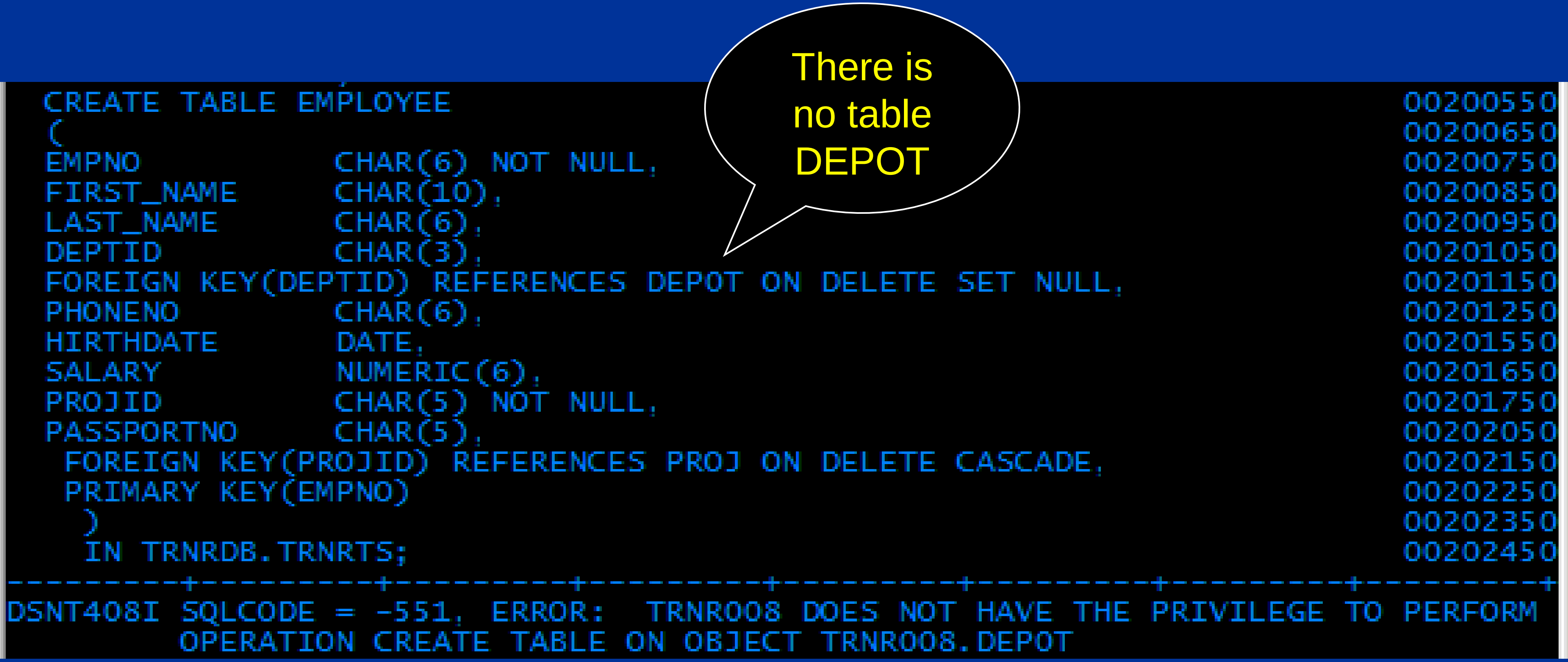


```
--*************************************************************************    00253005
--         INSERT                                                              00254005
--*************************************************************************    00255005
  INSERT INTO PROJ                                                             00290134
  VALUES ('PJ001','ICICI','NEWTON','CHENNAI');                                 00290234
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
  INSERT INTO PROJ                                                             00290334
  VALUES ('PJ002','HDFC','SACHIN','CHENNAI');                                  00290434
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

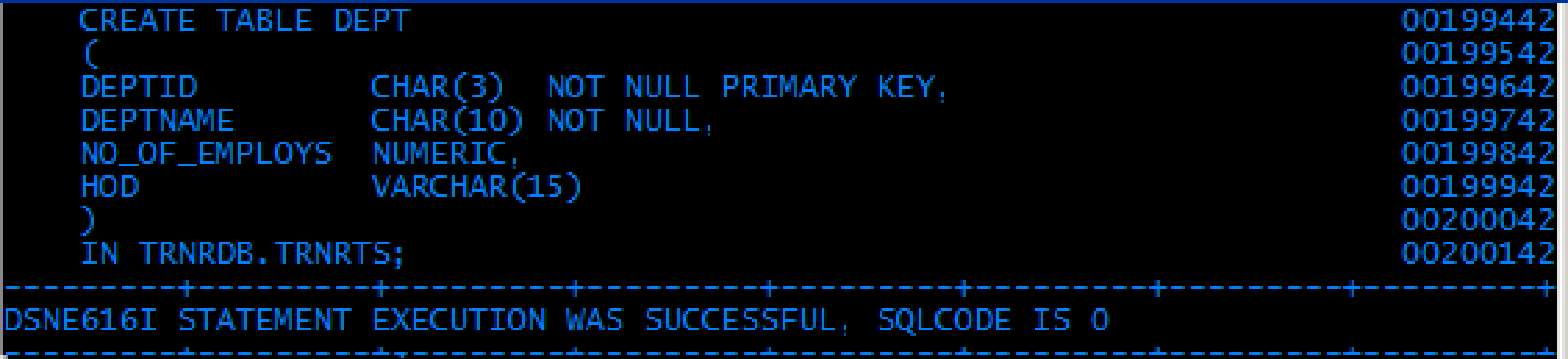


```
  INSERT INTO PROJ                                                             00290134
  VALUES ('PJ003','ISRO','SELTON','DELHI');                                    00290235
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
  INSERT INTO PROJ                                                             00290334
  VALUES ('PJ004','BHEL','FREDDI','PUNE');                                     00290435
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

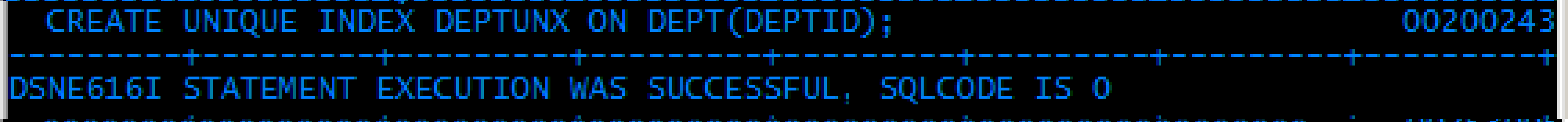
Table with foreign key that references a table which does not exist.



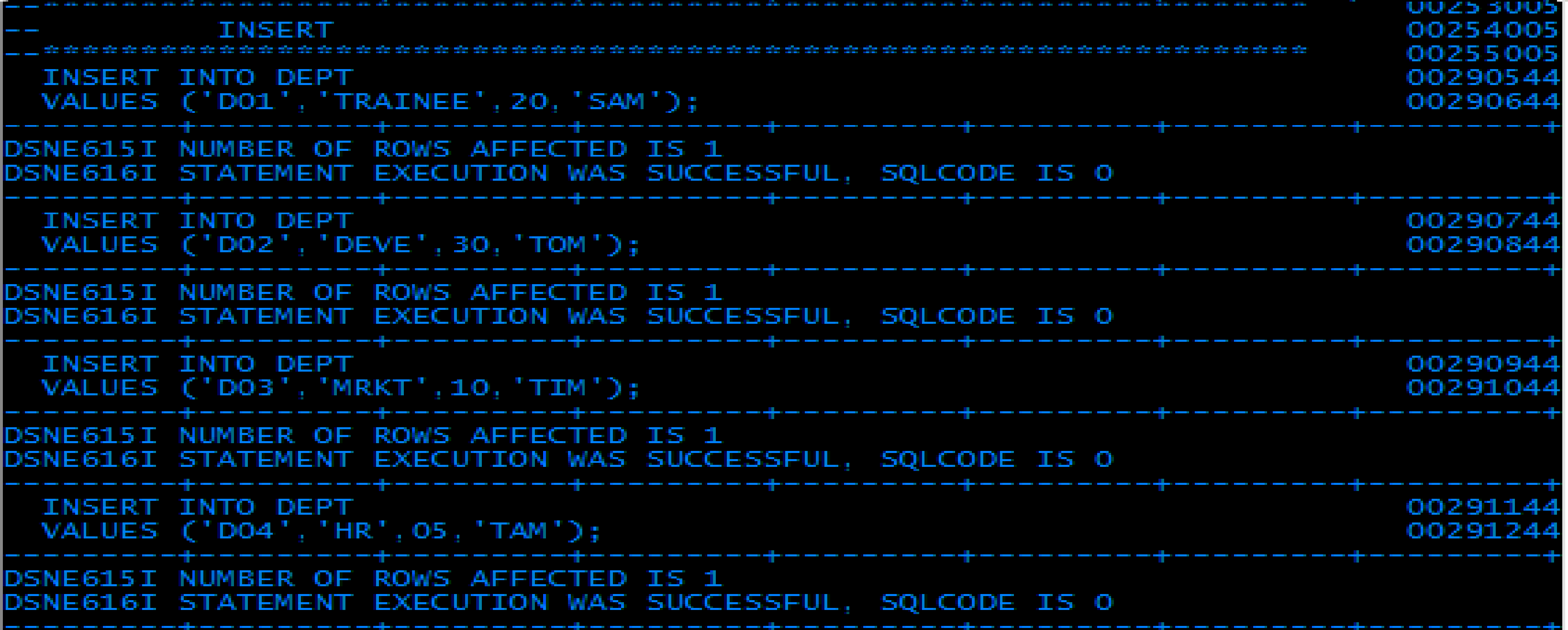
```
  CREATE TABLE EMPLOYEE                                                 00200550
  (                                                                     00200650
  EMPNO           CHAR(6) NOT NULL,                                     00200750
  FIRST_NAME      CHAR(10),                                             00200850
  LAST_NAME       CHAR(6),                                              00200950
  DEPTID          CHAR(3),                                              00201050
  FOREIGN KEY(DEPTID) REFERENCES DEPOT ON DELETE SET NULL,              00201150
  PHONENO         CHAR(6),                                              00201250
  HIRTHDATE       DATE,                                                 00201550
  SALARY          NUMERIC(6),                                           00201650
  PROJID          CHAR(5) NOT NULL,                                     00201750
  PASSPORTNO      CHAR(5),                                              00202050
   FOREIGN KEY(PROJID) REFERENCES PROJ ON DELETE CASCADE,               00202150
   PRIMARY KEY(EMPNO)                                                   00202250
    )                                                                   00202350
    IN TRNRDB.TRNRTS;                                                   00202450
---------+---------+---------+---------+---------+---------+---------+---------+
DSNT408I SQLCODE = -551, ERROR:  TRNR008 DOES NOT HAVE THE PRIVILEGE TO PERFORM
         OPERATION CREATE TABLE ON OBJECT TRNR008.DEPOT
```



```
    CREATE TABLE DEPT                                                   00199442
    (                                                                   00199542
    DEPTID          CHAR(3)  NOT NULL PRIMARY KEY,                      00199642
    DEPTNAME        CHAR(10) NOT NULL,                                  00199742
    NO_OF_EMPLOYS   NUMERIC,                                            00199842
    HOD             VARCHAR(15)                                         00199942
    )                                                                   00200042
    IN TRNRDB.TRNRTS;                                                   00200142
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

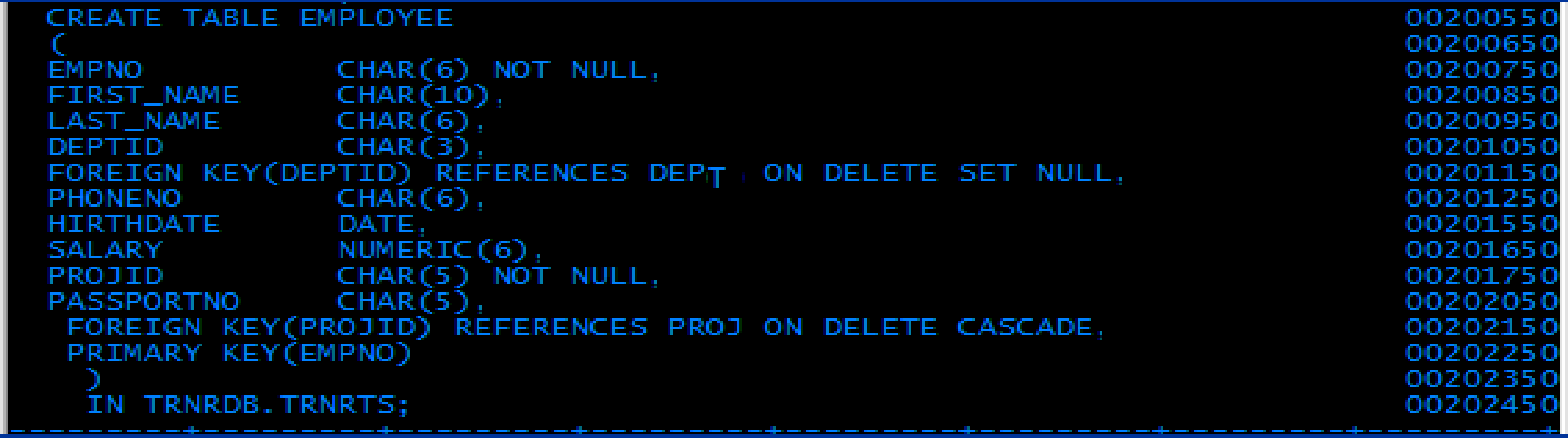


```
  CREATE UNIQUE INDEX DEPTUNX ON DEPT(DEPTID);                          00200243
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
```

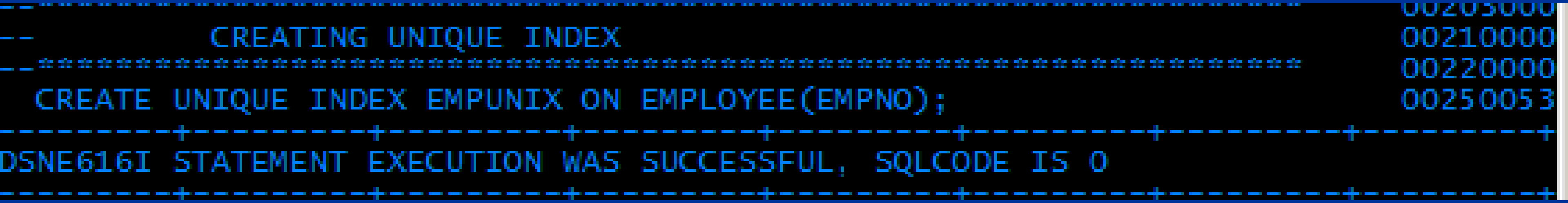


```
--*****************************************************************     00253005
--          INSERT                                                      00254005
--*****************************************************************     00255005
  INSERT INTO DEPT                                                      00290544
  VALUES ('D01','TRAINEE',20,'SAM');                                    00290644
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
  INSERT INTO DEPT                                                      00290744
  VALUES ('D02','DEVE',30,'TOM');                                       00290844
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
  INSERT INTO DEPT                                                      00290944
  VALUES ('D03','MRKT',10,'TIM');                                       00291044
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
  INSERT INTO DEPT                                                      00291144
  VALUES ('D04','HR',05,'TAM');                                         00291244
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

Creating a table with proper parent table

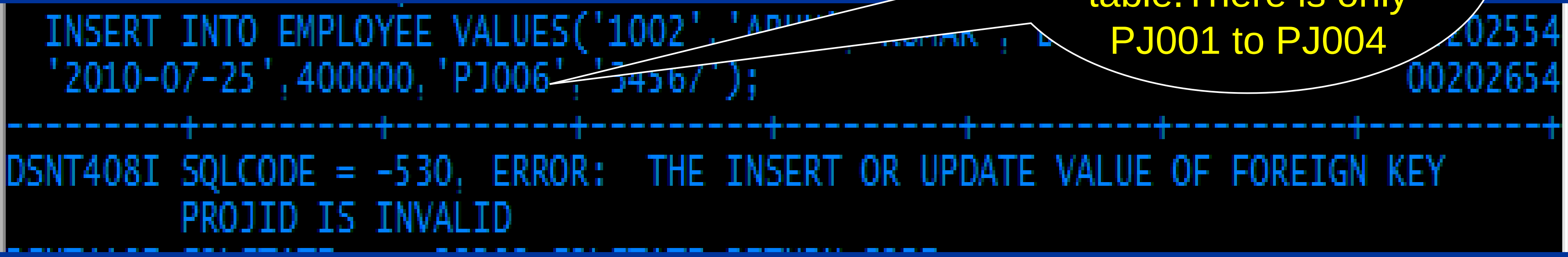


```
  CREATE TABLE EMPLOYEE
   (
   EMPNO            CHAR(6) NOT NULL,
   FIRST_NAME       CHAR(10),
   LAST_NAME        CHAR(6),
   DEPTID           CHAR(3),
   FOREIGN KEY(DEPTID) REFERENCES DEPT ON DELETE SET NULL,
   PHONENO          CHAR(6),
   HIRTHDATE        DATE,
   SALARY           NUMERIC(6),
   PROJID           CHAR(5) NOT NULL,
   PASSPORTNO       CHAR(5),
    FOREIGN KEY(PROJID) REFERENCES PROJ ON DELETE CASCADE,
    PRIMARY KEY(EMPNO)
     )
     IN TRNRDB.TRNRTS;
---------+---------+---------+---------+---------+---------+---------+---------+
```



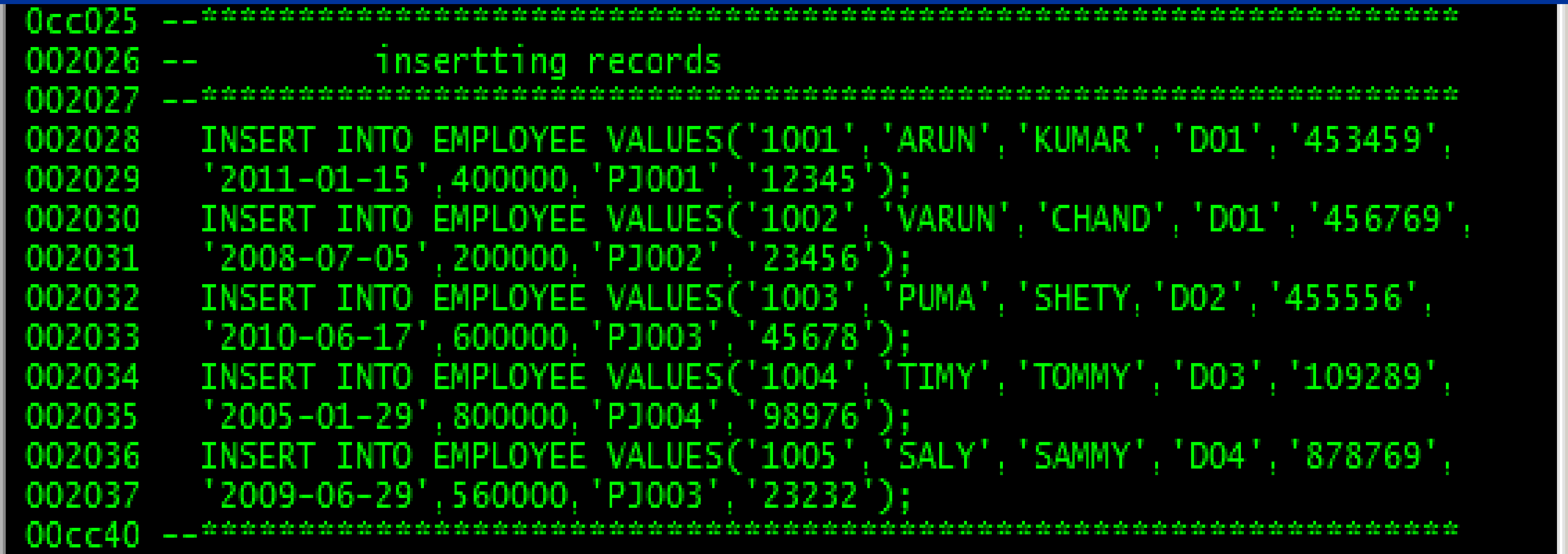
```
--**********************************************************************
--          CREATING UNIQUE INDEX
--**********************************************************************
  CREATE UNIQUE INDEX EMPUNIX ON EMPLOYEE(EMPNO);
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```

INSERTing values into a child table when the value is not in parent table



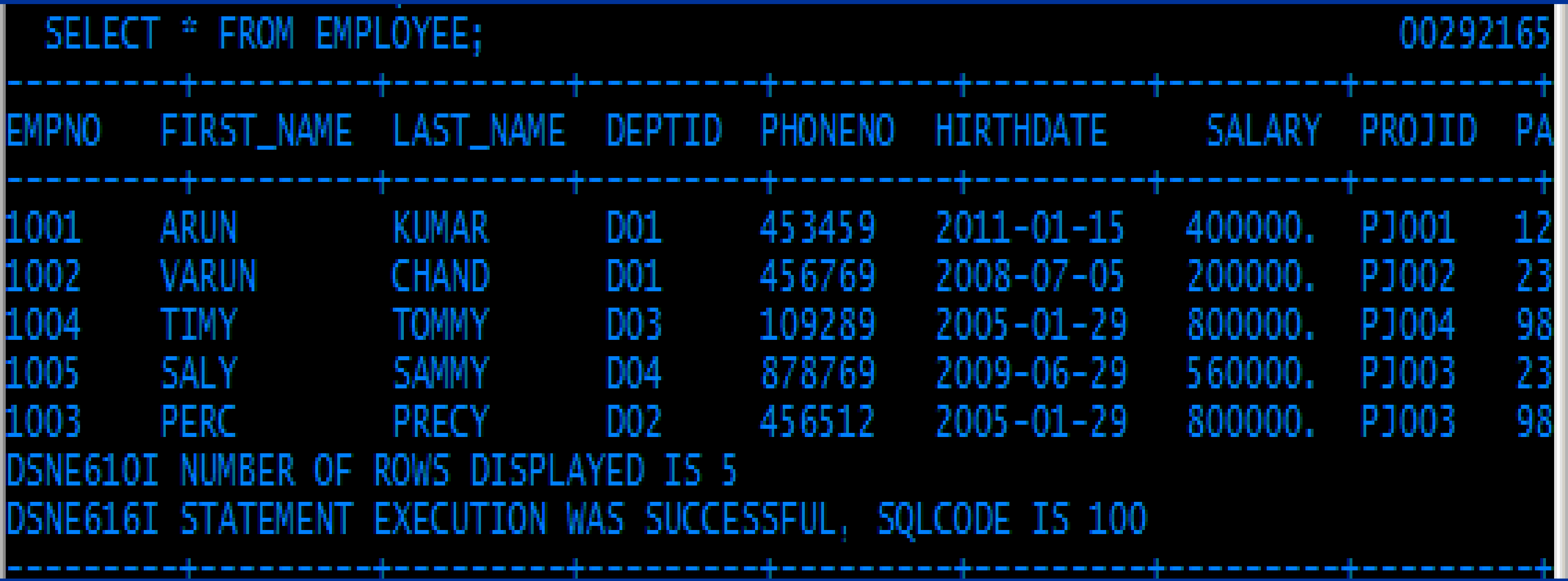
```
  INSERT INTO EMPLOYEE VALUES('1002','ARU        ','KUMAR',
  '2010-07-25',400000,'PJ006','34567');
---------+---------+---------+---------+---------+---------+---------+---------+
DSNT408I SQLCODE = -530, ERROR:  THE INSERT OR UPDATE VALUE OF FOREIGN KEY
         PROJID IS INVALID
```

There is no projid PJ006 in PROJ table.There is only PJ001 to PJ004



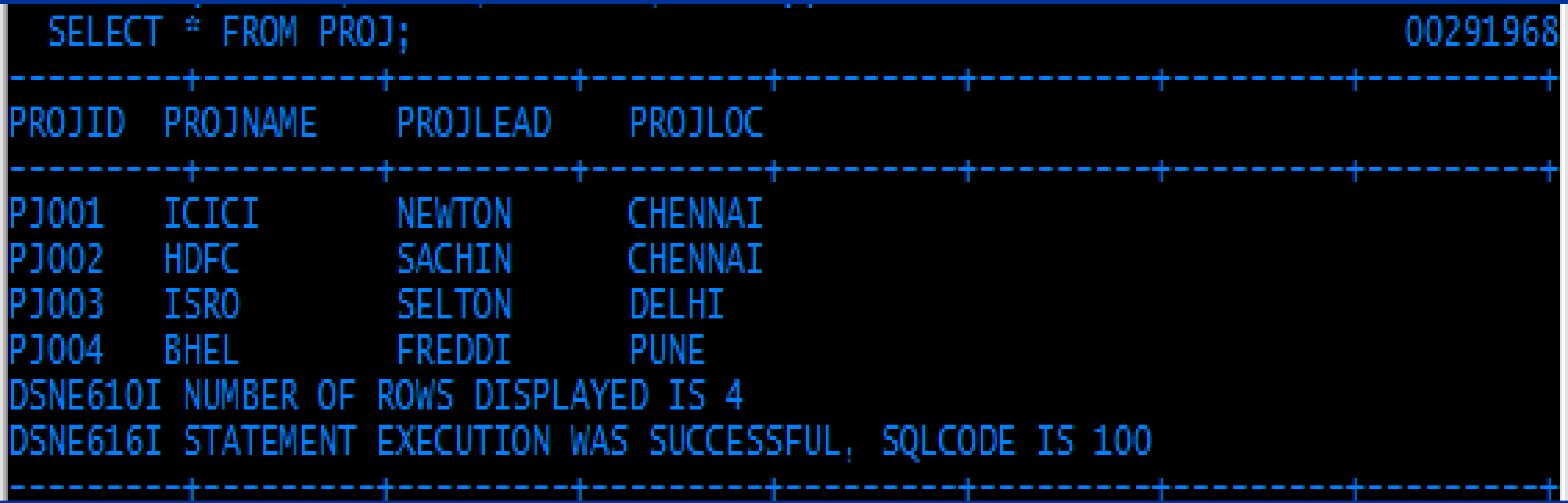
```
--*****************************************************************
--          insertting records
--*****************************************************************
  INSERT INTO EMPLOYEE VALUES('1001','ARUN','KUMAR','D01','453459',
  '2011-01-15',400000,'PJ001','12345');
  INSERT INTO EMPLOYEE VALUES('1002','VARUN','CHAND','D01','456769',
  '2008-07-05',200000,'PJ002','23456');
  INSERT INTO EMPLOYEE VALUES('1003','PUMA','SHETY,'D02','455556',
  '2010-06-17',600000,'PJ003','45678');
  INSERT INTO EMPLOYEE VALUES('1004','TIMY','TOMMY','D03','109289',
  '2005-01-29',800000,'PJ004','98976');
  INSERT INTO EMPLOYEE VALUES('1005','SALY','SAMMY','D04','878769',
  '2009-06-29',560000,'PJ003','23232');
--*****************************************************************
```

Content of the tables EMPLOYEE



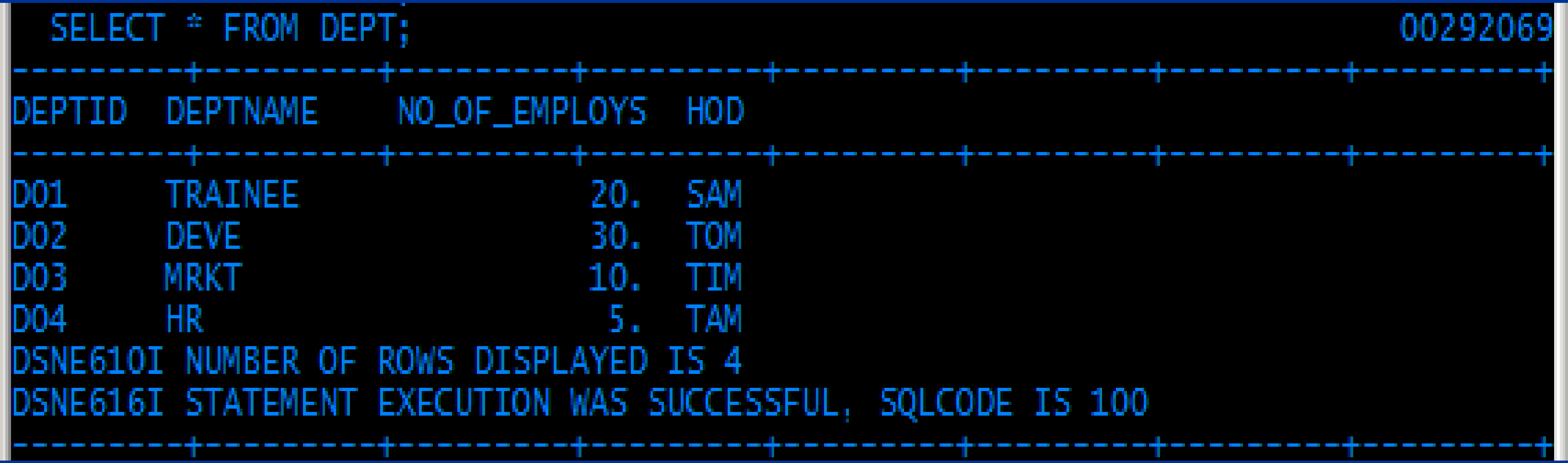
```
  SELECT * FROM EMPLOYEE;                                                 00292165
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME  LAST_NAME  DEPTID  PHONENO  HIRTHDATE     SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN        KUMAR      D01     453459   2011-01-15  400000.  PJ001   12
1002    VARUN       CHAND      D01     456769   2008-07-05  200000.  PJ002   23
1004    TIMY        TOMMY      D03     109289   2005-01-29  800000.  PJ004   98
1005    SALY        SAMMY      D04     878769   2009-06-29  560000.  PJ003   23
1003    PERC        PRECY      D02     456512   2005-01-29  800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Content of the tables PROJ



```
  SELECT * FROM PROJ;                                                      00291968
---------+---------+---------+---------+---------+---------+---------+---------+
PROJID  PROJNAME    PROJLEAD    PROJLOC
---------+---------+---------+---------+---------+---------+---------+---------+
PJ001   ICICI       NEWTON      CHENNAI
PJ002   HDFC        SACHIN      CHENNAI
PJ003   ISRO        SELTON      DELHI
PJ004   BHEL        FREDDI      PUNE
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Content of the tables DEPT



```
  SELECT * FROM DEPT;                                                              00292069
---------+---------+---------+---------+---------+---------+---------+---------+
DEPTID  DEPTNAME    NO_OF_EMPLOYS  HOD
---------+---------+---------+---------+---------+---------+---------+---------+
D01     TRAINEE               20.  SAM
D02     DEVE                  30.  TOM
D03     MRKT                  10.  TIM
D04     HR                     5.  TAM
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

2.DATA MANIPULATION LANGUAGE (DML)

1. Insert

2. Update

3. Delete

Condtional usage of the above commands associates WHERE Clause.

2.DATA MANIPULATION LANGUAGE (DML)

1.INSERT

Inserting New Record into a Table

- Inserts values to the columns.
- The values are positional, so that the first value goes in to the first column of the table.

Syntax:

```
        INSERT INTO <Table_name>
            Values ( col1_value, col2,Value,………….)

        INSERT INTO DEPT
            VALUES(‘D001’,’Finance’);
```

2.DATA MANIPULATION LANGUAGE (DML)

Inserting New Record into a Table

While inserting the records, Sequence of the values can vary corresponding to column list mentioned in the SQL rather than as in table.

Syntax:

```
INSERT INTO <Table_name> (Col3,clo1,…..)
          Values ( col3_value, col1_Value,…………)


INSERT INTO DEPT( Dept_name, Dept_No )
     VALUES('Finance','D001');
```

2.DATA MANIPULATION LANGUAGE (DML)

Inserting Null values into the Table.

NULL values can be inserted into the columns by NULL key word and even can insert with the name of the column.

Syntax to insert NULL value in column 2:

```
INSERT INTO <Table_name> (col3,col1,col4)
        Values ( col3_value, col1_Value,col4_val)

INSERT INTO <Table_name>
        Values ( col1_value,NULL,col3_Value, col4_val)
```

TABLE1

TABLE2

To copy all records from table1 to table2

Insert into table2 values(select * from table1);

2.DML

2.UPDATE

Modifying Value (s) of a column

UPDATE command will modify the values in the existing column.

Syntax:

```
UPDATE <table name>
        SET <column name>=value/expression
Update employee
        set salary = salary + salary * 0.6
Update employee
        set mobile_no = '9841231231'
```

Note:

This statement will update the values (mobile numbers) of all records in the table.

2.DML

MODIFYING MORE THAN ONE column VALUE IN A ROW

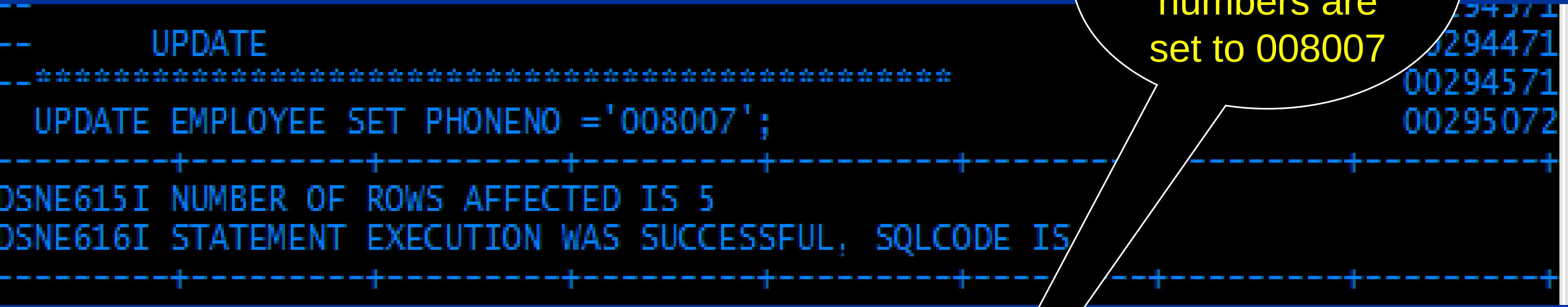
```
UPDATE <table name>
      SET  col1 =  value/expression,
              col2 = value/expression ;


UPDATE EMP
      SET SALARY = salary + salary * 0.6 ,
            Mobile_no = ‘9841231231’;
```

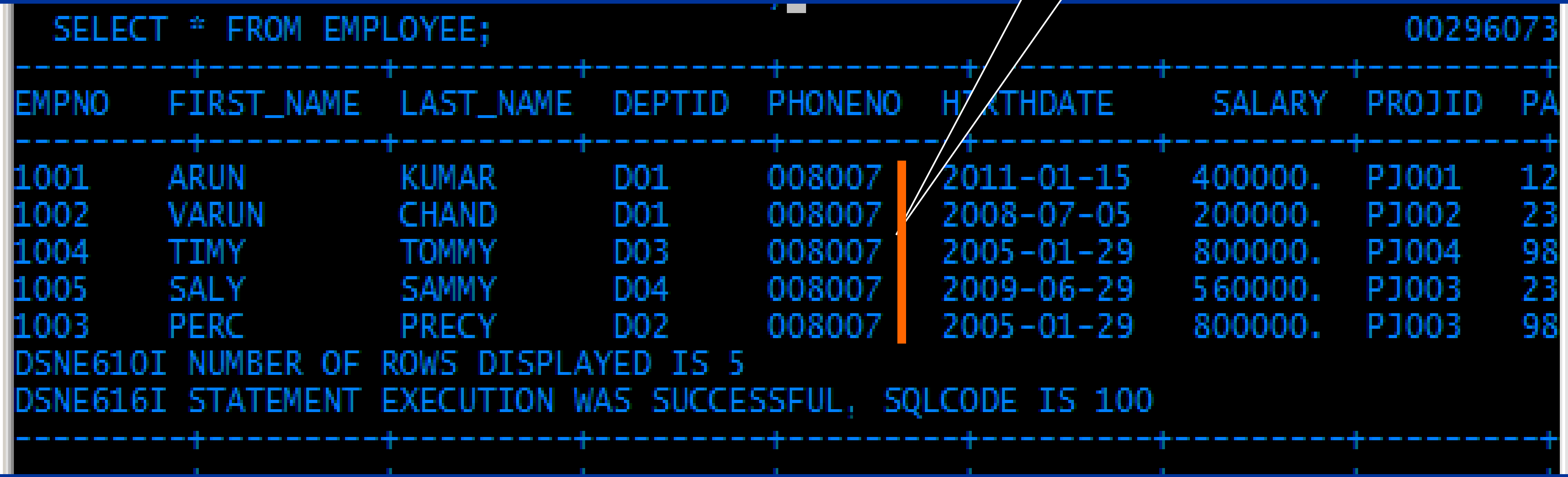
2.DML

Update Screen Shot

All the telephone numbers are set to 008007



```
--                                                                   94571
--      UPDATE                                                     00294471
--*****************************************************          00294571
 UPDATE EMPLOYEE SET PHONENO ='008007';                            00295072
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS
---------+---------+---------+---------+---------+---------+---------+---------+
```



```
 SELECT * FROM EMPLOYEE;                                             00296073
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME  LAST_NAME  DEPTID  PHONENO  HIRTHDATE    SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN        KUMAR      D01     008007   2011-01-15  400000.  PJ001   12
1002    VARUN       CHAND      D01     008007   2008-07-05  200000.  PJ002   23
1004    TIMY        TOMMY      D03     008007   2005-01-29  800000.  PJ004   98
1005    SALY        SAMMY      D04     008007   2009-06-29  560000.  PJ003   23
1003    PERC        PRECY      D02     008007   2005-01-29  800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

2.DML

3.DELETE
Deleting the Records in a table.

DELETE command will delete the records in the table.

Syntax

```
DELETE from <table_ name>
```

Example

```
Delete  from employee
```

This statement will remove all the records from the table employee.

2.DQL

1. SELECT

Retrieval of ALL Columns

SELECTting all the columns from the table.

<u>Syntax:</u>

```
        Select * from <tablename>;
```

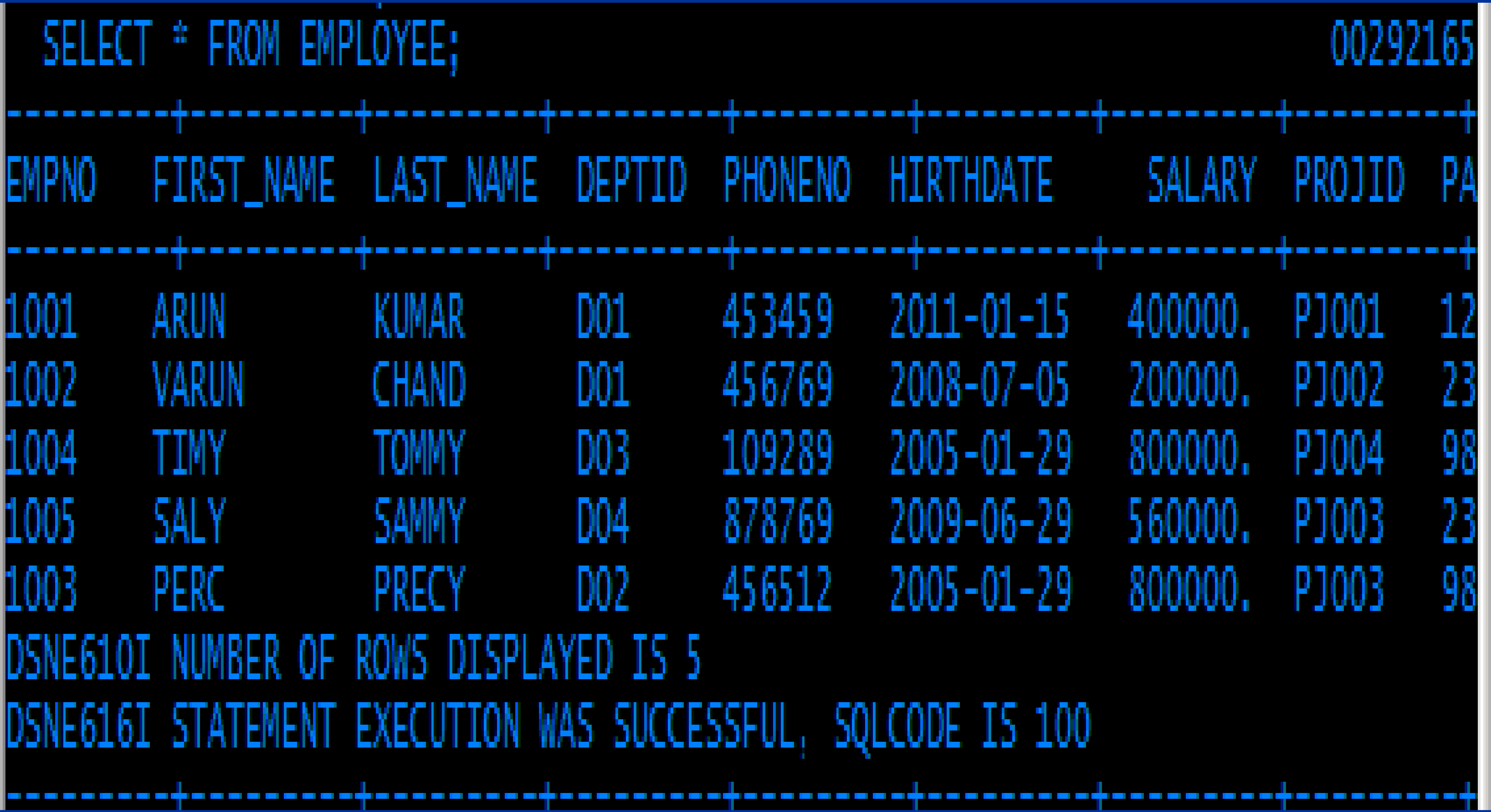
- → all the columns.
- where

Example:

```
        Select * from Employee;
```

2.DML

4. SELECT



```
  SELECT * FROM EMPLOYEE;                                                  00292165
--------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME  LAST_NAME  DEPTID  PHONENO  HIRTHDATE     SALARY  PROJID  PA
--------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN        KUMAR      D01     453459   2011-01-15   400000.  PJ001   12
1002    VARUN       CHAND      D01     456769   2008-07-05   200000.  PJ002   23
1004    TIMY        TOMMY      D03     109289   2005-01-29   800000.  PJ004   98
1005    SALY        SAMMY      D04     878769   2009-06-29   560000.  PJ003   23
1003    PERC        PRECY      D02     456512   2005-01-29   800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
--------+---------+---------+---------+---------+---------+---------+---------+
```

2.DML

4. SELECT

SELECTing SELECTIVE columns from the table.

Syntax:

```
Select col1, col2,col3 from <table name>
```

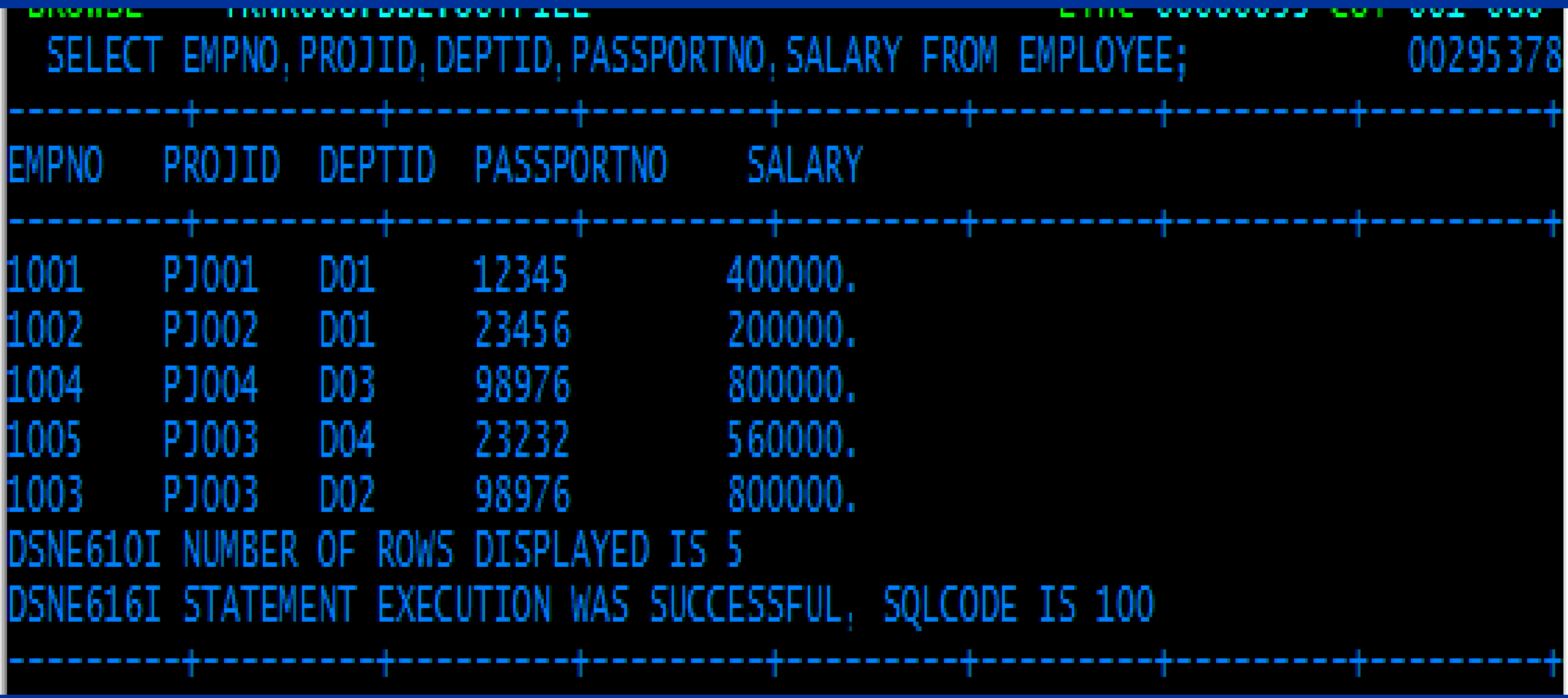
Example:

```
Select
     emp_code,
     emp_name,
     salary
from
     employee ;
```

2.DML

4. SELECT

SELECT EMPNO,PROJID,DEPTID,PASSPORTNO, SALARY from EMPLOYEE



```
 SELECT EMPNO,PROJID,DEPTID,PASSPORTNO,SALARY FROM EMPLOYEE;           00295378
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    PROJID  DEPTID  PASSPORTNO    SALARY
---------+---------+---------+---------+---------+---------+---------+---------+
1001     PJ001   D01     12345        400000.
1002     PJ002   D01     23456        200000.
1004     PJ004   D03     98976        800000.
1005     PJ003   D04     23232        560000.
1003     PJ003   D02     98976        800000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE “WHERE” CLAUSE

THE “WHERE” CLAUSE

The “where” clause can be given in any DML statement except INSERT statement.

That is Where clause comes along with Update, Delete and Select statement.

If Where clause is omitted in any statement, that operation will perform on all the records in the table.

THE “WHERE” CLAUSE

Various condition in Where clause

Condition:

- Relational Operators.
- Logical Operators.
- “Between” Operator
- “IN” operator
- “IS” operator
- “Like” Operator.

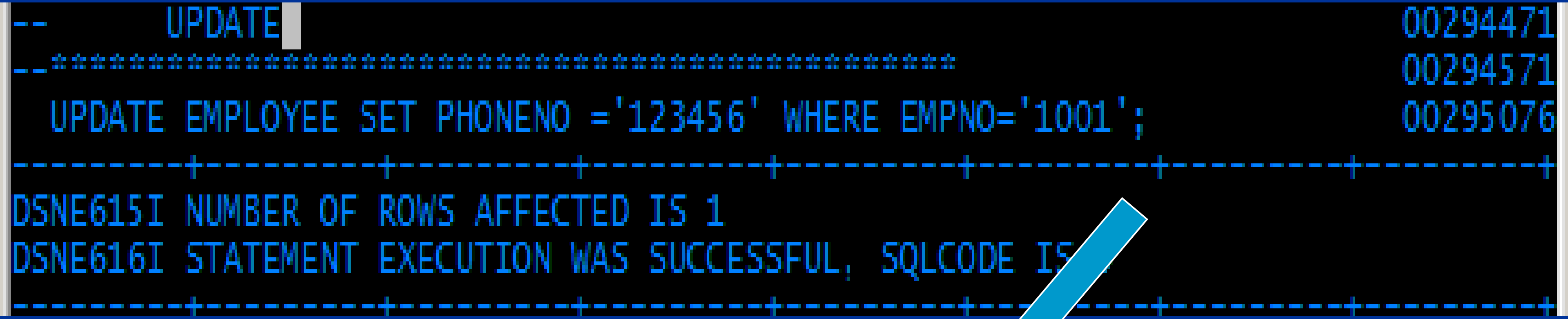
THE “WHERE” CLAUSE

Relational operators.

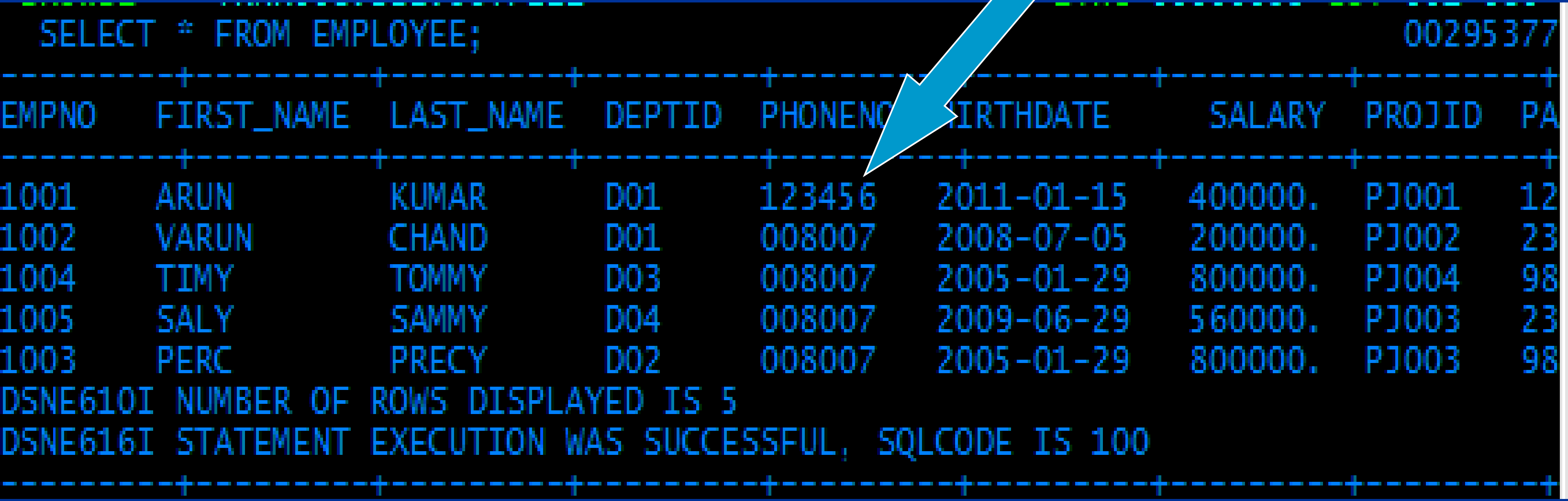
<	Less than
>	Greater than
=	Equal to
<=	Less than or Equal
>=	Greater than or equal.
<>	Not equal.

THE “WHERE” CLAUSE

Relational operators.



```
--      UPDATE                                                    00294471
--**********************************************                  00294571
  UPDATE EMPLOYEE SET PHONENO ='123456' WHERE EMPNO='1001';       00295076
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE615I NUMBER OF ROWS AFFECTED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
---------+---------+---------+---------+---------+---------+---------+---------+
```



```
 SELECT * FROM EMPLOYEE;                                          00295377
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME  LAST_NAME  DEPTID  PHONENO  BIRTHDATE     SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN        KUMAR      D01     123456   2011-01-15  400000.  PJ001   12
1002    VARUN       CHAND      D01     008007   2008-07-05  200000.  PJ002   23
1004    TIMY        TOMMY      D03     008007   2005-01-29  800000.  PJ004   98
1005    SALY        SAMMY      D04     008007   2009-06-29  560000.  PJ003   23
1003    PERC        PRECY      D02     008007   2005-01-29  800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE “WHERE” CLAUSE

Logical operators.

More than one Condition can be applied in where clause using logical Operators

AND

OR

NOT

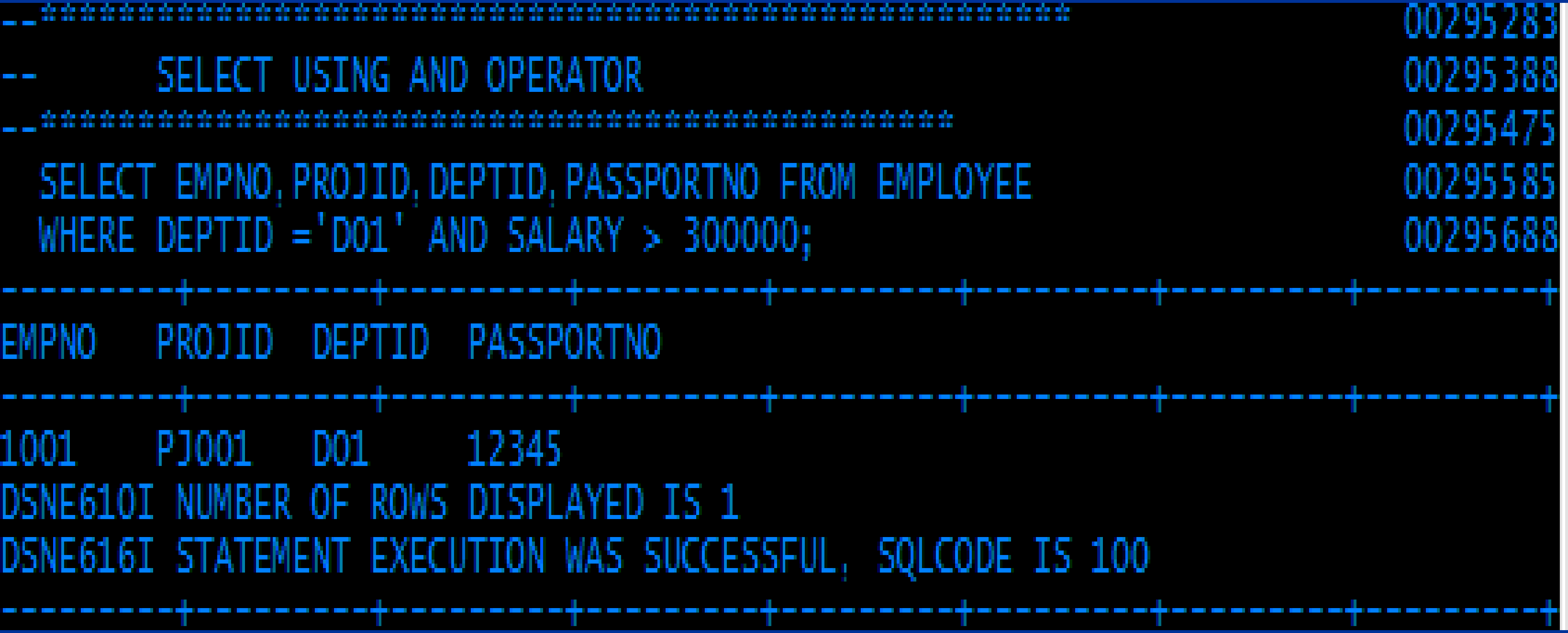
THE “WHERE” CLAUSE
Logical operators.

Example with Logical Operators.

```
1. UPDATE  employee
       set salary = salary + 5000
       Where  dept_no= ‘D001’  AND desig =‘manager’
2. UPDATE  employee
      set salary = salary + 5000
      where  salary >= 25000 OR desg=‘Supervisor’;
3.UPDATE employee
     set salary = salary + 5000
     where  desig NOT = ‘Manger’ ;
```

THE “WHERE” CLAUSE

Logical operators.



```
--*******************************************************               00295283
--      SELECT USING AND OPERATOR                                        00295388
--*************************************************                     00295475
  SELECT EMPNO,PROJID,DEPTID,PASSPORTNO FROM EMPLOYEE                    00295585
  WHERE DEPTID ='D01' AND SALARY > 300000;                               00295688
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   PROJID  DEPTID  PASSPORTNO
---------+---------+---------+---------+---------+---------+---------+---------+
1001    PJ001   D01     12345
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE “WHERE” CLAUSE

Between operator.

To choose the list of employees whose salary is greater than equal to 10000 and less than equal to 15000 we use the following command

```
SELECT * from employee
        Where salary >= 10000 AND salary <=15000
```

The above operation can be done with a single operator.

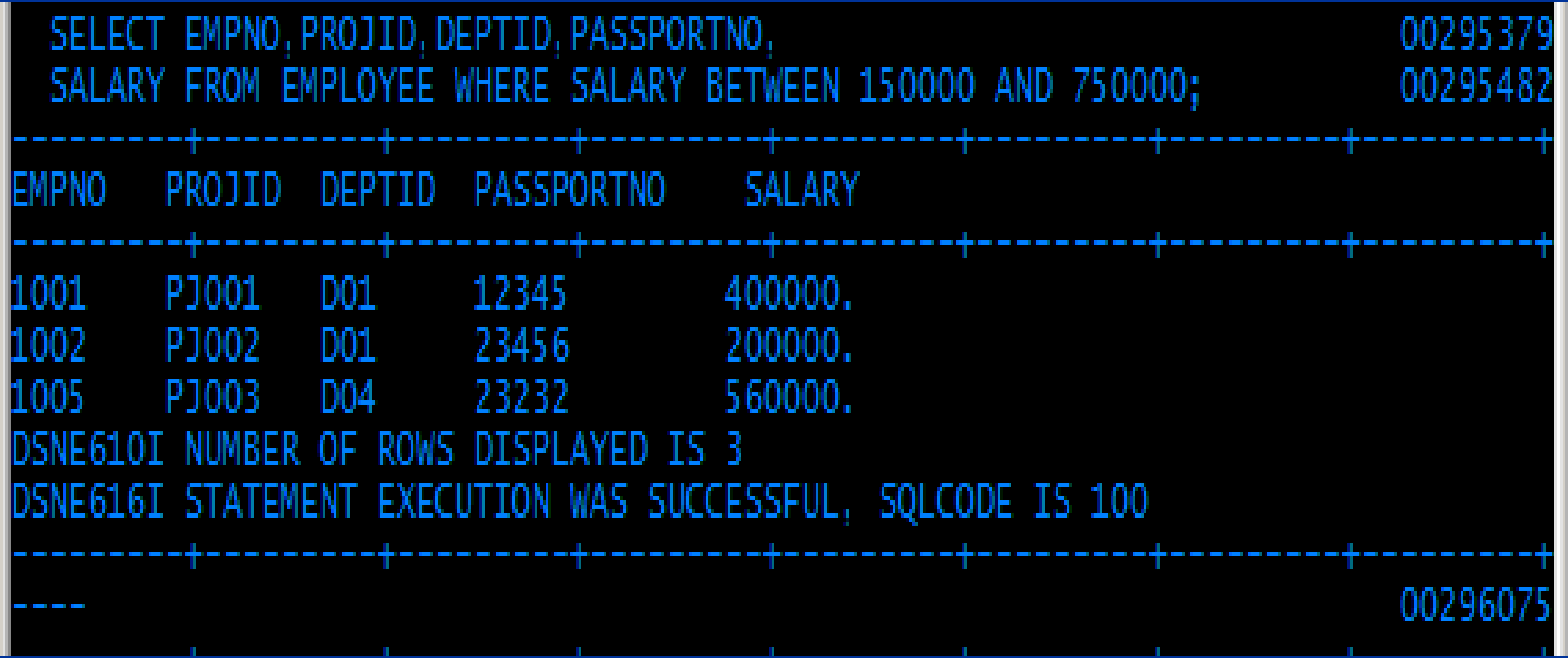
Between operator.

```
SELECT name,salary from employee
        Where salary between 10000 and 15000
```

NOTE: both the boundary values will be included.

THE “WHERE” CLAUSE

Between operator.



```
  SELECT EMPNO,PROJID,DEPTID,PASSPORTNO,                             00295379
  SALARY FROM EMPLOYEE WHERE SALARY BETWEEN 150000 AND 750000;       00295482
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    PROJID   DEPTID   PASSPORTNO     SALARY
---------+---------+---------+---------+---------+---------+---------+---------+
1001     PJ001    D01      12345         400000.
1002     PJ002    D01      23456         200000.
1005     PJ003    D04      23232         560000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
----                                                                 00296075
```

THE “WHERE” CLAUSE

IN operator.

Gender char(01) check (gender in(‘m’,’f’,’o’)

To choose from a list.

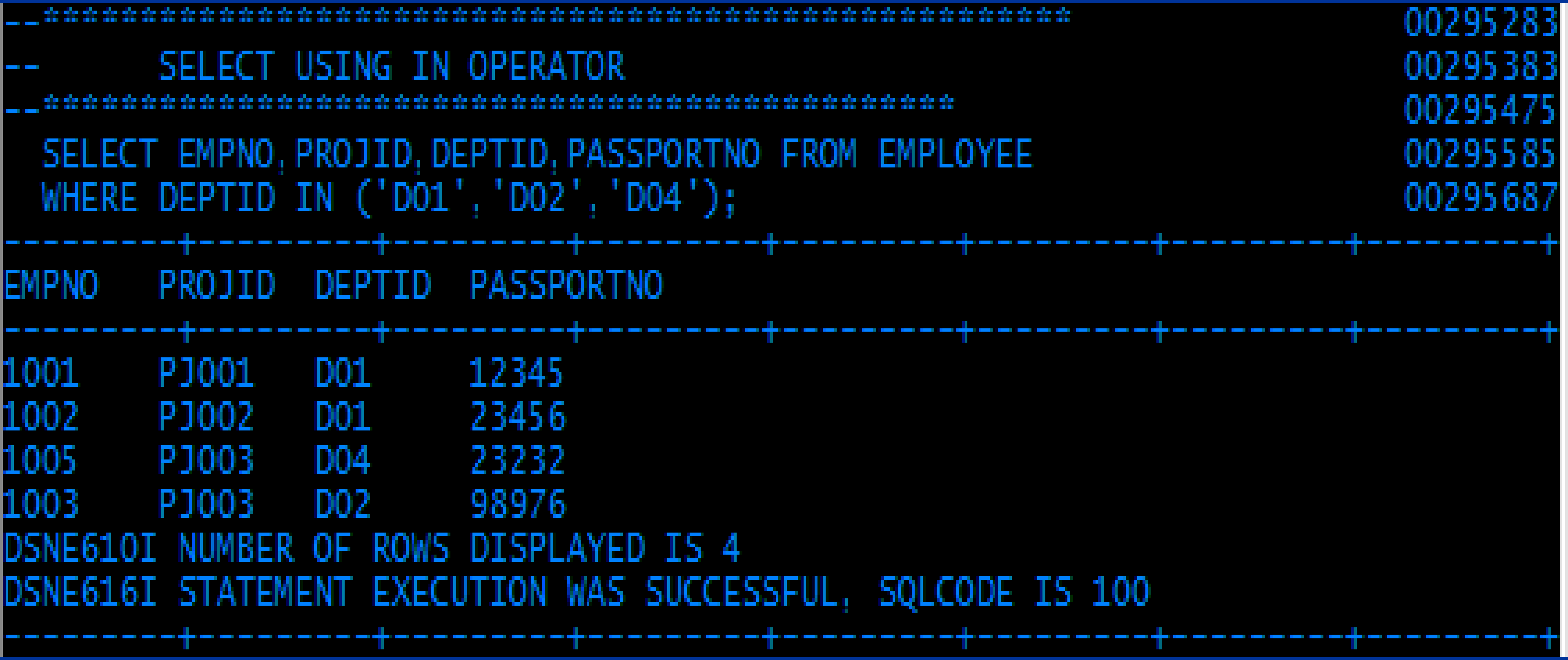
```
 SELECT salary from employee
  Where dept_no = ‘D001’ or
            dept_no =‘D005’,or,
            dept_no =’D003’,or,
            dept_no =’D010’
```

Eg

```
: SELECT salary from employee
  Where dept_no in  ( ‘D001’, ‘D005’,’D003’,’D010’)
```

THE “WHERE” CLAUSE

IN operator.



```
--*****************************************************************        00295283
--       SELECT USING IN OPERATOR                                          00295383
--***********************************************************              00295475
  SELECT EMPNO,PROJID,DEPTID,PASSPORTNO FROM EMPLOYEE                      00295585
  WHERE DEPTID IN ('D01','D02','D04');                                     00295687
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    PROJID  DEPTID  PASSPORTNO
---------+---------+---------+---------+---------+---------+---------+---------+
1001     PJ001   D01     12345
1002     PJ002   D01     23456
1005     PJ003   D04     23232
1003     PJ003   D02     98976
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE “WHERE” CLAUSE

IS operator.

‘IS’ -the operator which is used along where clause to choose the records for operation whose values would be NULL or NOT NULL

update Employee set salary = salary + 5000

where dept_no **is** NULL.

update Employee set salary = salary + 3000

where dept_no **is** NOT NULL.

THE “WHERE” CLAUSE

LIKE operator.

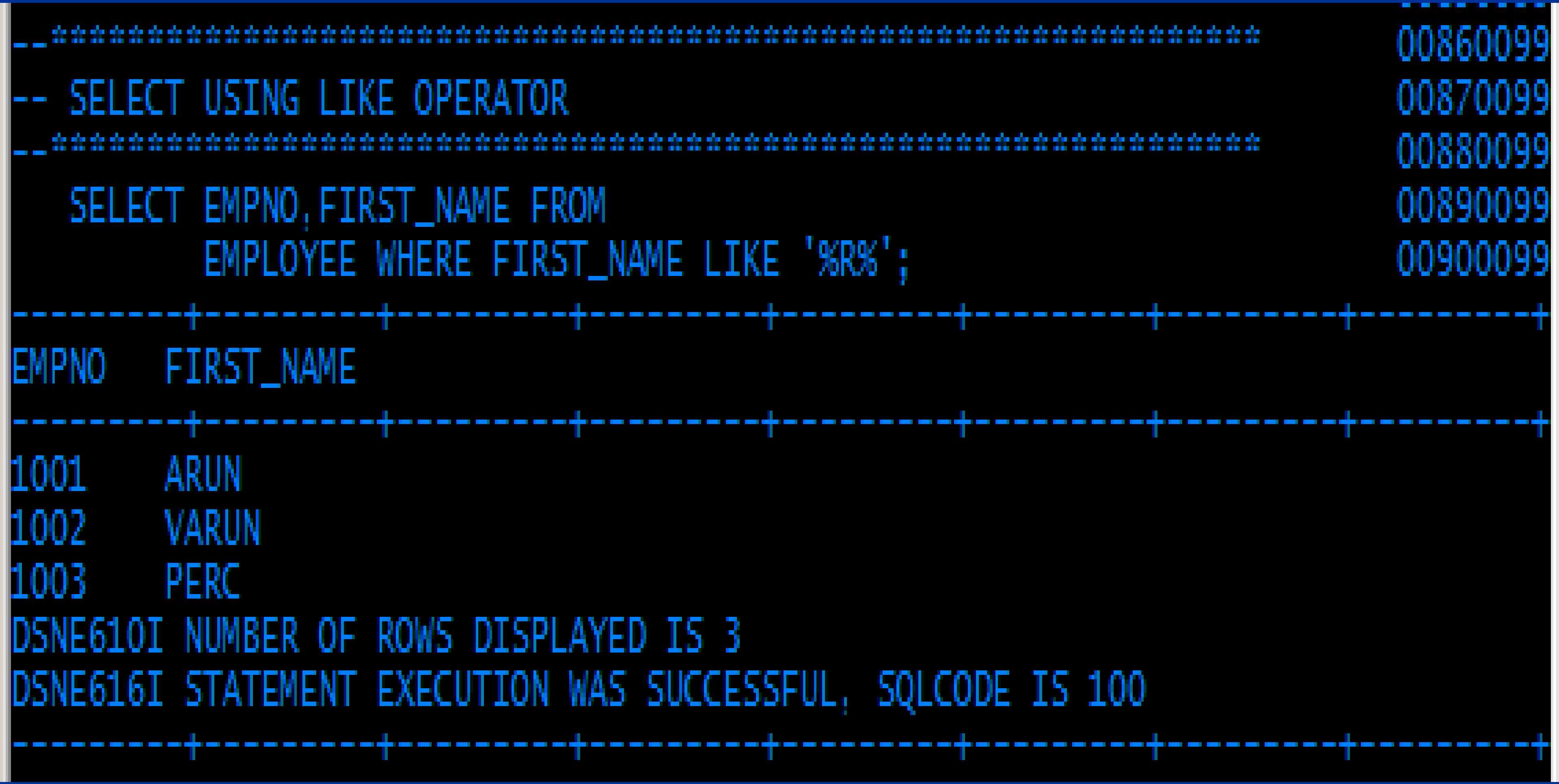
```
E.g. SELECT CUST_NO, CUST_NAME, CUST_ADDR
        FROM CUSTOMER
        WHERE  CUST_ID like/not like ‘425%’
```

WILD CARD charectors

- ‘_’ for a single char ;
- ‘%’ for a string of chars
- ‘\’ - escape char; if precedes ‘_’ or ‘%’ overrides their meaning.

THE “WHERE” CLAUSE

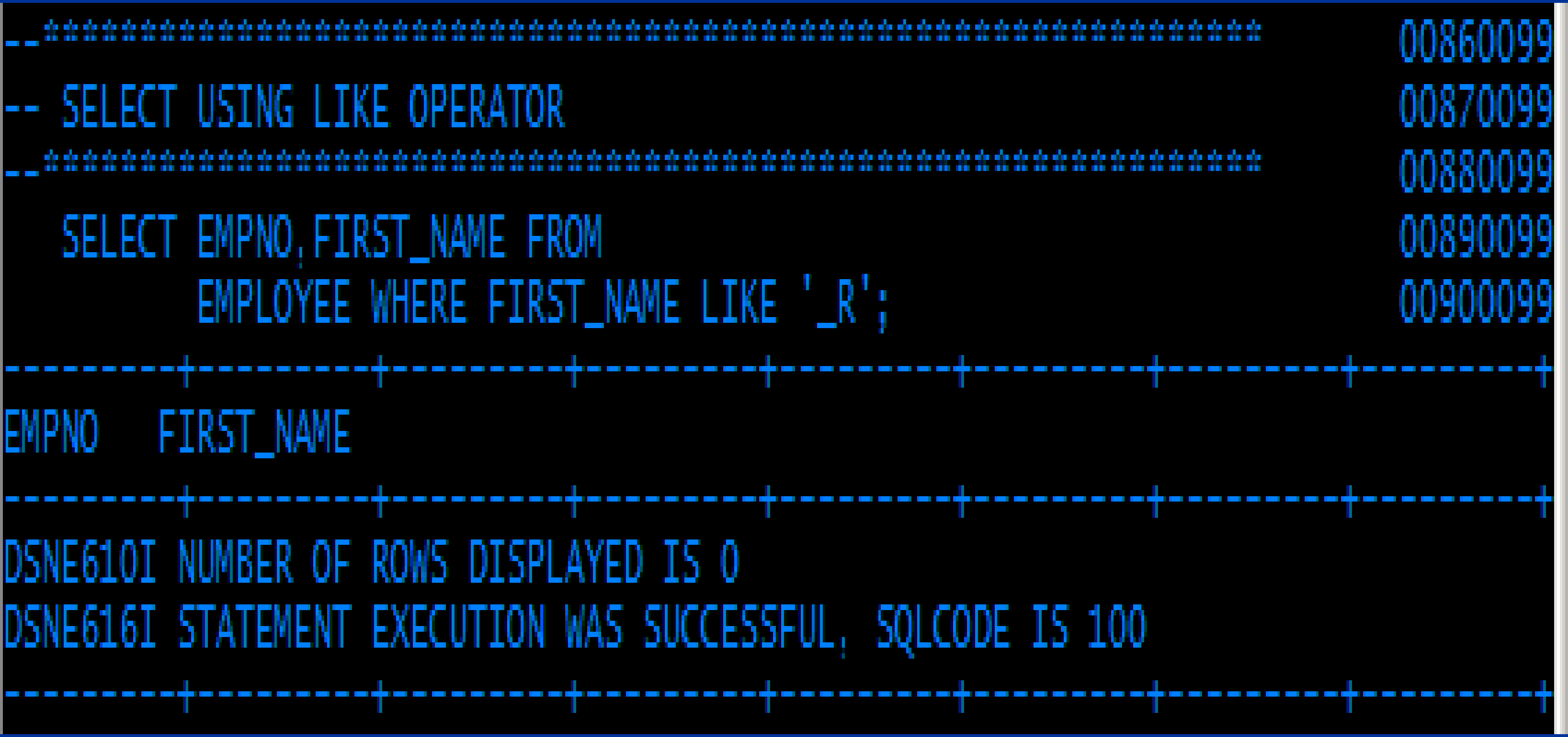
LIKE operator.



```
--****************************************************************          00860099
-- SELECT USING LIKE OPERATOR                                                00870099
--****************************************************************          00880099
   SELECT EMPNO,FIRST_NAME FROM                                              00890099
          EMPLOYEE WHERE FIRST_NAME LIKE '%R%';                              00900099
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
1001     ARUN
1002     VARUN
1003     PERC
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

THE “WHERE” CLAUSE

LIKE operator.



```
--*********************************************************************     00860099
-- SELECT USING LIKE OPERATOR                                               00870099
--*********************************************************************     00880099
   SELECT EMPNO,FIRST_NAME FROM                                             00890099
          EMPLOYEE WHERE FIRST_NAME LIKE '_R';                              00900099
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
DSNE610I NUMBER OF ROWS DISPLAYED IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

4. SELECT - revisited

SELECT is the statement that retrieves the records from the table.

Overall Syntax:

```
SELECT <column_list>  FROM  <table_name>

        {WHERE <search_condition> }  -  optional

        {GROUP BY <grouping cols> }  -  optional

        {HAVING <group search> }        -  optional

        {ORDER BY <sort_order> }        -  optional
```

SELECT – ORDER BY clause

Arranging the Retrieved Data

- Data displayed in SELECT clause is normally not arranged in any sequence.
- If rows need to be sorted, ORDER BY clause needs to be added to it.

```
SELECT column names
      FROM Tablename
      ORDER BY columnname Sequence;
```

SELECT – ORDER BY clause

Arranging the Retrieved Data

```
SELECT EMPNO ,SALARY
FROM EMP
ORDER BY SALARY;
```

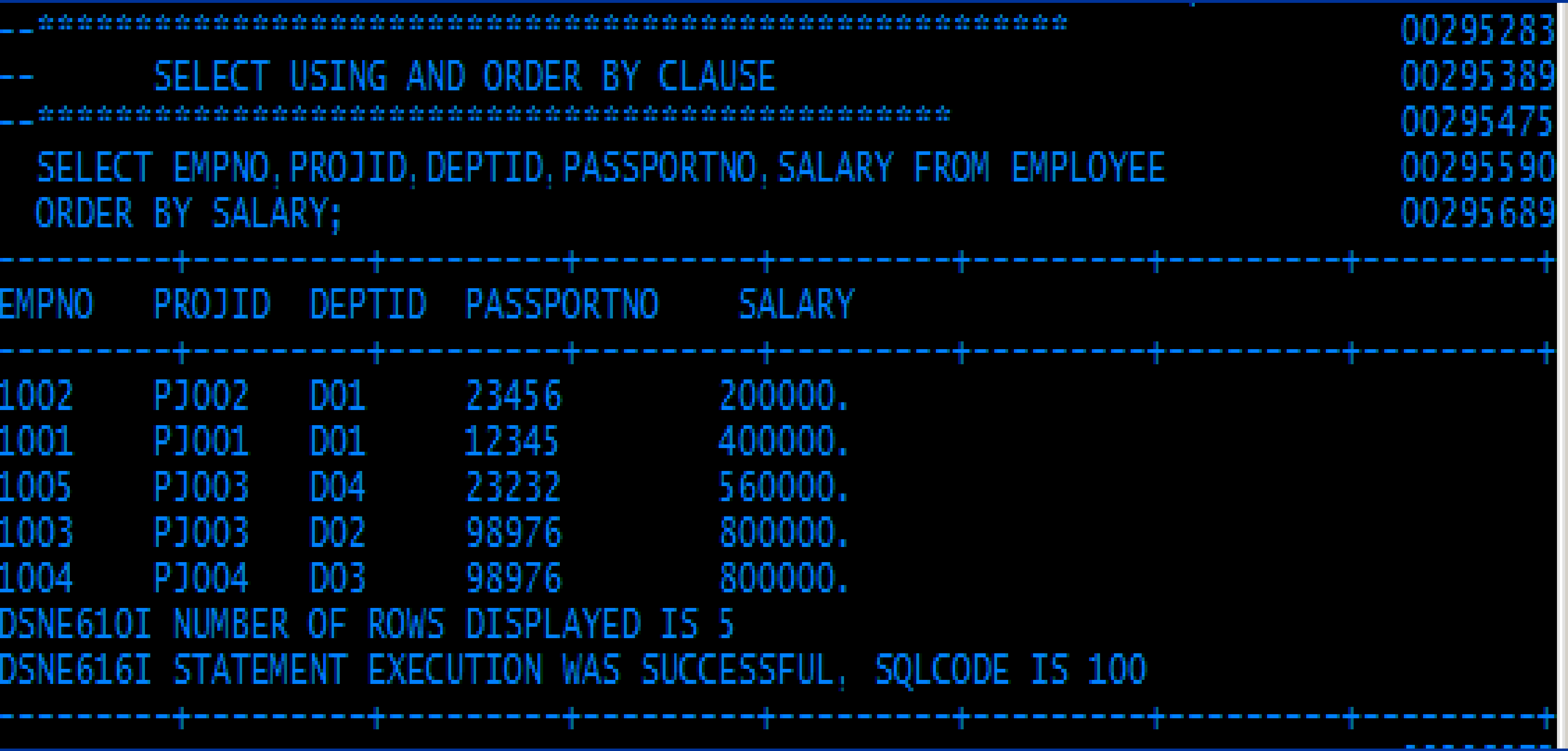
- By default it will display in the ascending order of the salary

```
SELECT EMPNO ,SALARY
FROM EMP
ORDER BY SALARY desc;
```

- Now it will display in the descending order of the salary

SELECT – ORDER BY clause

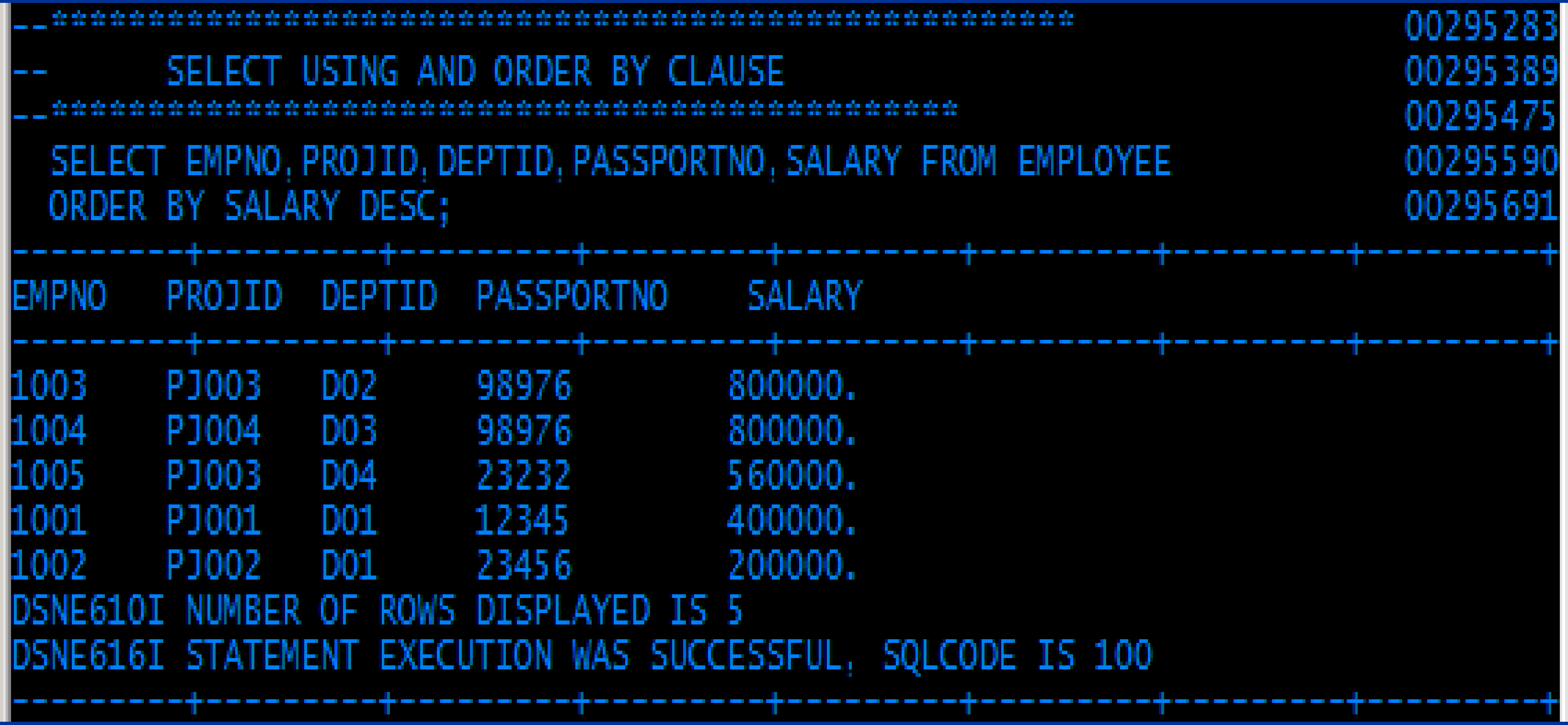
Arranging the Retrieved Data - ASCENDING



```
--*****************************************************************          00295283
--      SELECT USING AND ORDER BY CLAUSE                                        00295389
--**********************************************************                    00295475
  SELECT EMPNO,PROJID,DEPTID,PASSPORTNO,SALARY FROM EMPLOYEE                    00295590
  ORDER BY SALARY;                                                              00295689
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   PROJID  DEPTID  PASSPORTNO    SALARY
---------+---------+---------+---------+---------+---------+---------+---------+
1002    PJ002   D01     23456        200000.
1001    PJ001   D01     12345        400000.
1005    PJ003   D04     23232        560000.
1003    PJ003   D02     98976        800000.
1004    PJ004   D03     98976        800000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

SELECT – ORDER BY clause

Arranging the Retrieved Data - DESCENDING



```
--*****************************************************          00295283
--       SELECT USING AND ORDER BY CLAUSE                        00295389
--***********************************************                00295475
  SELECT EMPNO,PROJID,DEPTID,PASSPORTNO,SALARY FROM EMPLOYEE     00295590
  ORDER BY SALARY DESC;                                          00295691
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   PROJID  DEPTID  PASSPORTNO    SALARY
---------+---------+---------+---------+---------+---------+---------+---------+
1003    PJ003   D02     98976        800000.
1004    PJ004   D03     98976        800000.
1005    PJ003   D04     23232        560000.
1001    PJ001   D01     12345        400000.
1002    PJ002   D01     23456        200000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

SELECT – ORDER BY clause

Arranging the Retrieved Data
based on More than one column

```
SELECT EMPNO,
  SALARY,DEPT_NO
      FROM EMP
      ORDER BY DEPT_NO Desc ,
                       SALARY Asc ;
```

SELECT – GROUP BY clause

- Used to divide the rows in a table in to smaller groups.
- Group functions can then be used to return summary information for each group.

```
SELECT <COLUMNS>
    FROM TABLENAME
    GROUP BY COLUMN;
```

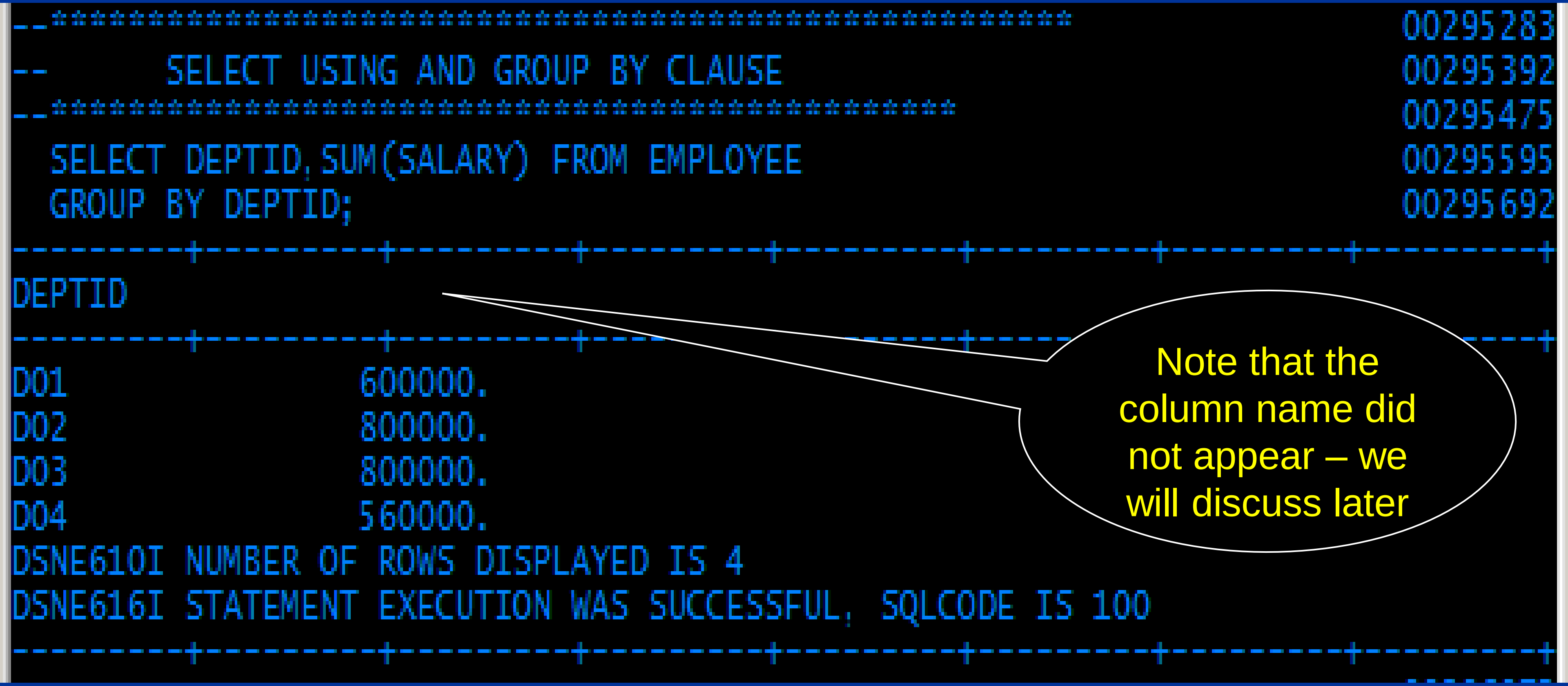
SELECT – GROUP BY clause

- The GROUP BY clause can also be used to get result for groups within groups.
- The following command would display total salary paid for each department.

```
SELECT deptno, SUM(Salary)
    FROM emp
    GROUP BY Deptno;
```

Aggregate functions will be discussed in a short while

SELECT – GROUP BY clause



```
--*************************************************************  00295283
--      SELECT USING AND GROUP BY CLAUSE                          00295392
--*****************************************************           00295475
  SELECT DEPTID,SUM(SALARY) FROM EMPLOYEE                         00295595
  GROUP BY DEPTID;                                                00295692
---------+---------+---------+---------+---------+---------+---------+---------+
DEPTID
---------+---------+---------+---------+---------+---------+---------+---------+
D01                 600000.
D02                 800000.
D03                 800000.
D04                 560000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Note that the column name did not appear – we will discuss later

SELECT – GROUP BY – HAVING clause

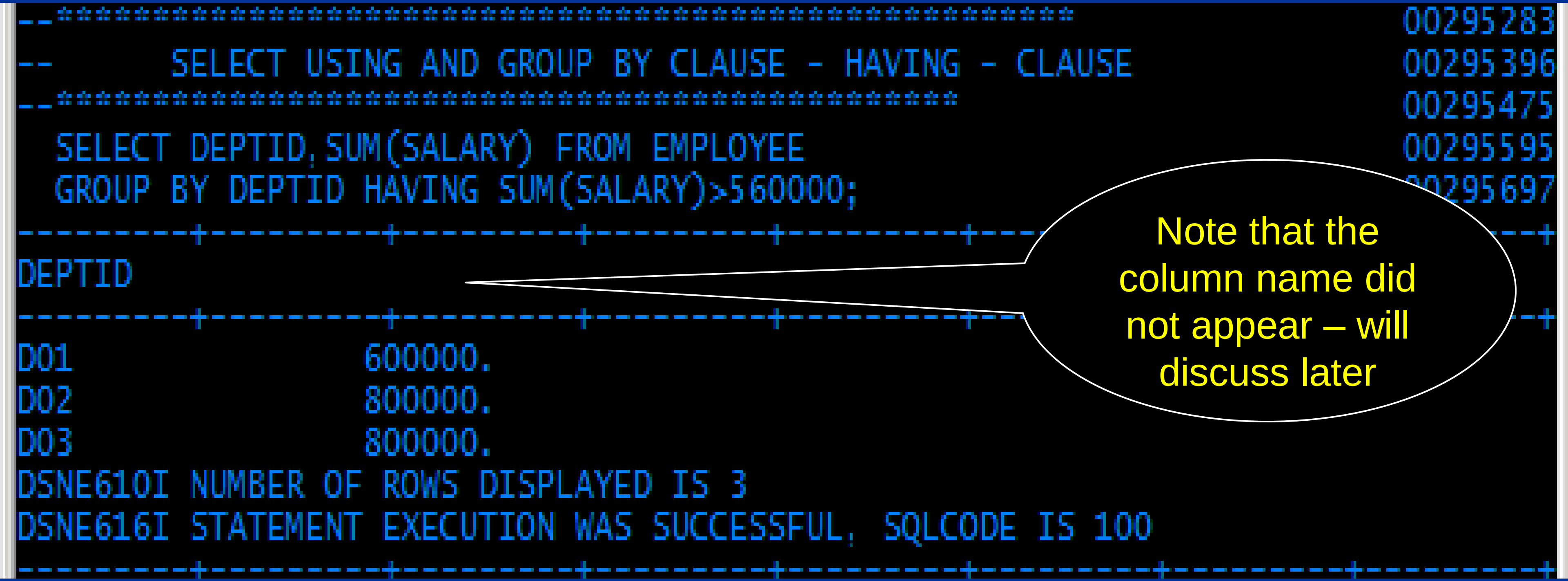
- To Apply the condition on grouped record for retrieval can be done through HAVING Clause.
- **Syntax:**

```
SELECT c1,c2,c3….
       FROM   tablename
       GROUP BY c1
       HAVING condition;
```

Example :

```
SELECT deptno, SUM(Salary)
      FROM emp
      GROUP BY Deptno
Having sum(salary) >100000;
```

SELECT – GROUP BY – HAVING clause



```
--**********************************************************           00295283
--      SELECT USING AND GROUP BY CLAUSE - HAVING - CLAUSE              00295396
--***************************************************                   00295475
  SELECT DEPTID,SUM(SALARY) FROM EMPLOYEE                               00295595
  GROUP BY DEPTID HAVING SUM(SALARY)>560000;                            00295697
---------+---------+---------+---------+---------+----                 ---+
DEPTID
---------+---------+---------+---------+---------+---                  --+
D01                  600000.
D02                  800000.
D03                  800000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 3
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Note that the column name did not appear – will discuss later

SELECT–GROUP BY–HAVING-WHERE clause

- To Apply the condition on grouped record for retrieval can be done through HAVING Clause.

Example :

```
SELECT deptno, SUM(Salary)
        FROM emp
        GROUP BY Deptno
        Having sum(salary) >100000
        where emp_status=‘Permanent’;
```

FETCH CLAUSE

To retrieve set of records by specifying its sequence from a table, FETCH clause can be used.

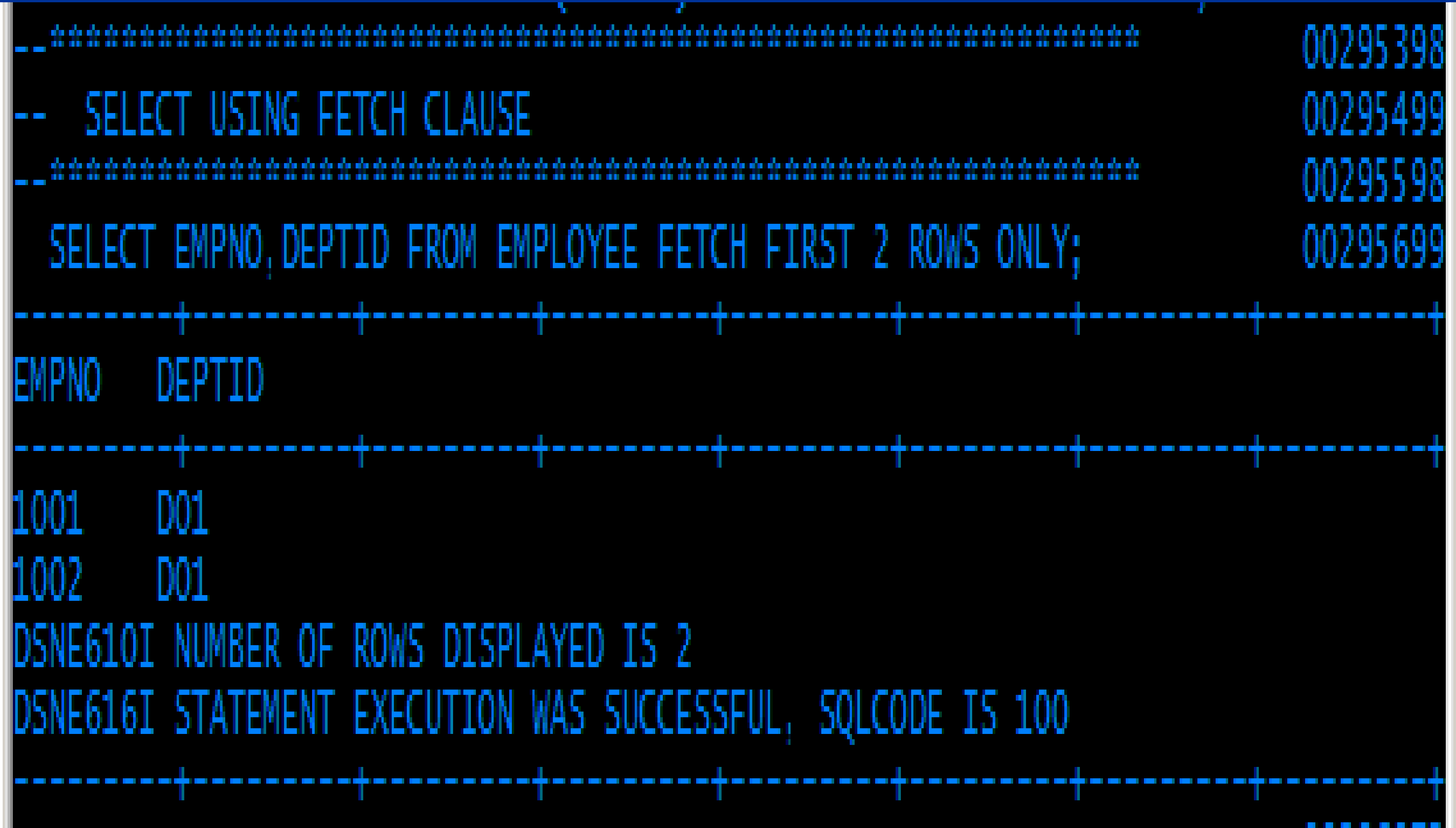
SYNTAX:

```
SELECT * FROM EMP  FETCH FIRST 5 ROWS ONLY.
```

- The above SQL query will retrieve the first five records from the table EMP.

- There is no ”LAST” option.

FETCH CLAUSE



```
--************************************************************          00295398
--  SELECT USING FETCH CLAUSE                                             00295499
--************************************************************          00295598
  SELECT EMPNO,DEPTID FROM EMPLOYEE FETCH FIRST 2 ROWS ONLY;              00295699
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   DEPTID
---------+---------+---------+---------+---------+---------+---------+---------+
1001    D01
1002    D01
DSNE610I NUMBER OF ROWS DISPLAYED IS 2
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

SUBSTITUTION OF NULL VALUES -COALESCE

When a value of a column is NULL the user can assign a value for that particular row and column using COALESCE clause in the SQL statement.

The user assigning value need not be of the same data type as that of the column.

SUBSTITUTION OF NULL VALUES -COALESCE

```
SELECT      DEPTNAME,
            COALESCE (MGRNO, 'UNKNOWN')
            AS MANAGER
FROM        DEPARTMENT
ORDER BY        DEPTNAME;
```

When the value for MGRNO is NULL, “UNKNOWN” will replace it.

DEPTNAME	MANAGER
ADMINISTRATION SYSTEMS	000070
DEVELOPMENT CENTER	UNKNOWN
INFORMATION CENTER	000030
MANUFACTURING SYSTEMS	000060

FUNCTIONS

- Types of Functions are :
  - **Arithmetic Functions**
  - **Aggregate (Column) Function**
  - **Scalar Function**

NOTE: The column name, on which the function is applied, will not appear on the output. But, ALIAS name can be used. It will be discussed in detail later.

ARITHMETIC FUNCTIONS

An arithmetic expression is a combination of one or more values,operators and functions which evaluate to a value.

May contain column names, constant numeric values and the arithmetic operators.

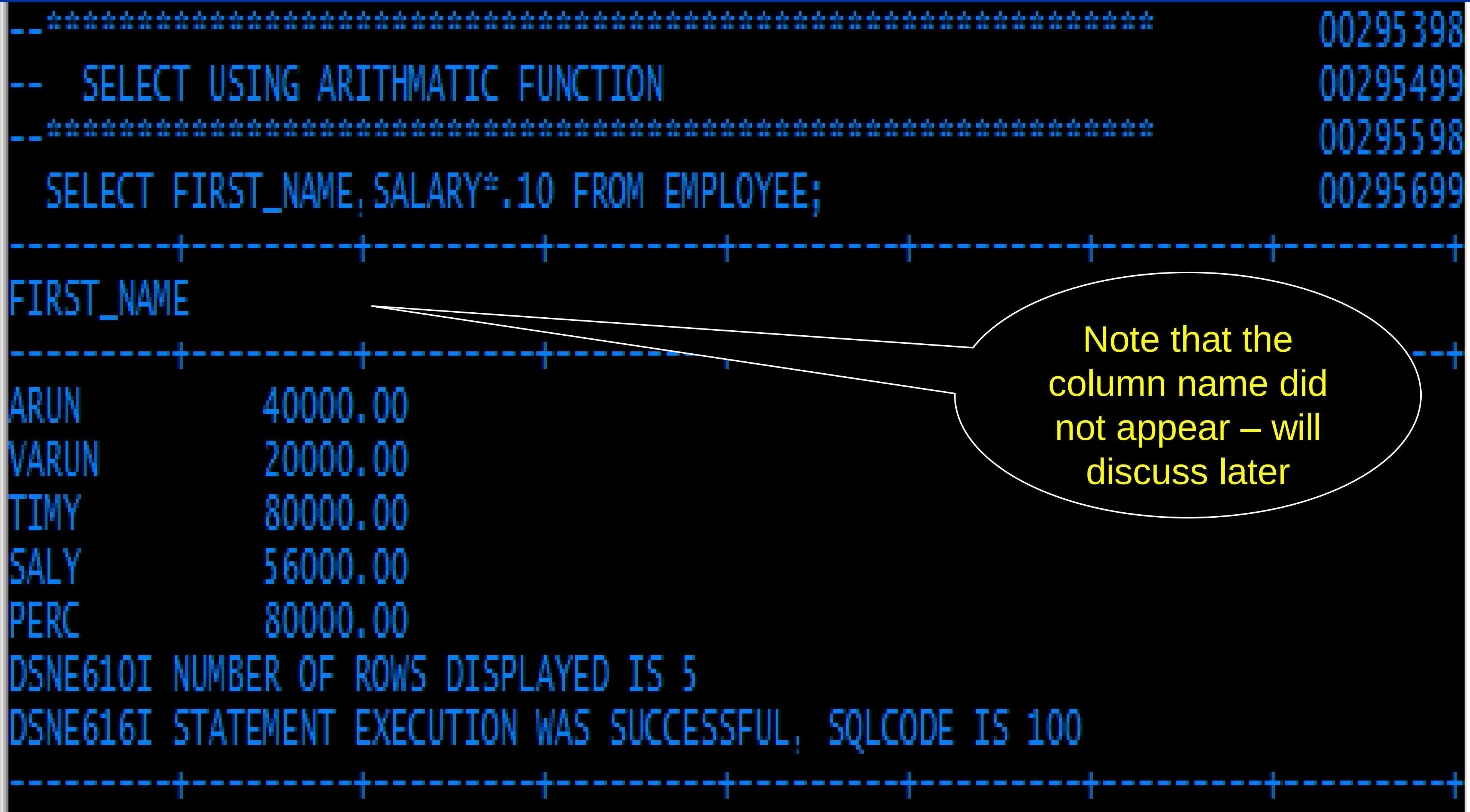
+ add, - subtract, * multiply , / divide

E.G.

```
Select EMPNAME,SALARY*1.1/100
                    From EMP
```

ARITHMETIC FUNCTIONS

MULTIPLICATION



```
--*****************************************************************        00295398
--   SELECT USING ARITHMATIC FUNCTION                                       00295499
--*****************************************************************        00295598
  SELECT FIRST_NAME,SALARY*.10 FROM EMPLOYEE;                               00295699
---------+---------+---------+---------+---------+---------+---------+---------+
FIRST_NAME
---------+---------+---------+---------+---------+---------+---------+---------+
ARUN            40000.00
VARUN           20000.00
TIMY            80000.00
SALY            56000.00
PERC            80000.00
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Note that the column name did not appear – will discuss later

ARITHMETIC WITH NULL VALUES - COALESCE

```
SELECT    EMPNO, SALARY, COMM,
          SALARY + COMM   AS "TOTAL INCOME"
FROM      EMPLOYEE;
```

```
EMPNO      SALARY        COMM          TOTAL INCOME
000210     18270.00      1462.00           19732.00
000260     17250.00            -                  -
000290     15340.00      1227.00           16567.00
000300     17750.00            -                  -
... ... ... ...
```

ARITHMETIC WITH NULL VALUES

```
SELECT EMPNO, SALARY, COMM,
SALARY +COALESCE (COMM, 0)
AS "TOTAL INCOME"
FROM EMPLOYEE
```

EMPNO	SALARY	COMM	TOTAL INCOME
000210	18270.00	1462.00	19732.00
000260	17250.00	-	17250.00
000290	15340.00	1227.00	16567.00
000300	17750.00	-	17750.00

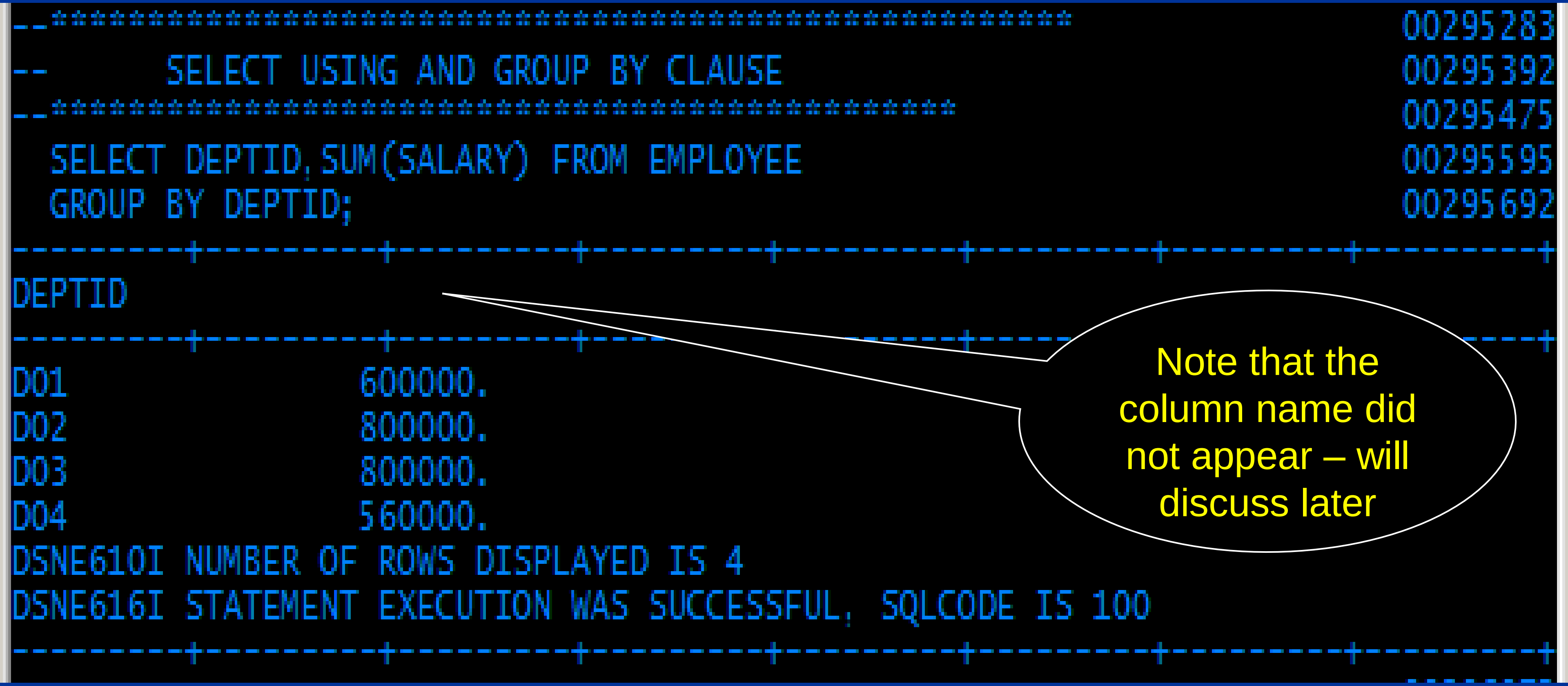
AGGREGATE (COLUMN) FUNCTIONS

- Compute from a group of rows aggregate value for a specified column's
- AVG, COUNT, MAX, MIN, SUM

```
SELECT AVG(SALARY) A00_AVGSAL
FROM DSN0010.EMP
WHERE WORKDEPT = 'A00'
```

Try the others..

AGGREGATE (COLUMN) FUNCTIONS

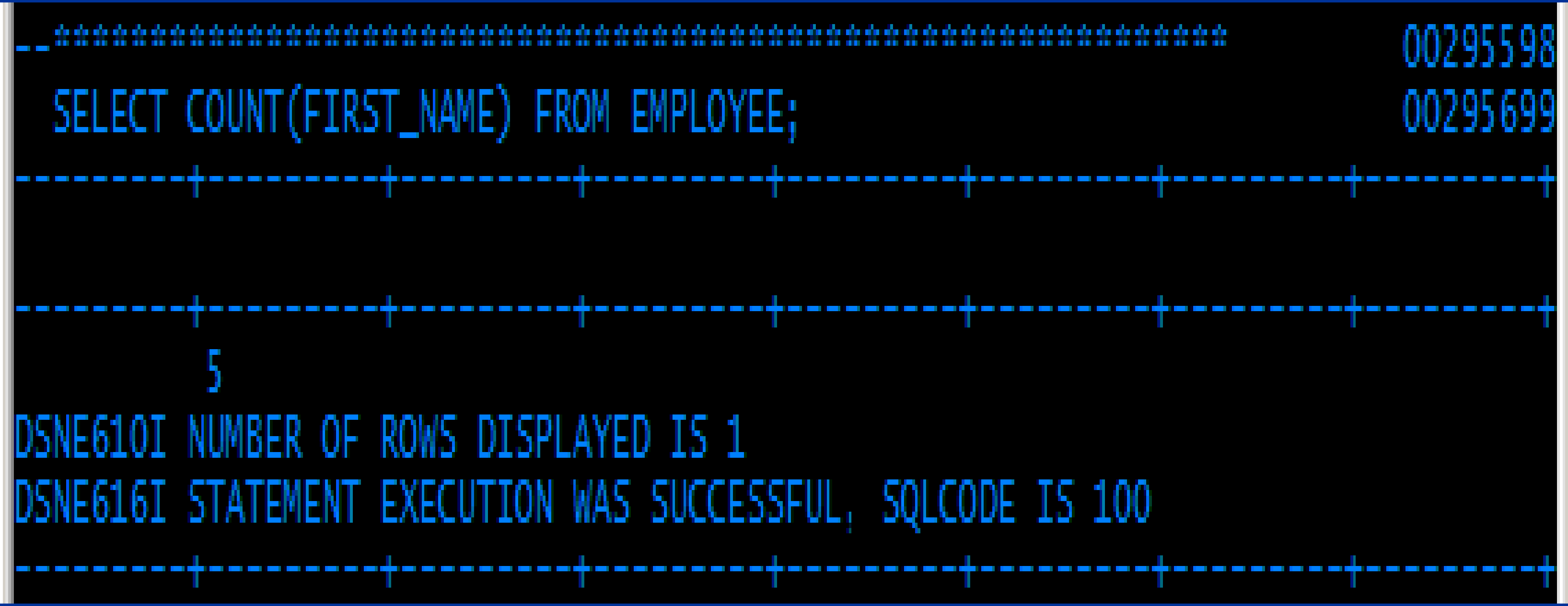


```
--**********************************************************      00295283
--       SELECT USING AND GROUP BY CLAUSE                            00295392
--****************************************************               00295475
  SELECT DEPTID,SUM(SALARY) FROM EMPLOYEE                            00295595
  GROUP BY DEPTID;                                                   00295692
---------+---------+---------+---------+---------+---------+---------+---------+
DEPTID
---------+---------+---------+---------+---------+---------+---------+---------+
D01                 600000.
D02                 800000.
D03                 800000.
D04                 560000.
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

Note that the column name did not appear – will discuss later

AGGREGATE (COLUMN) FUNCTIONS

COUNT



```
--*****************************************************************          00295598
  SELECT COUNT(FIRST_NAME) FROM EMPLOYEE;                                     00295699
---------+---------+---------+---------+---------+---------+---------+---------+


---------+---------+---------+---------+---------+---------+---------+---------+
          5
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

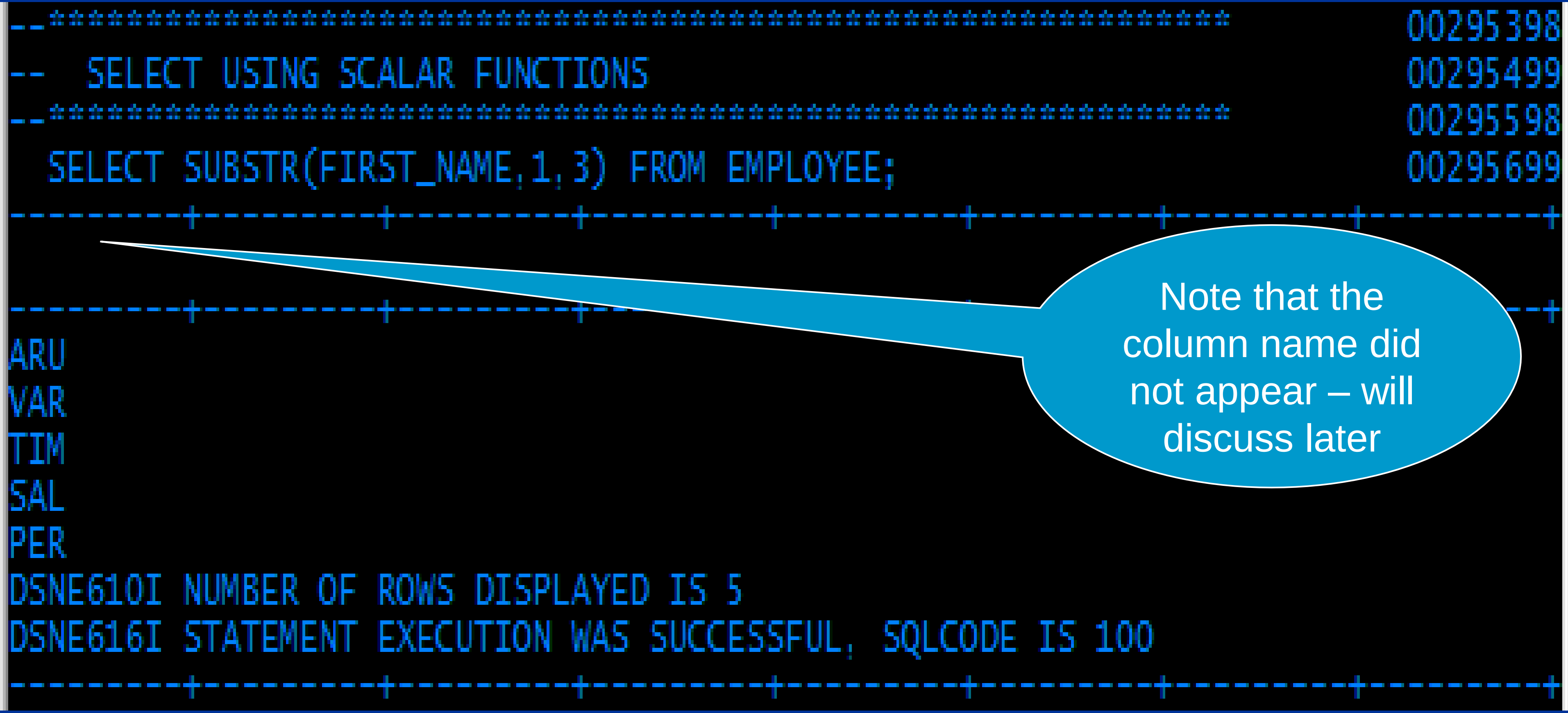
SCALAR FUNCTIONS

- Are applied to a column or expression and operate on a single value.
- CHAR, DATE, DAY(S), DECIMAL, DIGITS, FLOAT, HEX, HOUR, INTEGER, LENGTH, MICROSECOND, MINUTE, MONTH, SECOND, SUBSTR, TIME, TIMESTAMP, VALUE, YEAR

```
SELECT SUBSTR(CUST_NAME,1,7)
FROM DSN0010.CUSTOMER
```

SCALAR FUNCTIONS - SUBSTR

SUBSTR

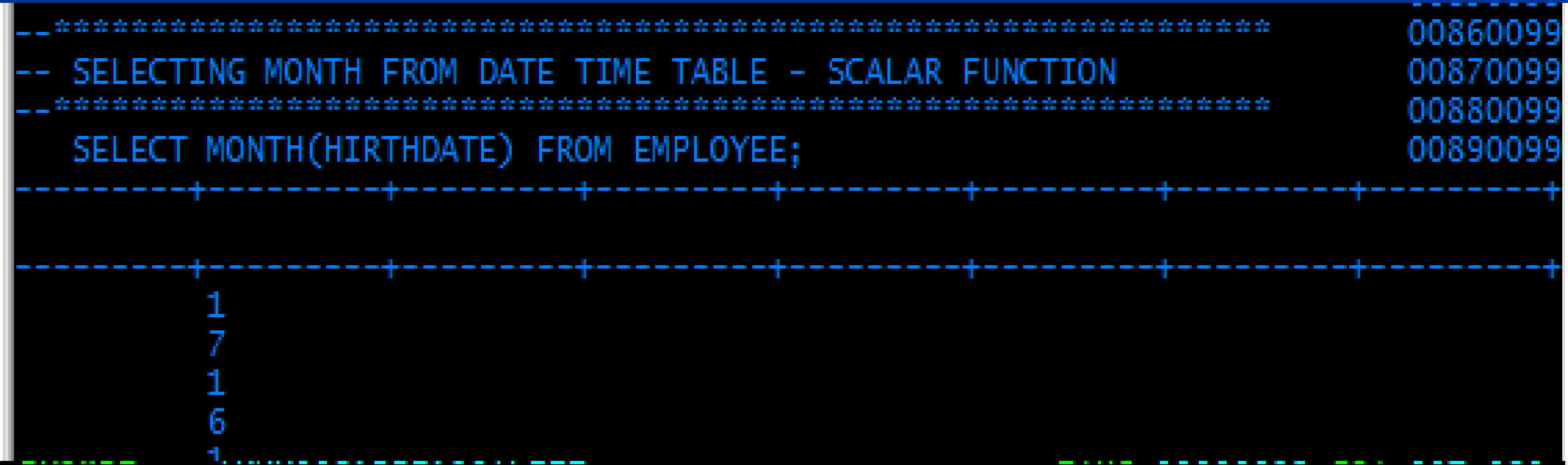


```
--*****************************************************************      00295398
--   SELECT USING SCALAR FUNCTIONS                                          00295499
--*****************************************************************      00295598
  SELECT SUBSTR(FIRST_NAME,1,3) FROM EMPLOYEE;                              00295699
---------+---------+---------+---------+---------+---------+---------+---------+

---------+---------+---------+---------+---------+---------+---------+---------+
ARU
VAR
TIM
SAL
PER
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

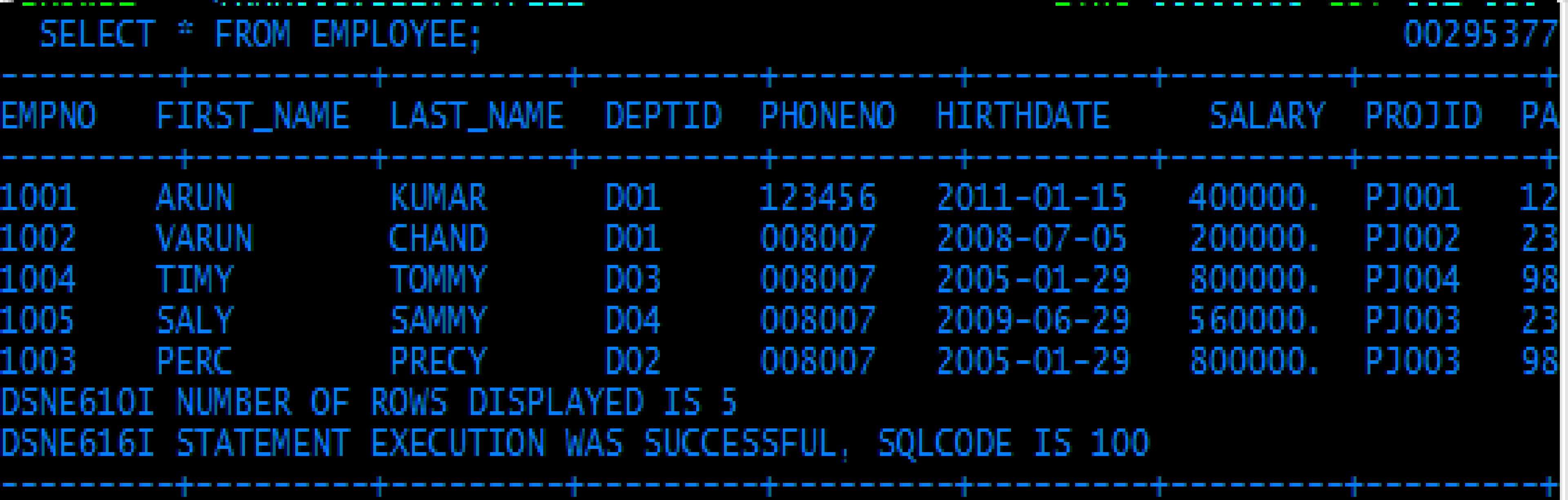
Note that the column name did not appear – will discuss later

SCALAR FUNCTIONS - MONTH



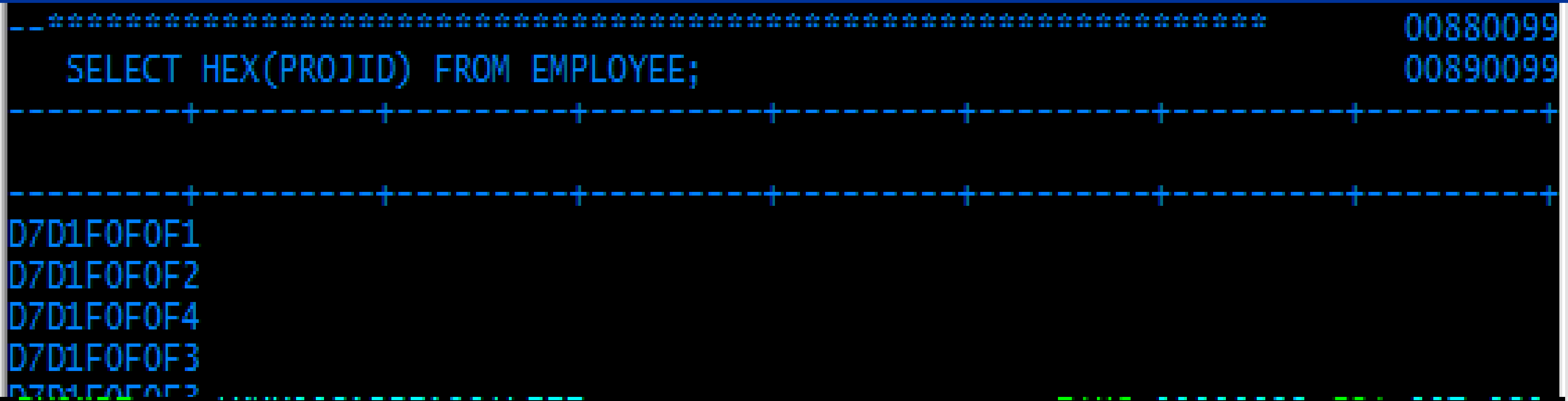
```
--*********************************************************************  00860099
-- SELECTING MONTH FROM DATE TIME TABLE - SCALAR FUNCTION                 00870099
--*********************************************************************  00880099
   SELECT MONTH(HIRTHDATE) FROM EMPLOYEE;                                 00890099
---------+---------+---------+---------+---------+---------+---------+---------+


---------+---------+---------+---------+---------+---------+---------+---------+
          1
          7
          1
          6
```



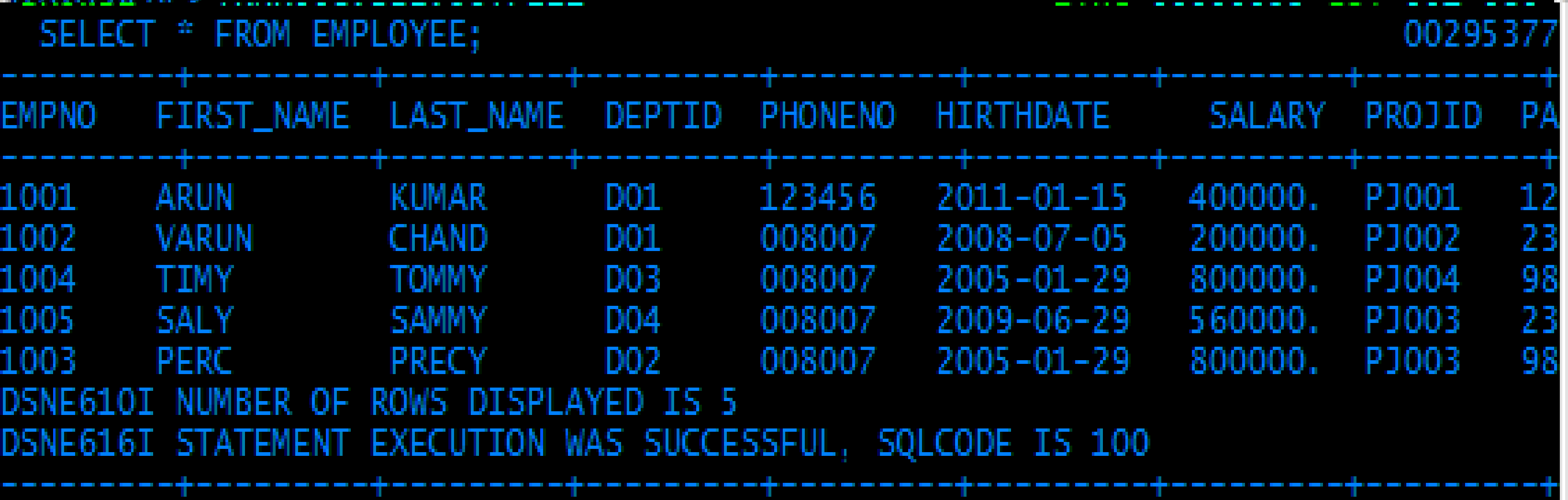
```
  SELECT * FROM EMPLOYEE;                                                 00295377
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO    FIRST_NAME  LAST_NAME  DEPTID  PHONENO  HIRTHDATE    SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001     ARUN        KUMAR      D01     123456   2011-01-15  400000.  PJ001   12
1002     VARUN       CHAND      D01     008007   2008-07-05  200000.  PJ002   23
1004     TIMY        TOMMY      D03     008007   2005-01-29  800000.  PJ004   98
1005     SALY        SAMMY      D04     008007   2009-06-29  560000.  PJ003   23
1003     PERC        PRECY      D02     008007   2005-01-29  800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

SCALAR FUNCTIONS - HEX



```
--****************************************************************      00880099
   SELECT HEX(PROJID) FROM EMPLOYEE;                                        00890099
---------+---------+---------+---------+---------+---------+---------+---------+


---------+---------+---------+---------+---------+---------+---------+---------+
D7D1F0F0F1
D7D1F0F0F2
D7D1F0F0F4
D7D1F0F0F3
```



```
  SELECT * FROM EMPLOYEE;                                                 00295377
---------+---------+---------+---------+---------+---------+---------+---------+
EMPNO   FIRST_NAME  LAST_NAME  DEPTID  PHONENO  HIRTHDATE     SALARY  PROJID  PA
---------+---------+---------+---------+---------+---------+---------+---------+
1001    ARUN        KUMAR      D01     123456   2011-01-15   400000.  PJ001   12
1002    VARUN       CHAND      D01     008007   2008-07-05   200000.  PJ002   23
1004    TIMY        TOMMY      D03     008007   2005-01-29   800000.  PJ004   98
1005    SALY        SAMMY      D04     008007   2009-06-29   560000.  PJ003   23
1003    PERC        PRECY      D02     008007   2005-01-29   800000.  PJ003   98
DSNE610I NUMBER OF ROWS DISPLAYED IS 5
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```

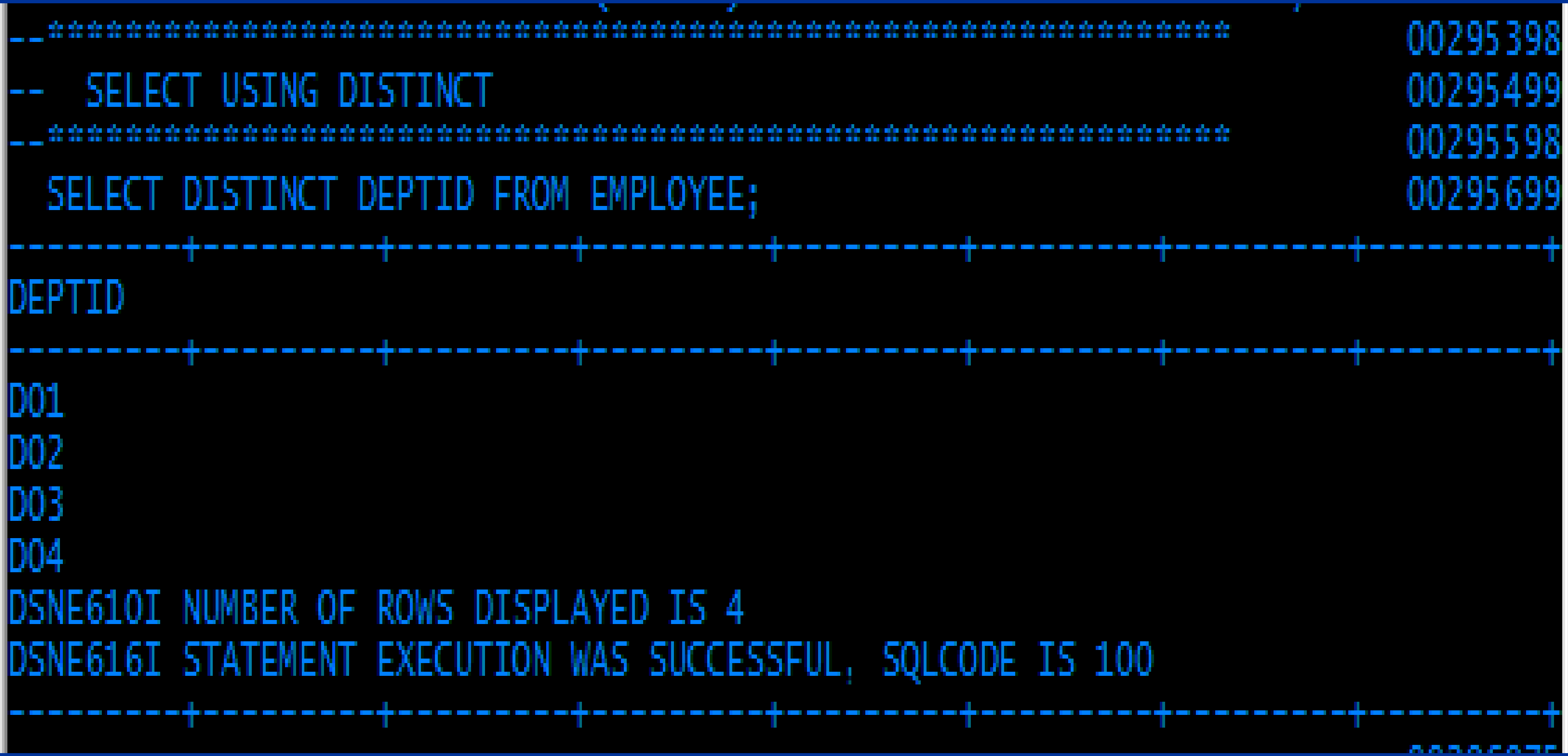
ELIMINATION OF DUPLICATION

- To avoid duplication of entries in a SELECT output, use the DISTINCT clause.

Example :

```
SELECT DISTINCT DEPTNO
                                    FROM EMP;
```

ELIMINATION OF DUPLICATION



```
--*****************************************************************          00295398
--  SELECT USING DISTINCT                                                    00295499
--*****************************************************************          00295598
  SELECT DISTINCT DEPTID FROM EMPLOYEE;                                      00295699
---------+---------+---------+---------+---------+---------+---------+---------+
DEPTID
---------+---------+---------+---------+---------+---------+---------+---------+
D01
D02
D03
D04
DSNE610I NUMBER OF ROWS DISPLAYED IS 4
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 100
---------+---------+---------+---------+---------+---------+---------+---------+
```