1	Give a listing of all the ssns,first names and the class descriptions of all the classes the students are taking. If there are no class _descriptions display 'No description is available yet'. (USE NVL)

```
194 SELECT s.ssn, s.fname, NVL(c.class_description, 'No description is available yet')
195 FROM Hert_student s
196 JOIN Hert_student_class sc ON s.ssn = sc.ssn
197 LEFT JOIN Hert_class c ON sc.class_code = c.class_code;
```

Script Output x | Query Result x

SQL | All Rows Fetched: 14 in 0.043 seconds

	SSN	FNAME	NVL(C.CLASS_DESCRIPTION,'NODESCRIPTIONISAVAILABLEYET')
1	172-32-1176	Johnson	Database Programming
2	213-46-8915	Marjorie	Introduction to C programming
3	267-41-2394	Michael	Intro to principles
4	409-56-7008	Abraham	Database Programming
5	427-17-2319	Ann	Intro to principles
6	472-27-2349	Burt	Introduction to C programming
7	486-29-1786	Chastity	No description is available yet
8	527-72-3246	Morningstar	No description is available yet
9	648-92-1872	Reginald	No description is available yet
10	672-71-3249	Akiko	Introduction to Computers
11	712-45-1867	Innes	Database Programming
12	846-92-7186	Sheryl	Introduction to C programming
13	998-72-3567	Albert	Introduction to Computers
14	998-72-3567	Albert	No description is available yet

2	Give a listing of only the lname and the class_code for students who are taking 'Introduction to C programming'. (Inner join)

```
207 select s.lname, c.class_code
208 from Hert_student_class sc
209 inner join Hert_student s
210 on sc.ssn = s.ssn
211 inner join Hert_class c
212 on sc.class_code = c.class_code
213 where c.class_description = 'Introduction to C programming';
```

Script Output | Query Result

SQL | All Rows Fetched: 3 in 0.04 seconds

	LNAME	CLASS_CODE
1	Green	32
2	Gringlesby	32
3	Hunter	32

3

Give a lising of all the class_descriptions and the number of students enrolled in each class for all students who are older than the average age where the total number of students for the class is more than 1 student. Order by the number of students. If there is no class description replace it with 'Other Classes'
(Note: Take it in steps. First do all those who are older than the average age, then do the group by, then add the having clause and then the order and then combine everything together)

```
218 select NVL(c.class_description, 'Other Classes') "class_description", COUNT(s.ssn) "number_of_students"
219 from Hert_student s
220 inner join Hert_student_class sc
221 on s.ssn = sc.ssn
222 inner join Hert_class c
223 on sc.class_code = c.class_code
224 where (SYSDATE - s.dob)/365.25 >
225 (select avg((SYSDATE - dob)/365.25)
226 from Hert_student)
227 group by c.class_code, c.class_description
228 having COUNT(s.ssn) > 1
229 order by COUNT(s.ssn);
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.04 seconds

	class_description	number_of_students
1	Introduction to C programming	2
2	Database Programming	2

4

Give a listing of all the classes for which no students are enrolled in (use in or not in clause) (subquery)

```
221 SELECT class_code, NVL(class_description, 'No description is available yet') AS class_description
222 FROM Hert_class
223 WHERE class_code NOT IN (SELECT class_code FROM Hert_student_class);
```

Script Output | Query Result

SQL | All Rows Fetched: 1 in 0.043 seconds

	CLASS_CODE	CLASS_DESCRIPTION
1	14A	Operating systems

5	Give a listing of all the students who are not enrolled in any classes (Note: Use Exists or not Exists)

```
237  SELECT * from Hert_student s where not exists
238  (select ssn from Hert_student_class where ssn = s.ssn);
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.056 seconds

	SSN	LNAME	FNAME	PHONE	ADDRESS
1	238-95-7766	Gren	Cheryl	415 548-7723	589 Darwin Ln.
2	999-00-0000	Al	Cal	615 297-2723	22 Graybar House Rd.

6	create a new table that contains the list of all the students and class descriptions. Include In this table the list of all students who are not enrolled in any classes (display no classes). If there are no class descriptions then display 'no description' (Use combination of inner join, union and minus) (Note: minus will deal with the students who are not enrolled in any classes)

```
241  CREATE TABLE new_table AS
242
243  SELECT fname, lname, NVL(class_description, 'No Description') AS "Class Description"
244  FROM Hert_STUDENT st inner join Hert_STUDENT_CLASS sc
245  on st.ssn = sc.ssn inner join Hert_CLASS cl on sc.class_code = cl.class_code
246  UNION
247  ((SELECT fname, lname, NVL(class_description, 'No Description') AS "Class Description"
248  FROM Hert_STUDENT st inner join Hert_STUDENT_CLASS sc
249  on st.ssn = sc.ssn inner join Hert_CLASS cl on sc.class_code = cl.class_code)
250  MINUS
251  (SELECT fname, lname, NVL(class_description, 'No Description') AS "Class Description"
252  FROM Hert_STUDENT st inner join Hert_STUDENT_CLASS sc
253  on st.ssn = sc.ssn inner join Hert_CLASS cl on sc.class_code = cl.class_code));
```

Script Output | Query Result

SQL | All Rows Fetched: 14 in 0.046 seconds

	FNAME	LNAME	Class Description
1	Abraham	Bennet	Database Programming
2	Akiko	Yokomoto	Introduction to Computers
3	Albert	Greeenr	Introduction to Computers
4	Albert	Greeenr	No Description
5	Ann	Dull	Intro to principles
6	Burt	Gringlesby	Introduction to C programming
7	Chastity	Locksley	No Description
8	Innes	del Castillo	Database Programming
9	Johnson	White	Database Programming
10	Marjorie	Green	Introduction to C programming
11	Michael	O'Leary	Intro to principles
12	Morningstar	Greene	No Description
13	Reginald	Blotchet-Halls	No Description
14	Sheryl	Hunter	Introduction to C programming

7	repeat question 6 using a combination of inner join, union and not exists (Note: Not exists will deal with the students who are not enrolled in any classes)

```
259  SELECT fname, lname, NVL(class_description, 'No Description') AS "Class Description"
260  FROM Hert_STUDENT s INNER JOIN Hert_student_class sc
261  ON s.ssn = sc.ssn
262  INNER JOIN class c ON sc.class_code = c.class_code
263  UNION
264  (
265  SELECT fname, lname, 'no classes' FROM student s
266  WHERE NOT EXISTS (SELECT * FROM Hert_student_Class sc
267  WHERE s.ssn = sc.ssn)
268  )
```

Script Output x | Query Result x | Query Result 1 x

SQL | All Rows Fetched: 2 in 0.036 seconds

	FNAME	LNAME	Class Description
1	Cal	Al	no classes
2	Cheryl	Gren	no classes

8	**create a view.** We want to find out which courses are being taken by the different students for all those whose age is greater than the average age. Give a listing of the course descriptions and student names (Inner join)

```
272  CREATE OR REPLACE VIEW average_age_view AS
273  SELECT AVG(EXTRACT(YEAR FROM sysdate) - EXTRACT(YEAR FROM dob)) AS avg_age
274  FROM Hert_student;
275
276  CREATE OR REPLACE VIEW course_student_view AS
277  SELECT c.class_description, s.fname || ' ' || s.lname AS student_name
278  FROM Hert_student s
279  JOIN Hert_student_class sc ON s.ssn = sc.ssn
280  JOIN class c ON sc.class_code = c.class_code
281  JOIN average_age_view avg_age ON EXTRACT(YEAR FROM sysdate) - EXTRACT(YEAR FROM s.dob) > avg_age.avg_age;
282
283  SELECT * FROM course_student_view;
```

Script Output x | Query Result x

Task completed in 0.218 seconds

```
view AVERAGE_AGE_VIEW created.
view COURSE_STUDENT_VIEW created.
>>Query Run In:Query Result
```

9	We want to find out the courses that each student is not enrolled in. Give a listing of the course descriptions, and the students (lname) who are not taking that specific course (Use a cartesian product and union it with a minus)

```
289 SELECT lname, class_description FROM Hert_STUDENT, Hert_CLASS, Hert_STUDENT_CLASS
290 UNION
291 (SELECT lname, class_description
292 FROM Hert_STUDENT st, Hert_CLASS cl, Hert_STUDENT_CLASS sc
293 WHERE st.ssn = sc.ssn (+) AND sc.class_code = cl.class_code (+))
294 MINUS
295 (SELECT lname, class_description
296 FROM Hert_STUDENT st, Hert_CLASS cl, Hert_STUDENT_CLASS sc
297 WHERE st.ssn = sc.ssn AND sc.class_code = cl.class_code);
```

55	Locksley	Introduction to Computers
59	O'Leary	Introduction to Computers
64	White	Introduction to Computers
74	del Castillo	Introduction to Computers

10 Use the system catalog tables to display the results to find out the following:(Note show me the SQL syntax along with your results) Only a single SQL statement for each question.

a) Primary key name and the columns that make up the primary key for **student** table

```
300 SELECT cols.table_name, cols.column_name
301 FROM all_constraints cons, all_cons_columns cols
302 WHERE cols.table_name = 'STUDENT'
303 AND cons.constraint_type = 'P'
304 AND cons.constraint_name = cols.constraint_name
305 AND cons.owner = cols.owner;
```

Script Output x | Query Result x

SQL | All Rows Fetched: 1 in 0.188 seconds

	TABLE_NAME	COLUMN_NAME
1	STUDENT	SSN

b) Unique key name and the columns that make up the unique key for the **student** table

```
308 SELECT cols.table_name, cols.column_name
309 FROM all_constraints cons, all_cons_columns cols
310 WHERE cols.table_name = 'STUDENT'
311 AND cons.constraint_type = 'U'
312 AND cons.constraint_name = cols.constraint_name
313 AND cons.owner = cols.owner;
```

Script Output | Query Result

SQL | All Rows Fetched: 2 in 0.201 seconds

	TABLE_NAME	COLUMN_NAME
1	STUDENT	LNAME
2	STUDENT	FNAME

c) Foreign key name, the columns that make up the foreign key and the columns it references in the parent table for **student_class** table

```
316 select * from all_constraints where constraint_name in (select r_constraint_name from
317 all_constraints where table_name in 'STUDENT');
```

OWNER	CONSTRA...	CONSTRA...	TABLE_N...	SEARCH_CO...	SEARCH_...	R_OWNER

d) Name of all the check constraints and their conditions for the **student** table

```
327 SELECT constraint_name, search_condition
328 FROM user_constraints
329 WHERE table_name = 'STUDENT'
330 AND constraint_type = 'C';
```