EXPERIMENT 1
DATE :

1. **Create** the following Relation (Tables) with primary key integrity constraint
-- create
CREATE TABLE instructor (
 ID INTEGER PRIMARY KEY,
 name TEXT NOT NULL,
 dept_name TEXT NOT NULL,
 salary INTEGER NOT NULL
);

-- insert
INSERT INTO instructor (ID, name, dept_name, salary) VALUES
(10101, 'Srinivasan', 'Comp. Sci.', 65000),
(12121, 'Wu', 'Finance', 90000),
(15151, 'Mozart', 'Music', 40000),
(22222, 'Einstein', 'Physics', 95000),
(32343, 'El Said', 'History', 60000),
(33456, 'Gold', 'Physics', 87000),
(45565, 'Katz', 'Comp. Sci.', 75000),
(58583, 'Califieri', 'History', 6200),
(76543, 'Singh', 'Finance', 80000),
(76766, 'Crick', 'Biology', 72000),
(83821, 'Brandt', 'Comp. Sci.', 92000),
(98345, 'Kim', 'Elec. Eng', 80000);
-- fetch
SELECT * FROM instructor;

```
+-------+-----------+------------+--------+
| ID    | name      | dept_name  | salary |
+-------+-----------+------------+--------+
| 10101 | Srinivasan| Comp. Sci. |  65000 |
| 12121 | Wu        | Finance    |  90000 |
| 15151 | Mozart    | Music      |  40000 |
| 22222 | Einstein  | Physics    |  95000 |
| 32343 | El Said   | History    |  60000 |
| 33456 | Gold      | Physics    |  87000 |
| 45565 | Katz      | Comp. Sci. |  75000 |
| 58583 | Califieri | History    |   6200 |
| 76543 | Singh     | Finance    |  80000 |
| 76766 | Crick     | Biology    |  72000 |
| 83821 | Brandt    | Comp. Sci. |  92000 |
| 98345 | Kim       | Elec. Eng  |  80000 |
+-------+-----------+------------+--------+
```

2. **Create** the following Relation (Tables)
teaches CREATE TABLE teaches (

```
  ID int NOT NULL,
  course_id varchar(255) NOT NULL,
  sec_id int NOT NULL,
  semester varchar(255) NOT NULL,
  year int NOT NULL,
  FOREIGN KEY (ID) REFERENCES instructor(ID)
);

INSERT INTO teaches (ID, course_id, sec_id, semester, year) VALUES
(10101, 'CS-101', 1, 'Fall', 2017),
(10101, 'CS-315', 1, 'Spring', 2018),
(10101, 'CS-347', 1, 'Fall', 2017),
(12121, 'FIN-201', 1, 'Spring', 2018),
(15151, 'MU-199', 1, 'Spring', 2015),
(22222, 'PHY-101', 1, 'Fall', 2017),
(32343, 'HIS-351', 1, 'Spring', 2018),
(45565, 'CS-101', 1, 'Spring', 2018),
(45565, 'CS-319', 1, 'Spring', 2018),
(76766, 'BIO-101', 1, 'Summer', 2017),
(76766, 'BIO-301', 1, 'Summer', 2018),
(83821, 'CS-190', 1, 'Spring', 2017),
(83821, 'CS-190', 2, 'Spring', 2017),
(83821, 'CS-319', 2, 'Spring', 2018),
(98345, 'EE-181', 1, 'Spring', 2017);

SELECT * FROM teaches;
```

```
+-------+-----------+--------+----------+------+
| ID    | course_id | sec_id | semester | year |
+-------+-----------+--------+----------+------+
| 10101 | CS-101    |      1 | Fall     | 2017 |
| 10101 | CS-315    |      1 | Spring   | 2018 |
| 10101 | CS-347    |      1 | Fall     | 2017 |
| 12121 | FIN-201   |      1 | Spring   | 2018 |
| 15151 | MU-199    |      1 | Spring   | 2015 |
| 22222 | PHY-101   |      1 | Fall     | 2017 |
| 32343 | HIS-351   |      1 | Spring   | 2018 |
| 45565 | CS-101    |      1 | Spring   | 2018 |
| 45565 | CS-319    |      1 | Spring   | 2018 |
| 76766 | BIO-101   |      1 | Summer   | 2017 |
| 76766 | BIO-301   |      1 | Summer   | 2018 |
| 83821 | CS-190    |      1 | Spring   | 2017 |
| 83821 | CS-190    |      2 | Spring   | 2017 |
| 83821 | CS-319    |      2 | Spring   | 2018 |
| 98345 | EE-181    |      1 | Spring   | 2017 |
+-------+-----------+--------+----------+------+
```

3. **Insert** following additional tuple in instructor ('10211', 'Smith', 'Biology', 66000) INSERT INTO instructor VALUES ('10211', 'Smith', 'Biology', 66000);
SELECT * FROM instructor;

```
+-------+-----------+-----------+--------+
| ID    | name      | dept_name | salary |
+-------+-----------+-----------+--------+
| 10101 | Srinivasan | Comp. Sci. |  65000 |
| 10211 | Smith     | Biology   |  66000 |
| 12121 | Wu        | Finance   |  90000 |
| 15151 | Mozart    | Music     |  40000 |
| 22222 | Einstein  | Physics   |  95000 |
| 32343 | El Said   | History   |  60000 |
| 33456 | Gold      | Physics   |  87000 |
| 45565 | Katz      | Comp. Sci. |  75000 |
| 58583 | Califieri | History   |   6200 |
| 76543 | Singh     | Finance   |  80000 |
| 76766 | Crick     | Biology   |  72000 |
| 83821 | Brandt    | Comp. Sci. |  92000 |
| 98345 | Kim       | Elec. Eng |  80000 |
+-------+-----------+-----------+--------+
```

4. **Delete** this tuple from instructor ('10211', 'Smith', 'Biology', 66000) DELETE FROM instructor WHERE ID=10211;
SELECT * FROM instructor;

```
+-------+-----------+-----------+--------+
| ID    | name      | dept_name | salary |
+-------+-----------+-----------+--------+
| 10101 | Srinivasan | Comp. Sci. |  65000 |
| 12121 | Wu        | Finance   |  90000 |
| 15151 | Mozart    | Music     |  40000 |
| 22222 | Einstein  | Physics   |  95000 |
| 32343 | El Said   | History   |  60000 |
| 33456 | Gold      | Physics   |  87000 |
| 45565 | Katz      | Comp. Sci. |  75000 |
| 58583 | Califieri | History   |   6200 |
| 76543 | Singh     | Finance   |  80000 |
| 76766 | Crick     | Biology   |  72000 |
| 83821 | Brandt    | Comp. Sci. |  92000 |
| 98345 | Kim       | Elec. Eng |  80000 |
+-------+-----------+-----------+--------+
```

5. **Select** tuples from instructor where dept_name = 'History'
SELECT * FROM instructor where dept_name='History';

```
+-------+-----------+------------+--------+
| ID    | name      | dept_name  | salary |
+-------+-----------+------------+--------+
| 32343 | El Said   | History    |  60000 |
| 58583 | Califieri | History    |   6200 |
+-------+-----------+------------+--------+
```

6. **Find** the Cartesian product instructor x
teaches. SELECT * FROM instructor CROSS JOIN
teaches;

```
+-------+-----------+------------+--------+-------+-----------+--------+----------+------+
| ID    | name      | dept_name  | salary | ID    | course_id | sec_id | semester | year |
+-------+-----------+------------+--------+-------+-----------+--------+----------+------+
| 98345 | Kim       | Elec. Eng  |  80000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 76766 | Crick     | Biology    |  72000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 76543 | Singh     | Finance    |  80000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 58583 | Califieri | History    |   6200 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 45565 | Katz      | Comp. Sci. |  75000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 33456 | Gold      | Physics    |  87000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 32343 | El Said   | History    |  60000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 22222 | Einstein  | Physics    |  95000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 15151 | Mozart    | Music      |  40000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 12121 | Wu        | Finance    |  90000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 10101 | Srinivasan| Comp. Sci. |  65000 | 10101 | CS-101    |      1 | Fall     | 2017 |
| 98345 | Kim       | Elec. Eng  |  80000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 76766 | Crick     | Biology    |  72000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 76543 | Singh     | Finance    |  80000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 58583 | Califieri | History    |   6200 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 45565 | Katz      | Comp. Sci. |  75000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 33456 | Gold      | Physics    |  87000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 32343 | El Said   | History    |  60000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 22222 | Einstein  | Physics    |  95000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 15151 | Mozart    | Music      |  40000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 12121 | Wu        | Finance    |  90000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 10101 | Srinivasan| Comp. Sci. |  65000 | 10101 | CS-315    |      1 | Spring   | 2018 |
| 98345 | Kim       | Elec. Eng  |  80000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 76766 | Crick     | Biology    |  72000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 76543 | Singh     | Finance    |  80000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 58583 | Califieri | History    |   6200 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 45565 | Katz      | Comp. Sci. |  75000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 33456 | Gold      | Physics    |  87000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 32343 | El Said   | History    |  60000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 22222 | Einstein  | Physics    |  95000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 15151 | Mozart    | Music      |  40000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 12121 | Wu        | Finance    |  90000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 10101 | Srinivasan| Comp. Sci. |  65000 | 10101 | CS-347    |      1 | Fall     | 2017 |
| 98345 | Kim       | Elec. Eng  |  80000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 76766 | Crick     | Biology    |  72000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 76543 | Singh     | Finance    |  80000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 58583 | Califieri | History    |   6200 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 45565 | Katz      | Comp. Sci. |  75000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 33456 | Gold      | Physics    |  87000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 32343 | El Said   | History    |  60000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 22222 | Einstein  | Physics    |  95000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 15151 | Mozart    | Music      |  40000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 12121 | Wu        | Finance    |  90000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 10101 | Srinivasan| Comp. Sci. |  65000 | 12121 | FIN-201   |      1 | Spring   | 2018 |
| 98345 | Kim       | Elec. Eng  |  80000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 76766 | Crick     | Biology    |  72000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 76543 | Singh     | Finance    |  80000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 58583 | Califieri | History    |   6200 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 45565 | Katz      | Comp. Sci. |  75000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 33456 | Gold      | Physics    |  87000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 32343 | El Said   | History    |  60000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 22222 | Einstein  | Physics    |  95000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 15151 | Mozart    | Music      |  40000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 12121 | Wu        | Finance    |  90000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 10101 | Srinivasan| Comp. Sci. |  65000 | 15151 | MU-199    |      1 | Spring   | 2015 |
| 98345 | Kim       | Elec. Eng  |  80000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 83821 | Brandt    | Comp. Sci. |  92000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 76766 | Crick     | Biology    |  72000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 76543 | Singh     | Finance    |  80000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 58583 | Califieri | History    |   6200 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 45565 | Katz      | Comp. Sci. |  75000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 33456 | Gold      | Physics    |  87000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 32343 | El Said   | History    |  60000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
| 22222 | Einstein  | Physics    |  95000 | 22222 | PHY-101   |      1 | Fall     | 2017 |
```

15151	Mozart	Music	40000	22222	PHY-101	1	Fall	2017
12121	Wu	Finance	90000	22222	PHY-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
98345	Kim	Elec. Eng	80000	32343	HIS-351	1	Spring	2018
83821	Brandt	Comp. Sci.	92000	32343	HIS-351	1	Spring	2018
76766	Crick	Biology	72000	32343	HIS-351	1	Spring	2018
76543	Singh	Finance	80000	32343	HIS-351	1	Spring	2018
58583	Califieri	History	6200	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	32343	HIS-351	1	Spring	2018
33456	Gold	Physics	87000	32343	HIS-351	1	Spring	2018
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
22222	Einstein	Physics	95000	32343	HIS-351	1	Spring	2018
15151	Mozart	Music	40000	32343	HIS-351	1	Spring	2018
12121	Wu	Finance	90000	32343	HIS-351	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	32343	HIS-351	1	Spring	2018
98345	Kim	Elec. Eng	80000	45565	CS-101	1	Spring	2018
83821	Brandt	Comp. Sci.	92000	45565	CS-101	1	Spring	2018
76766	Crick	Biology	72000	45565	CS-101	1	Spring	2018
76543	Singh	Finance	80000	45565	CS-101	1	Spring	2018
58583	Califieri	History	6200	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
33456	Gold	Physics	87000	45565	CS-101	1	Spring	2018
32343	El Said	History	60000	45565	CS-101	1	Spring	2018
22222	Einstein	Physics	95000	45565	CS-101	1	Spring	2018
15151	Mozart	Music	40000	45565	CS-101	1	Spring	2018
12121	Wu	Finance	90000	45565	CS-101	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	45565	CS-101	1	Spring	2018
98345	Kim	Elec. Eng	80000	45565	CS-319	1	Spring	2018
83821	Brandt	Comp. Sci.	92000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	45565	CS-319	1	Spring	2018
76543	Singh	Finance	80000	45565	CS-319	1	Spring	2018
58583	Califieri	History	6200	45565	CS-319	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
33456	Gold	Physics	87000	45565	CS-319	1	Spring	2018
32343	El Said	History	60000	45565	CS-319	1	Spring	2018
22222	Einstein	Physics	95000	45565	CS-319	1	Spring	2018
15151	Mozart	Music	40000	45565	CS-319	1	Spring	2018
12121	Wu	Finance	90000	45565	CS-319	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	45565	CS-319	1	Spring	2018
98345	Kim	Elec. Eng	80000	76766	BIO-101	1	Summer	2017
83821	Brandt	Comp. Sci.	92000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76543	Singh	Finance	80000	76766	BIO-101	1	Summer	2017
58583	Califieri	History	6200	76766	BIO-101	1	Summer	2017
45565	Katz	Comp. Sci.	75000	76766	BIO-101	1	Summer	2017
33456	Gold	Physics	87000	76766	BIO-101	1	Summer	2017
32343	El Said	History	60000	76766	BIO-101	1	Summer	2017
22222	Einstein	Physics	95000	76766	BIO-101	1	Summer	2017
15151	Mozart	Music	40000	76766	BIO-101	1	Summer	2017
12121	Wu	Finance	90000	76766	BIO-101	1	Summer	2017
10101	Srinivasan	Comp. Sci.	65000	76766	BIO-101	1	Summer	2017
98345	Kim	Elec. Eng	80000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	76766	BIO-301	1	Summer	2018
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
76543	Singh	Finance	80000	76766	BIO-301	1	Summer	2018
58583	Califieri	History	6200	76766	BIO-301	1	Summer	2018
45565	Katz	Comp. Sci.	75000	76766	BIO-301	1	Summer	2018
33456	Gold	Physics	87000	76766	BIO-301	1	Summer	2018
32343	El Said	History	60000	76766	BIO-301	1	Summer	2018
22222	Einstein	Physics	95000	76766	BIO-301	1	Summer	2018
15151	Mozart	Music	40000	76766	BIO-301	1	Summer	2018
12121	Wu	Finance	90000	76766	BIO-301	1	Summer	2018
10101	Srinivasan	Comp. Sci.	65000	76766	BIO-301	1	Summer	2018
98345	Kim	Elec. Eng	80000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
76766	Crick	Biology	72000	83821	CS-190	1	Spring	2017
76543	Singh	Finance	80000	83821	CS-190	1	Spring	2017
58583	Califieri	History	6200	83821	CS-190	1	Spring	2017
45565	Katz	Comp. Sci.	75000	83821	CS-190	1	Spring	2017
33456	Gold	Physics	87000	83821	CS-190	1	Spring	2017
32343	El Said	History	60000	83821	CS-190	1	Spring	2017
22222	Einstein	Physics	95000	83821	CS-190	1	Spring	2017
15151	Mozart	Music	40000	83821	CS-190	1	Spring	2017
12121	Wu	Finance	90000	83821	CS-190	1	Spring	2017
10101	Srinivasan	Comp. Sci.	65000	83821	CS-190	1	Spring	2017
98345	Kim	Elec. Eng	80000	83821	CS-190	2	Spring	2017

```
| 76766 | Crick     | Biology    | 72000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 76543 | Singh     | Finance    | 80000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 58583 | Califieri | History    |  6200 | 83821 | CS-190 |   2 | Spring | 2017 |
| 45565 | Katz      | Comp. Sci. | 75000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 33456 | Gold      | Physics    | 87000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 32343 | El Said   | History    | 60000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 22222 | Einstein  | Physics    | 95000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 15151 | Mozart    | Music      | 40000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 12121 | Wu        | Finance    | 90000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 10101 | Srinivasan| Comp. Sci. | 65000 | 83821 | CS-190 |   2 | Spring | 2017 |
| 98345 | Kim       | Elec. Eng  | 80000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 83821 | Brandt    | Comp. Sci. | 92000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 76766 | Crick     | Biology    | 72000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 76543 | Singh     | Finance    | 80000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 58583 | Califieri | History    |  6200 | 83821 | CS-319 |   2 | Spring | 2018 |
| 45565 | Katz      | Comp. Sci. | 75000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 33456 | Gold      | Physics    | 87000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 32343 | El Said   | History    | 60000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 22222 | Einstein  | Physics    | 95000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 15151 | Mozart    | Music      | 40000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 12121 | Wu        | Finance    | 90000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 10101 | Srinivasan| Comp. Sci. | 65000 | 83821 | CS-319 |   2 | Spring | 2018 |
| 98345 | Kim       | Elec. Eng  | 80000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 83821 | Brandt    | Comp. Sci. | 92000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 76766 | Crick     | Biology    | 72000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 76543 | Singh     | Finance    | 80000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 58583 | Califieri | History    |  6200 | 98345 | EE-181 |   1 | Spring | 2017 |
| 45565 | Katz      | Comp. Sci. | 75000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 33456 | Gold      | Physics    | 87000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 32343 | El Said   | History    | 60000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 22222 | Einstein  | Physics    | 95000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 15151 | Mozart    | Music      | 40000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 12121 | Wu        | Finance    | 90000 | 98345 | EE-181 |   1 | Spring | 2017 |
| 10101 | Srinivasan| Comp. Sci. | 65000 | 98345 | EE-181 |   1 | Spring | 2017 |
+-------+-----------+------------+-------+-------+--------+-----+--------+------+
```

7. **Find** the names of all instructors who have taught some course and the course_id SELECT i.name, t.course_id FROM instructor i INNER JOIN teaches t on i.ID= t.ID;

```
+------------+-----------+
| name       | course_id |
+------------+-----------+
| Srinivasan | CS-101    |
| Srinivasan | CS-315    |
| Srinivasan | CS-347    |
| Wu         | FIN-201   |
| Mozart     | MU-199    |
| Einstein   | PHY-101   |
| El Said    | HIS-351   |
| Katz       | CS-101    |
| Katz       | CS-319    |
| Crick      | BIO-101   |
| Crick      | BIO-301   |
| Brandt     | CS-190    |
| Brandt     | CS-190    |
| Brandt     | CS-319    |
| Kim        | EE-181    |
+------------+-----------+
```

8. **Find** the names of all instructors whose name includes the substring "dar".
SELECT name FROM instructor where name LIKE "%dar%";

9. **Find** the names of all instructors with salary between 90,000 and 100,000 (that is, ≥ 90,000 and ≤ 100,000)
SELECT name FROM instructor where salary>= 90000 AND salary<=100000;

```
+----------+
| name     |
+----------+
| Wu       |
| Einstein |
| Brandt   |
+----------+
```

EXPERIMENT 2

1. **Order** the tuples in the instructors relation as per their
salary. SELECT * FROM instructor ORDER BY salary;

```
+-------+-----------+-----------+--------+
| ID    | name      | dept_name | salary |
+-------+-----------+-----------+--------+
| 58583 | Califieri | History   |   6200 |
| 15151 | Mozart    | Music     |  40000 |
| 32343 | El Said   | History   |  60000 |
| 10101 | Srinivasan| Comp. Sci.|  65000 |
| 76766 | Crick     | Biology   |  72000 |
| 45565 | Katz      | Comp. Sci.|  75000 |
| 76543 | Singh     | Finance   |  80000 |
| 98345 | Kim       | Elec. Eng |  80000 |
| 33456 | Gold      | Physics   |  87000 |
| 12121 | Wu        | Finance   |  90000 |
| 83821 | Brandt    | Comp. Sci.|  92000 |
| 22222 | Einstein  | Physics   |  95000 |
+-------+-----------+-----------+--------+
```

2. **Find** courses that ran in Fall 2017 or in Spring 2018
SELECT DISTINCT course_id FROM teaches WHERE (semester='Fall'and
year=2017)OR (semester='Spring' and year=2018);

```
+-----------+
| course_id |
+-----------+
| CS-101    |
| CS-315    |
| CS-347    |
| FIN-201   |
| PHY-101   |
| HIS-351   |
| CS-319    |
+-----------+
```

3. **Find** courses that ran in Fall 2017 and in Spring 2018
SELECT DISTINCT course_id FROM teaches WHERE (semester='Fall'and year=2017)
AND (semester='Spring' and year=2018);

4. **Find** courses that ran in Fall 2017 but not in Spring 2018
SELECT DISTINCT course_id FROM teaches t1 WHERE (t1.semester='Fall'and
t1.year=2017) AND NOT EXISTS (SELECT 1 FROM teaches t2 WHERE t2.course_id=
t1.course_id AND t2.semester='Spring' AND t2.year=2018);

```
+-----------+
| course_id |
+-----------+
| CS-347    |
| PHY-101   |
+-----------+
```

5. **Insert** following additional tuples in instructor :('10211', 'Smith', 'Biology', 66000), ('10212', 'Tom', 'Biology', NULL)
INSERT INTO instructor VALUES ('10211', 'Smith', 'Biology', 66000), ('10212', 'Tom', 'Biology', NULL);
SELECT * FROM instructor;

```
+-------+-----------+------------+--------+
| ID    | name      | dept_name  | salary |
+-------+-----------+------------+--------+
| 10101 | Srinivasan | Comp. Sci. |  65000 |
| 10211 | Smith     | Biology    |  66000 |
| 10212 | Tom       | Biology    |   NULL |
| 12121 | Wu        | Finance    |  90000 |
| 15151 | Mozart    | Music      |  40000 |
| 22222 | Einstein  | Physics    |  95000 |
| 32343 | El Said   | History    |  60000 |
| 33456 | Gold      | Physics    |  87000 |
| 45565 | Katz      | Comp. Sci. |  75000 |
| 58583 | Califieri | History    |   6200 |
| 76543 | Singh     | Finance    |  80000 |
| 76766 | Crick     | Biology    |  72000 |
| 83821 | Brandt    | Comp. Sci. |  92000 |
| 98345 | Kim       | Elec. Eng  |  80000 |
+-------+-----------+------------+--------+
```

6. **Find** all instructors whose salary is null.
SELECT name FROM instructor WHERE salary IS NULL;

```
+------+
| name |
+------+
| Tom  |
+------+
```

7. **Find** the average salary of instructors in the Computer Science department.
SELECT AVG(salary) AS avg_salary FROM instructor WHERE dept_name='Comp. Sci.';

```
+------------+
| avg_salary |
+------------+
| 77333.3333 |
+------------+
```

EXPERIMENT 3

1. **Find** the total number of instructors who teach a course in the Spring 2018 semester. SELECT COUNT(DISTINCT ID) AS total_instructors FROM teaches WHERE semester='Spring' AND year=2018;

```
+-------------------+
| total_instructors |
+-------------------+
|                 5 |
+-------------------+
```

2. **Find** the number of tuples in the teaches relation SELECT COUNT(*) AS num_tuples FROM teaches;

```
+-------------+
| num_tuples  |
+-------------+
|          15 |
+-------------+
```

3. **Find** the average salary of instructors in each department
SELECT dept_name, AVG(salary) as avg_salary FROM instructor GROUP BY dept_name;

```
+------------+------------+
| dept_name  | avg_salary |
+------------+------------+
| Comp. Sci. | 77333.3333 |
| Biology    | 69000.0000 |
| Finance    | 85000.0000 |
| Music      | 40000.0000 |
| Physics    | 91000.0000 |
| History    | 33100.0000 |
| Elec. Eng  | 80000.0000 |
+------------+------------+
```

4. **Find** the names and average salaries of all departments whose average salary is greater than 42000
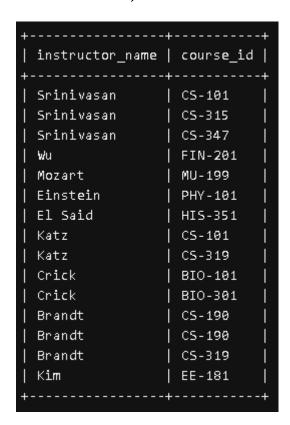SELECT dept_name, AVG(salary) as avg_salary FROM instructor GROUP BY dept_name HAVING AVG(salary)>42000;

```
+------------+------------+
| dept_name  | avg_salary |
+------------+------------+
| Comp. Sci. | 77333.3333 |
| Biology    | 69000.0000 |
| Finance    | 85000.0000 |
| Physics    | 91000.0000 |
| Elec. Eng  | 80000.0000 |
+------------+------------+
```

5. Name all instructors whose name is neither "Mozart" nor Einstein"
SELECT name FROM instructor WHERE name NOT IN ("Mozart","Einstein");

```
+-----------+
| name      |
+-----------+
| Srinivasan |
| Smith     |
| Tom       |
| Wu        |
| El Said   |
| Gold      |
| Katz      |
| Califieri |
| Singh     |
| Crick     |
| Brandt    |
| Kim       |
+-----------+
```

6. **Find** names of instructors with salary greater than that of some (at least one) instructor in the Biology department.
SELECT l.name FROM instructor l WHERE l.salary > (SELECT salary FROM instructor WHERE dept_name='Biology' AND name="Crick");

```
+----------+
| name     |
+----------+
| Wu       |
| Einstein |
| Gold     |
| Katz     |
| Singh    |
| Brandt   |
| Kim      |
+----------+
```

7. **Find** the names of all instructors whose salary is greater than the salary of all instructors in the Biology department.
SELECT l.name FROM instructor l WHERE l.salary > (SELECT max(salary) FROM instructor WHERE dept_name='Biology');

```
+-----------+
| name      |
+-----------+
| Wu        |
| Einstein  |
| Gold      |
| Katz      |
| Singh     |
| Brandt    |
| Kim       |
+-----------+
```

8. **Find** the average instructors' salaries of those departments where the average salary is greater than 42,000
SELECT dept_name, AVG(salary) as average_salary FROM instructor GROUP BY dept_name HAVING AVG(salary)>42000;

```
+------------+----------------+
| dept_name  | average_salary |
+------------+----------------+
| Comp. Sci. |     77333.3333 |
| Biology    |     69000.0000 |
| Finance    |     85000.0000 |
| Physics    |     91000.0000 |
| Elec. Eng  |     80000.0000 |
+------------+----------------+
```

EXPERIMENT 4

1. **Find** all departments where the total salary is greater than the average of the total salary at all departments

   SELECT dept_name, SUM(salary) AS total_salary
   FROM instructor GROUP BY dept_name
   HAVING SUM(salary) > (SELECT AVG(total_salary) FROM (SELECT SUM(salary) AS total_salary FROM instructor GROUP BY dept_name) AS avg_salary);

   ```
   +------------+--------------+
   | dept_name  | total_salary |
   +------------+--------------+
   | Comp. Sci. |       232000 |
   | Finance    |       170000 |
   | Physics    |       182000 |
   +------------+--------------+
   ```

2. List the names of instructors along with the course ID of the courses that they taught

   SELECT i.name AS instructor_name, t.course_id
   FROM instructor i JOIN teaches t ON i.ID = t.ID;

   ```
   +-----------------+-----------+
   | instructor_name | course_id |
   +-----------------+-----------+
   | Srinivasan      | CS-101    |
   | Srinivasan      | CS-315    |
   | Srinivasan      | CS-347    |
   | Wu              | FIN-201   |
   | Mozart          | MU-199    |
   | Einstein        | PHY-101   |
   | El Said         | HIS-351   |
   | Katz            | CS-101    |
   | Katz            | CS-319    |
   | Crick           | BIO-101   |
   | Crick           | BIO-301   |
   | Brandt          | CS-190    |
   | Brandt          | CS-190    |
   | Brandt          | CS-319    |
   | Kim             | EE-181    |
   +-----------------+-----------+
   ```

3. List the names of instructors along with the course ID of the courses that they taught. In case, an instructor teaches no courses keep the course ID as null.
   SELECT i.name AS instructor_name, t.course_id

FROM instructor i LEFT JOIN teaches t ON i.ID = t.ID;

```
+------------------+-----------+
| instructor_name  | course_id |
+------------------+-----------+
| Srinivasan       | CS-101    |
| Srinivasan       | CS-315    |
| Srinivasan       | CS-347    |
| Wu               | FIN-201   |
| Mozart           | MU-199    |
| Einstein         | PHY-101   |
| El Said          | HIS-351   |
| Gold             | NULL      |
| Katz             | CS-101    |
| Katz             | CS-319    |
| Califieri        | NULL      |
| Singh            | NULL      |
| Crick            | BIO-101   |
| Crick            | BIO-301   |
| Brandt           | CS-190    |
| Brandt           | CS-190    |
| Brandt           | CS-319    |
| Kim              | EE-181    |
+------------------+-----------+
```

4. **Create** a view of instructors without their salary called faculty CREATE VIEW faculty AS SELECT ID, name, dept_name
FROM instructor;
SELECT * FROM faculty;

```
+-------+-----------+------------+
| ID    | name      | dept_name  |
+-------+-----------+------------+
| 10101 | Srinivasan | Comp. Sci. |
| 12121 | Wu         | Finance    |
| 15151 | Mozart     | Music      |
| 22222 | Einstein   | Physics    |
| 32343 | El Said    | History    |
| 33456 | Gold       | Physics    |
| 45565 | Katz       | Comp. Sci. |
| 58583 | Califieri  | History    |
| 76543 | Singh      | Finance    |
| 76766 | Crick      | Biology    |
| 83821 | Brandt     | Comp. Sci. |
| 98345 | Kim        | Elec. Eng  |
+-------+-----------+------------+
```

5. Give select privileges on the view faculty to the new user.

GRANT SELECT ON faculty TO new_user;

EXPERIMENT 5

1. **Create** a view of instructors without their salary called faculty

   CREATE VIEW faculty1 AS SELECT ID, name, dept_name
   FROM instructor;
   SELECT * FROM faculty1;

```
+-------+-----------+------------+
| ID    | name      | dept_name  |
+-------+-----------+------------+
| 10101 | Srinivasan | Comp. Sci. |
| 12121 | Wu        | Finance    |
| 15151 | Mozart    | Music      |
| 22222 | Einstein  | Physics    |
| 32343 | El Said   | History    |
| 33456 | Gold      | Physics    |
| 45565 | Katz      | Comp. Sci. |
| 58583 | Califieri | History    |
| 76543 | Singh     | Finance    |
| 76766 | Crick     | Biology    |
| 83821 | Brandt    | Comp. Sci. |
| 98345 | Kim       | Elec. Eng  |
+-------+-----------+------------+
```

2. **Create** a view of department salary totals

   CREATE VIEW department_salary_totals AS SELECT dept_name, SUM(salary) AS
   total_salary FROM instructor GROUP BY dept_name;
   SELECT * FROM department_salary_totals;

```
+------------+--------------+
| dept_name  | total_salary |
+------------+--------------+
| Comp. Sci. |       232000 |
| Finance    |       170000 |
| Music      |        40000 |
| Physics    |       182000 |
| History    |        66200 |
| Biology    |        72000 |
| Elec. Eng  |        80000 |
+------------+--------------+
```

3. **Create** a role of student

   CREATE ROLE student;

4. Give select privileges on the view faculty to the role student.

   GRANT SELECT ON faculty TO student;

5. Create a new user and assign her the role of student.
    CREATE USER khaif@localhost IDENTIFIED BY
    'khaifhasan1331'; GRANT student TO
    khaif@localhost;

6. Login as this new user and find all instructors in the Biology department.
    GRANT ALL PRIVILEGES ON student.* TO khaif@localhost;

    SELECT * FROM faculty WHERE dept_name = 'Biology';

	ID	name	dept_name
▶	10211	Smith	Biology
	10212	Tom	Biology
	76766	Crick	Biology

7. Revoke privileges of the new user
    REVOKE student FROM khaif@localhost;
8. Remove the role of student.
    DROP ROLE student;
9. Give select privileges on the view faculty to the new
    user. GRANT SELECT ON faculty TO khaif@localhost;

10. Login as this new user and find all instructors in the finance department.
    SELECT * FROM faculty WHERE dept_name = 'Finance';

	ID	name	dept_name
▶	12121	Wu	Finance
	76543	Singh	Finance

11. Login again as root user
12. **Create** table teaches2 with same columns as teaches but with additional constraint
    that that semester is one of fall, winter, spring or summer

    CREATE TABLE teaches2 (
      ID INT NOT NULL,
      course_id VARCHAR(255) NOT NULL,
      sec_id INT NOT NULL,
      semester VARCHAR(255) NOT NULL CHECK (semester IN ('Fall', 'Winter', 'Spring',
    'Summer')),
      year INT NOT NULL,
      FOREIGN KEY (ID) REFERENCES instructor(ID)
    );

13. **Create** index ID column of teaches. Compare the difference in time to obtain
    query results with or without index.
    CREATE INDEX idx_ID ON teaches (ID);

14. Drop the index to free up the space.
    DROP INDEX idx_ID ON teaches;

EXPERIMENT 6

Accessing the database through Python
1. **Insert** following additional tuple in instructor : ('10211', 'Smith', 'Biology', 66000)
2. **Delete** this tuple from instructor : ('10211', 'Smith', 'Biology', 66000)
3. **Select** tuples from instructor where dept_name = 'History'
4. **Find** the Cartesian product instructor x teaches.
5. **Find** the names of all instructors who have taught some course and the course_id
6. **Find** the names of all instructors whose name includes the substring "dar".
7. Find the names of all instructors with salary between 90,000 and 100,000 (that is, $\geq$ 90,000 and $\leq$ 100,000)

```python
import mysql.connector

conn = mysql.connector.connect(
    host='khaif@localhost',
    user='root',
    password='khaifhasan1331',
    database='adbms'
)

cursor = conn.cursor()

create_table_query = """
CREATE TABLE instructor (
ID INT PRIMARY KEY,
 name VARCHAR(255) NOT NULL,
 dept_name VARCHAR(255) NOT NULL,
 salary INT
)
"""
cursor.execute(create_table_query)

insert_query = """
INSERT INTO instructor (ID, name, dept_name, salary) VALUES
(10101, 'Srinivasan', 'Comp. Sci.', 65000),
(12121, 'Wu', 'Finance', 90000),
(15151, 'Mozart', 'Music', 40000),
(22222, 'Einstein', 'Physics', 95000),
(32343, 'El Said', 'History', 60000),
(33456, 'Gold', 'Physics', 87000),
(45565, 'Katz', 'Comp. Sci.', 75000),
(58583, 'Califieri', 'History', 62000),
(76543, 'Singh', 'Finance', 80000),
(76766, 'Crick', 'Biology', 72000),
(83821, 'Brandt', 'Comp. Sci.', 92000),
(98345, 'Kim', 'Elec. Eng', 80000)
"""
cursor.execute(insert_query)
```

```python
create_table_query = """
CREATE TABLE teaches (
ID INT,
 course_id VARCHAR(255),
 sec_id INT,
 semester VARCHAR(255),
 year INT,
 FOREIGN KEY (ID) REFERENCES instructor(ID)
)
"""
cursor.execute(create_table_query)

insert_query = """
INSERT INTO teaches (ID, course_id, sec_id, semester, year) VALUES
(10101, 'CS-101', 1, 'Fall', 2017),
(10101, 'CS-315', 1, 'Spring', 2018),
(10101, 'CS-347', 1, 'Fall', 2017),
(12121, 'FIN-201', 1, 'Spring', 2018),
(15151, 'MU-199', 1, 'Spring', 2015),
(22222, 'PHY-101', 1, 'Fall', 2017),
(32343, 'HIS-351', 1, 'Spring', 2018),
(45565, 'CS-101', 1, 'Spring', 2018),
(45565, 'CS-319', 1, 'Spring', 2018),
(76766, 'BIO-101', 1, 'Summer', 2017),
(76766, 'BIO-301', 1, 'Summer', 2018),
(83821, 'CS-190', 1, 'Spring', 2017),
(83821, 'CS-190', 2, 'Spring', 2017),
(83821, 'CS-319', 2, 'Spring', 2018),
(98345, 'EE-181', 1, 'Spring', 2017)
"""
cursor.execute(insert_query)

# 1
insert_query = """
INSERT INTO instructor (ID, name, dept_name, salary) VALUES
('10211', 'Smith', 'Biology', 66000)
"""
cursor.execute(insert_query)

# 2
tuple_to_delete = ('10211', 'Smith', 'Biology', 66000)

delete_query = "DELETE FROM instructor WHERE ID = %s AND name = %s AND dept_name = %s
AND salary = %s"
cursor.execute(delete_query, tuple_to_delete)

# 3
dept_name = 'History'
```

```python
select_query = "SELECT * FROM instructor WHERE dept_name = %s"
cursor.execute(select_query, (dept_name,))

results = cursor.fetchall()

for row in results:
    print(row)

# 4
cartesian_query = """
SELECT * FROM instructor, teaches
"""

cursor.execute(cartesian_query)

results = cursor.fetchall()

for row in results:
    print(row)

# 5
query = """
SELECT DISTINCT instructor.name, teaches.course_id
FROM instructor
JOIN teaches ON instructor.ID = teaches.ID
"""

# Execute the query
cursor.execute(query)

# Fetch the results
results = cursor.fetchall()

# Print the results
for row in results:
    print(row)

# 6
query = """
SELECT name
FROM instructor
WHERE name LIKE '%dar%'
"""

cursor.execute(query)

results = cursor.fetchall()
```

```python
for row in results:
    print(row[0])

# 7
query = """
SELECT name
FROM instructor
WHERE salary BETWEEN 90000 AND 100000
"""

cursor.execute(query)

results = cursor.fetchall()

for row in results:
    print(row[0])

conn.commit()

cursor.close()
conn.close()
```

```
Question 3
(32343, 'El Said', 'History', 60000)
(58583, 'Califieri', 'History', 62000)


Question 4
(98345, 'Kim', 'Elec. Eng', 80000, 10101, 'CS-101', 1, 'Fall', 2017)
(83821, 'Brandt', 'Comp. Sci.', 92000, 10101, 'CS-101', 1, 'Fall', 2017)
(76766, 'Crick', 'Biology', 72000, 10101, 'CS-101', 1, 'Fall', 2017)
(76543, 'Singh', 'Finance', 80000, 10101, 'CS-101', 1, 'Fall', 2017)
(58583, 'Califieri', 'History', 62000, 10101, 'CS-101', 1, 'Fall', 2017)
(45565, 'Katz', 'Comp. Sci.', 75000, 10101, 'CS-101', 1, 'Fall', 2017)
(33456, 'Gold', 'Physics', 87000, 10101, 'CS-101', 1, 'Fall', 2017)
(32343, 'El Said', 'History', 60000, 10101, 'CS-101', 1, 'Fall', 2017)
(22222, 'Einstein', 'Physics', 95000, 10101, 'CS-101', 1, 'Fall', 2017)
(15151, 'Mozart', 'Music', 40000, 10101, 'CS-101', 1, 'Fall', 2017)
(12121, 'Wu', 'Finance', 90000, 10101, 'CS-101', 1, 'Fall', 2017)
(10101, 'Srinivasan', 'Comp. Sci.', 65000, 10101, 'CS-101', 1, 'Fall', 2017)
(98345, 'Kim', 'Elec. Eng', 80000, 10101, 'CS-315', 1, 'Spring', 2018)
(83821, 'Brandt', 'Comp. Sci.', 92000, 10101, 'CS-315', 1, 'Spring', 2018)
(76766, 'Crick', 'Biology', 72000, 10101, 'CS-315', 1, 'Spring', 2018)
(76543, 'Singh', 'Finance', 80000, 10101, 'CS-315', 1, 'Spring', 2018)
(58583, 'Califieri', 'History', 62000, 10101, 'CS-315', 1, 'Spring', 2018)
(45565, 'Katz', 'Comp. Sci.', 75000, 10101, 'CS-315', 1, 'Spring', 2018)
(33456, 'Gold', 'Physics', 87000, 10101, 'CS-315', 1, 'Spring', 2018)
(32343, 'El Said', 'History', 60000, 10101, 'CS-315', 1, 'Spring', 2018)
(22222, 'Einstein', 'Physics', 95000, 10101, 'CS-315', 1, 'Spring', 2018)
(15151, 'Mozart', 'Music', 40000, 10101, 'CS-315', 1, 'Spring', 2018)
(12121, 'Wu', 'Finance', 90000, 10101, 'CS-315', 1, 'Spring', 2018)
(10101, 'Srinivasan', 'Comp. Sci.', 65000, 10101, 'CS-315', 1, 'Spring', 2018)
(98345, 'Kim', 'Elec. Eng', 80000, 10101, 'CS-347', 1, 'Fall', 2017)
(83821, 'Brandt', 'Comp. Sci.', 92000, 10101, 'CS-347', 1, 'Fall', 2017)
(76766, 'Crick', 'Biology', 72000, 10101, 'CS-347', 1, 'Fall', 2017)
(76543, 'Singh', 'Finance', 80000, 10101, 'CS-347', 1, 'Fall', 2017)
(58583, 'Califieri', 'History', 62000, 10101, 'CS-347', 1, 'Fall', 2017)
(45565, 'Katz', 'Comp. Sci.', 75000, 10101, 'CS-347', 1, 'Fall', 2017)
(33456, 'Gold', 'Physics', 87000, 10101, 'CS-347', 1, 'Fall', 2017)
(32343, 'El Said', 'History', 60000, 10101, 'CS-347', 1, 'Fall', 2017)
(22222, 'Einstein', 'Physics', 95000, 10101, 'CS-347', 1, 'Fall', 2017)
(15151, 'Mozart', 'Music', 40000, 10101, 'CS-347', 1, 'Fall', 2017)
(12121, 'Wu', 'Finance', 90000, 10101, 'CS-347', 1, 'Fall', 2017)
(10101, 'Srinivasan', 'Comp. Sci.', 65000, 10101, 'CS-347', 1, 'Fall', 2017)
(98345, 'Kim', 'Elec. Eng', 80000, 12121, 'FIN-201', 1, 'Spring', 2018)
(83821, 'Brandt', 'Comp. Sci.', 92000, 12121, 'FIN-201', 1, 'Spring', 2018)
(76766, 'Crick', 'Biology', 72000, 12121, 'FIN-201', 1, 'Spring', 2018)
(76543, 'Singh', 'Finance', 80000, 12121, 'FIN-201', 1, 'Spring', 2018)
(58583, 'Califieri', 'History', 62000, 12121, 'FIN-201', 1, 'Spring', 2018)
(45565, 'Katz', 'Comp. Sci.', 75000, 12121, 'FIN-201', 1, 'Spring', 2018)
```

```
(83821, 'Brandt', 'Comp. Sci.', 92000, 83821, 'CS-319', 2, 'Spring', 2018)
(76766, 'Crick', 'Biology', 72000, 83821, 'CS-319', 2, 'Spring', 2018)
(76543, 'Singh', 'Finance', 80000, 83821, 'CS-319', 2, 'Spring', 2018)
(58583, 'Califieri', 'History', 62000, 83821, 'CS-319', 2, 'Spring', 2018)
(45565, 'Katz', 'Comp. Sci.', 75000, 83821, 'CS-319', 2, 'Spring', 2018)
(33456, 'Gold', 'Physics', 87000, 83821, 'CS-319', 2, 'Spring', 2018)
(32343, 'El Said', 'History', 60000, 83821, 'CS-319', 2, 'Spring', 2018)
(22222, 'Einstein', 'Physics', 95000, 83821, 'CS-319', 2, 'Spring', 2018)
(15151, 'Mozart', 'Music', 40000, 83821, 'CS-319', 2, 'Spring', 2018)
(12121, 'Wu', 'Finance', 90000, 83821, 'CS-319', 2, 'Spring', 2018)
(10101, 'Srinivasan', 'Comp. Sci.', 65000, 83821, 'CS-319', 2, 'Spring', 2018)
(98345, 'Kim', 'Elec. Eng', 80000, 98345, 'EE-181', 1, 'Spring', 2017)
(83821, 'Brandt', 'Comp. Sci.', 92000, 98345, 'EE-181', 1, 'Spring', 2017)
(76766, 'Crick', 'Biology', 72000, 98345, 'EE-181', 1, 'Spring', 2017)
(76543, 'Singh', 'Finance', 80000, 98345, 'EE-181', 1, 'Spring', 2017)
(58583, 'Califieri', 'History', 62000, 98345, 'EE-181', 1, 'Spring', 2017)
(45565, 'Katz', 'Comp. Sci.', 75000, 98345, 'EE-181', 1, 'Spring', 2017)
(33456, 'Gold', 'Physics', 87000, 98345, 'EE-181', 1, 'Spring', 2017)
(32343, 'El Said', 'History', 60000, 98345, 'EE-181', 1, 'Spring', 2017)
(22222, 'Einstein', 'Physics', 95000, 98345, 'EE-181', 1, 'Spring', 2017)
(15151, 'Mozart', 'Music', 40000, 98345, 'EE-181', 1, 'Spring', 2017)
(12121, 'Wu', 'Finance', 90000, 98345, 'EE-181', 1, 'Spring', 2017)
(10101, 'Srinivasan', 'Comp. Sci.', 65000, 98345, 'EE-181', 1, 'Spring', 2017)


Question 5
('Srinivasan', 'CS-101')
('Srinivasan', 'CS-315')
('Srinivasan', 'CS-347')
('Wu', 'FIN-201')
('Mozart', 'MU-199')
('Einstein', 'PHY-101')
('El Said', 'HIS-351')
('Katz', 'CS-101')
('Katz', 'CS-319')
('Crick', 'BIO-101')
('Crick', 'BIO-301')
('Brandt', 'CS-190')
('Brandt', 'CS-319')
('Kim', 'EE-181')


Question 6


Question 7
Wu
Einstein
Brandt
```

EXPERIMENT 7

1. **Order** the tuples in the instructors relation as per their salary.
2. **Find** courses that ran in Fall 2017 or in Spring 2018
3. **Find** courses that ran in Fall 2017 and in Spring 2018
4. **Find** courses that ran in Fall 2017 but not in Spring 2018
5. **Insert** following additional tuples in instructor ('10211', 'Smith', 'Biology', 66000) ('10212', 'Tom', 'Biology', NULL
6. **Find** all instructors whose salary is null.
7. **Find** the average salary of instructors in the Computer Science department.
8. **Find** the total number of instructors who teach a course in the Spring 2018 semester.
9. **Find** the number of tuples in the teaches relation
10. **Find** the average salary of instructors in each department
11. **Find** the names and average salaries of all departments whose average salary is greater than 42000
12. Name all instructors whose name is neither "Mozart" nor Einstein".
13. **Find** names of instructors with salary greater than that of some (at least one) instructor in the Biology department.
14. **Find** the names of all instructors whose salary is greater than the salary of all instructors in the Biology department.
15. **Find** the average instructors' salaries of those departments where the average salary is greater than 42,000.
16. **Find** all departments where the total salary is greater than the average of the total salary at all departments
17. List the names of instructors along with the course ID of the courses that they taught.
18. List the names of instructors along with the course ID of the courses that they taught. In case, an instructor teaches no courses keep the course ID as null.

```python
import mysql.connector

conn = mysql.connector.connect(
    host='khaifhasanlocalhost',
    user='root',
    password='khaifhasan1331',
    database='adbms'
)

cursor = conn.cursor()

# Order the tuples in the instructors relation as per their salary.
order_by_salary_query = """
SELECT * FROM instructor
ORDER BY salary
"""

cursor.execute(order_by_salary_query)
```

```python
results = cursor.fetchall()

print("Question1:")
for row in results:
    print(row)
print("\n")

# Find courses that ran in Fall 2017 or in Spring 2018
courses_in_spring_or_fall = """
SELECT DISTINCT course_id FROM teaches WHERE (semester='Fall'and year=2017)OR
(semester='Spring' and year=2018)
"""

cursor.execute(courses_in_spring_or_fall)

results = cursor.fetchall()

print("Question2:")
for row in results:
    print(row)
print("\n")

# Find courses that ran in Fall 2017 and in Spring 2018
courses_in_spring_and_fall = """
SELECT DISTINCT course_id FROM teaches WHERE (semester='Fall'and year=2017) AND
(semester='Spring' and year=2018)
"""

cursor.execute(courses_in_spring_and_fall)

results = cursor.fetchall()

print("Question3:")
for row in results:
    print(row)
print("\n")

# Find courses that ran in Fall 2017 but not in Spring 2018
course_in_fall_only = """
SELECT DISTINCT course_id FROM teaches t1 WHERE (t1.semester='Fall'and t1.year=2017) AND
NOT EXISTS (SELECT 1 FROM teaches t2 WHERE t2.course_id= t1.course_id AND
t2.semester='Spring' AND t2.year=2018)
"""

cursor.execute(course_in_fall_only)

results = cursor.fetchall()
```

```python
print("Question4:")
for row in results:
    print(row)
print("\n")


#  Insert following additional tuples in instructor
insert_tuples= """
INSERT INTO instructor VALUES ('10211', 'Smith', 'Biology', 66000), ('10212',
'Tom', 'Biology', NULL )
"""

cursor.execute(insert_tuples)

select_table = """
SELECT * FROM instructor
"""

cursor.execute(select_table)

results = cursor.fetchall()

print("Question5:")
for row in results:
    print(row)
print("\n")


#  Find all instructors whose salary is null.
instructor_salary_null = """
SELECT name FROM instructor WHERE salary IS NULL
"""

cursor.execute(instructor_salary_null)

results = cursor.fetchall()

print("Question6:")
for row in results:
    print(row)
print("\n")


# Find the average salary of instructors in the Computer Science department.
avg_cs_dept = """
SELECT AVG(salary) AS avg_salary FROM instructor WHERE dept_name='Comp. Sci.'
"""


cursor.execute(avg_cs_dept)
```

```python
results = cursor.fetchall()

print("Question7:")
for row in results:
    print(row)
print("\n")


# Find the total number of instructors who teach a course in the Spring 2018 semester.
instructors_spring = """
SELECT COUNT(DISTINCT ID) AS total_instructors FROM teaches WHERE semester='Spring' AND
year=2018
"""

cursor.execute(instructors_spring)

results = cursor.fetchall()

print("Question8:")
for row in results:
    print(row)
print("\n")


# Find the number of tuples in the teaches relation
teaches_count = """
SELECT COUNT(*) AS num_tuples FROM teaches
"""

cursor.execute(teaches_count)

results = cursor.fetchall()

print("Question9:")
for row in results:
    print(row)
print("\n")


# Find the average salary of instructors in each department
avg_instructor = """
SELECT dept_name, AVG(salary) as avg_salary FROM instructor GROUP BY dept_name
"""

cursor.execute(avg_instructor)

results = cursor.fetchall()

print("Question10:")
for row in results:
    print(row)
```

```python
print("\n")

# Find the names and average salaries of all departments whose average salary is greater than
42000
avg_salary_greater = """
SELECT dept_name, AVG(salary) as avg_salary FROM instructor GROUP BY dept_name HAVING
AVG(salary)>42000
"""

cursor.execute(avg_salary_greater)

results = cursor.fetchall()

print("Question11:")
for row in results:
    print(row)
print("\n")

#  Name all instructors whose name is neither "Mozart" nor Einstein".
instructor_name = """
SELECT name FROM instructor WHERE name NOT IN ("Mozart","Einstein")
"""

cursor.execute(instructor_name)

results = cursor.fetchall()

print("Question12:")
for row in results:
    print(row)
print("\n")

# Find names of instructors with salary greater than that of some (at least one) instructor in the
Biology department.
salary_greater= """
SELECT l.name FROM instructor l WHERE l.salary > (SELECT salary FROM instructor WHERE
dept_name='Biology' AND name="Crick")
"""

cursor.execute(salary_greater)

results = cursor.fetchall()

print("Question13:")
for row in results:
    print(row)
print("\n")
```

```python
# Find the names of all instructors whose salary is greater than the salary of all instructors in the
Biology department.
salary_greater_biology = """
SELECT l.name FROM instructor l WHERE l.salary > (SELECT max(salary) FROM instructor WHERE
dept_name='Biology')
"""

cursor.execute(salary_greater_biology)

results = cursor.fetchall()

print("Question14:")
for row in results:
    print(row)
print("\n")


# Find the average instructors' salaries of those departments where the average salary is greater
than 42,000.
avg_instructor_greater = """
SELECT dept_name, AVG(salary) as average_salary FROM instructor GROUP BY dept_name HAVING
AVG(salary)>42000
"""

cursor.execute(avg_instructor_greater)

results = cursor.fetchall()

print("Question15:")
for row in results:
    print(row)
print("\n")

# Find all departments where the total salary is greater than the average of the total salary at all
department_salary = """
SELECT dept_name
FROM (
    SELECT dept_name, SUM(salary) AS total_salary
    FROM instructor
    GROUP BY dept_name
) AS department_total_salary
WHERE total_salary > (
    SELECT AVG(total_salary)
    FROM (
        SELECT SUM(salary) AS total_salary
        FROM instructor
        GROUP BY dept_name
    ) AS avg_total_salary
)
```

```python
"""

cursor.execute(department_salary)

results = cursor.fetchall()

print("Question16:")
for row in results:
    print(row)
print("\n")

#  List the names of instructors along with the course ID of the courses that they taught
instructor_name_with_courseID = """
SELECT instructor.name, teaches.course_id
FROM instructor
JOIN teaches ON instructor.ID = teaches.ID
"""

cursor.execute(instructor_name_with_courseID)

results = cursor.fetchall()

print("Question17:")
for row in results:
    print(row)
print("\n")

#  List the names of instructors along with the course ID of the courses that they taught. In case, an
instructor teaches no courses keep the course ID as null.
instructor_name_with_courseID_with_null = """
SELECT instructor.name, teaches.course_id
FROM instructor
LEFT JOIN teaches ON instructor.ID = teaches.ID
"""

cursor.execute(instructor_name_with_courseID_with_null)

results = cursor.fetchall()

print("Question18:")
for row in results:
    print(row)
print("\n")
```

```
Question1:
(15151, 'Mozart', 'Music', 40000)
(32343, 'El Said', 'History', 60000)
(58583, 'Califieri', 'History', 62000)
(10101, 'Srinivasan', 'Comp. Sci.', 65000)
(76766, 'Crick', 'Biology', 72000)
(45565, 'Katz', 'Comp. Sci.', 75000)
(76543, 'Singh', 'Finance', 80000)
(98345, 'Kim', 'Elec. Eng', 80000)
(33456, 'Gold', 'Physics', 87000)
(12121, 'Wu', 'Finance', 90000)
(83821, 'Brandt', 'Comp. Sci.', 92000)
(22222, 'Einstein', 'Physics', 95000)


Question2:
('CS-101',)
('CS-315',)
('CS-347',)
('FIN-201',)
('PHY-101',)
('HIS-351',)
('CS-319',)


Question3:


Question4:
('CS-347',)
('PHY-101',)


Question5:
(10101, 'Srinivasan', 'Comp. Sci.', 65000)
(10211, 'Smith', 'Biology', 66000)
(10212, 'Tom', 'Biology', None)
(12121, 'Wu', 'Finance', 90000)
(15151, 'Mozart', 'Music', 40000)
(22222, 'Einstein', 'Physics', 95000)
(32343, 'El Said', 'History', 60000)
(33456, 'Gold', 'Physics', 87000)
(45565, 'Katz', 'Comp. Sci.', 75000)
(58583, 'Califieri', 'History', 62000)
(76543, 'Singh', 'Finance', 80000)
(76766, 'Crick', 'Biology', 72000)
(83821, 'Brandt', 'Comp. Sci.', 92000)
(98345, 'Kim', 'Elec. Eng', 80000)
```

```
Question6:
('Tom',)


Question7:
(Decimal('77333.3333'),)


Question8:
(5,)


Question9:
(15,)


Question10:
('Comp. Sci.', Decimal('77333.3333'))
('Biology', Decimal('69000.0000'))
('Finance', Decimal('85000.0000'))
('Music', Decimal('40000.0000'))
('Physics', Decimal('91000.0000'))
('History', Decimal('61000.0000'))
('Elec. Eng', Decimal('80000.0000'))


Question11:
('Comp. Sci.', Decimal('77333.3333'))
('Biology', Decimal('69000.0000'))
('Finance', Decimal('85000.0000'))
('Physics', Decimal('91000.0000'))
('History', Decimal('61000.0000'))
('Elec. Eng', Decimal('80000.0000'))


Question12:
('Srinivasan',)
('Smith',)
('Tom',)
('Wu',)
('El Said',)
('Gold',)
('Katz',)
('Califieri',)
('Singh',)
('Crick',)
('Brandt',)
('Kim',)
```

```
Question13:
('Wu',)
('Einstein',)
('Gold',)
('Katz',)
('Singh',)
('Brandt',)
('Kim',)


Question14:
('Wu',)
('Einstein',)
('Gold',)
('Katz',)
('Singh',)
('Brandt',)
('Kim',)


Question15:
('Comp. Sci.', Decimal('77333.3333'))
('Biology', Decimal('69000.0000'))
('Finance', Decimal('85000.0000'))
('Physics', Decimal('91000.0000'))
('History', Decimal('61000.0000'))
('Elec. Eng', Decimal('80000.0000'))


Question16:
('Comp. Sci.',)
('Biology',)
('Finance',)
('Physics',)


Question17:
('Srinivasan', 'CS-101')
('Srinivasan', 'CS-315')
('Srinivasan', 'CS-347')
('Wu', 'FIN-201')
('Mozart', 'MU-199')
('Einstein', 'PHY-101')
('El Said', 'HIS-351')
('Katz', 'CS-101')
('Katz', 'CS-319')
('Crick', 'BIO-101')
('Crick', 'BIO-301')
('Brandt', 'CS-190')
```

```
('Srinivasan', 'CS-101')
('Srinivasan', 'CS-315')
('Srinivasan', 'CS-347')
('Wu', 'FIN-201')
('Mozart', 'MU-199')
('Einstein', 'PHY-101')
('El Said', 'HIS-351')
('Katz', 'CS-101')
('Katz', 'CS-319')
('Crick', 'BIO-101')
('Crick', 'BIO-301')
('Brandt', 'CS-190')
('Brandt', 'CS-190')
('Brandt', 'CS-319')
('Kim', 'EE-181')


Question18:
('Srinivasan', 'CS-101')
('Srinivasan', 'CS-315')
('Srinivasan', 'CS-347')
('Smith', None)
('Tom', None)
('Wu', 'FIN-201')
('Mozart', 'MU-199')
('Einstein', 'PHY-101')
('El Said', 'HIS-351')
('Gold', None)
('Katz', 'CS-101')
('Katz', 'CS-319')
('Califieri', None)
('Singh', None)
('Crick', 'BIO-101')
('Crick', 'BIO-301')
('Brandt', 'CS-190')
('Brandt', 'CS-190')
('Brandt', 'CS-319')
('Kim', 'EE-181')
```

EXPERIMENT 8

1. **Create** a view of instructors without their salary called faculty
2. **Create** a view of department salary totals
3. **Create** a role of student
4. Give select privileges on the view faculty to the role student.
5. **Create** a new user and assign her the role of student.
6. Revoke privileges of the new user
7. Remove the role of student.
8. Give select privileges on the view faculty to the new user.
9. **Create** table teaches2 with same columns as teaches but with additional constraint that that semester is one of fall, winter, spring or summer.
10. **Create** index ID column of teaches. Compare the difference in time to obtain query results with or without index.
11. Drop the index to free up the space.

```python
import mysql.connector

conn = mysql.connector.connect(
    host='khaif@localhost',
    user='root',
    password='khaifhasan1331',
    database='adbms'
)

cursor = conn.cursor()

# Create a view of instructors without their salary called faculty
instructors_view_without_salary = """
CREATE VIEW faculty AS
SELECT ID, name, dept_name
FROM instructor
"""

cursor.execute(instructors_view_without_salary)

display_instructor_view = """
SELECT *
FROM faculty
"""

cursor.execute(display_instructor_view)

results = cursor.fetchall()

print("Question1:")
for row in results:
    print(row)
```

```python
print("\n")

# Create a view of department salary totals
department_salary_view = """
CREATE VIEW department_salary_totals AS SELECT dept_name, SUM(salary) AS total_salary FROM
instructor GROUP BY dept_name
"""

cursor.execute(department_salary_view)

display_department_view="""
SELECT * FROM department_salary_totals;
"""

cursor.execute(display_department_view)

results = cursor.fetchall()

print("Question2:")
for row in results:
    print(row)
print("\n")

# Create a role of student
role= """
CREATE ROLE 'student';
"""

cursor.execute(role)

# Give select privileges on the view faculty to the role student.
grant_select = """
GRANT SELECT ON faculty TO student;
"""
cursor.execute(grant_select)

# Create a new user and assign her the role of student.
new_role = """
CREATE USER khaif@localhost IDENTIFIED BY
'khaifhasan1331' """

cursor.execute(new_role)

grant_user = """
GRANT student TO khaif@localhost
"""

cursor.execute(grant_user)
```

```python
# Revoke privileges of the new user
revoke_user = """
REVOKE student FROM khaif@localhost
"""

cursor.execute(revoke_user)

# Remove the role of student.
remove_role = """
DROP ROLE student
"""

cursor.execute(remove_role)

# Give select privileges on the view faculty to the new user
select_user = """
GRANT SELECT ON faculty TO khaif@localhost
"""

cursor.execute(select_user)

# Create table teaches2 with same columns as teaches but with additional constraint that that
semester is one of fall, winter, spring or summer.
new_table= """
CREATE TABLE teaches2 (
  ID INT NOT NULL,
  course_id VARCHAR(255) NOT NULL,
  sec_id INT NOT NULL,
  semester VARCHAR(255) NOT NULL CHECK (semester IN ('Fall', 'Winter', 'Spring', 'Summer')),
  year INT NOT NULL,
  FOREIGN KEY (ID) REFERENCES instructor(ID)
)
"""

cursor.execute(new_table)

# Create index ID column of teaches. Compare the difference in time to obtain query results with or
without index.
create_index = """
CREATE INDEX idx_ID ON teaches (ID)
"""

cursor.execute(create_index)

# Drop the index to free up the space.
drop_index = """
DROP INDEX idx_ID ON teaches
```

```
"""
cursor.execute(drop_index)
```

```
Question1:
(10101, 'Srinivasan', 'Comp. Sci.')
(12121, 'Wu', 'Finance')
(15151, 'Mozart', 'Music')
(22222, 'Einstein', 'Physics')
(32343, 'El Said', 'History')
(33456, 'Gold', 'Physics')
(45565, 'Katz', 'Comp. Sci.')
(58583, 'Califieri', 'History')
(76543, 'Singh', 'Finance')
(76766, 'Crick', 'Biology')
(83821, 'Brandt', 'Comp. Sci.')
(98345, 'Kim', 'Elec. Eng')


Question2:
('Comp. Sci.', Decimal('232000'))
('Finance', Decimal('170000'))
('Music', Decimal('40000'))
('Physics', Decimal('182000'))
('History', Decimal('122000'))
('Biology', Decimal('72000'))
('Elec. Eng', Decimal('80000'))
```

EXPERIMENT 9

```
SQL*Plus: Release 21.0.0.0.0 - Production on Wed May 15 10:51:44 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle.  All rights reserved.

Enter user-name: system
Enter password:
Last Successful login time: Wed May 15 2024 10:29:18 +05:30

Connected to:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> create type addr_ty as object
  2  (street varchar2(60),
  3  city varchar2(30),
  4  state char(2),
  5  zip varchar(9));
  6  /

Type created.

SQL> CREATE TYPE person_ty AS OBJECT
  2   (name   varchar2(25),
  3   address  addr_ty);
  4  /

Type created.

SQL> CREATE TYPE emp_ty AS OBJECT
  2   (empt_id      varchar2(9),
  3   person  person_ty);
  4  /

Type created.

SQL> CREATE TABLE EMP_OO
  2   (full_emp emp_ty);

Table created.

SQL> insert into emp_oo values
  2  (emp_ty('100',
  3  person_ty('Ram',
  4  addr_ty('1000 TU',
  5  'Patiala', 'PB', '147001'))));
```

```
1 row created.

SQL> insert into emp_oo values
  2   (emp_ty('101',
  3    person_ty('Sham',
  4    addr_ty('1001 TU',
  5  'Patiala', 'PB', '147001'))));

1 row created.

SQL> select * from emp_oo;

FULL_EMP(EMPT_ID, PERSON(NAME, ADDRESS(STREET, CITY, STATE, ZIP)))
--------------------------------------------------------------------------
EMP_TY('100', PERSON_TY('Ram', ADDR_TY('1000 TU', 'Patiala', 'PB', '147001')))
EMP_TY('101', PERSON_TY('Sham', ADDR_TY('1001 TU', 'Patiala', 'PB', '147001')))

SQL> desc emp_oo;
 Name                              Null?   Type
 --------------------------------- ------- ---------------------------
 FULL_EMP                                  EMP_TY

SQL> select e.full_emp.empt_id ID,
  2  e.full_emp.person.name NAME,
  3  e.full_emp.person.address.city CITY
  4  from emp_oo e;

ID     NAME                 CITY
-------- ----------------------- ----------------------------
100     Ram                 Patiala
101     Sham                 Patiala

SQL> Update emp_oo e set
  2    e.full_emp.person.name='Raj'
  3    where
  4    e.full_emp.empt_id='100';

1 row updated.

SQL> select e.full_emp.empt_id ID,
  2    e.full_emp.person.name NAME,
  3    e.full_emp.person.address.city CITY
  4    from emp_oo e;

ID     NAME                 CITY
-------- ----------------------- ----------------------------
100     Raj                 Patiala
```

```
101     Sham            Patiala
```

```sql
SQL> create or replace type newemp_ty as object (firstname varchar2(25),
 2  lastname varchar2(25), birthdate date,
 3  member function AGE(birthdate in DATE) return NUMBER)
 4  /
```

Type created.

```sql
SQL> create or replace type body newemp_ty as
 2    member function AGE(BirthDate in DATE) return NUMBER is
 3    begin
 4        RETURN ROUND(SysDate - birthdate);
 5    end;
 6  end;
 7  /
```

Type body created.

```sql
SQL> create table new_emp_oo
 2  (employee newemp_ty);
```

Table created.

```sql
SQL> insert into new_emp_oo values
 2  (newemp_ty('Ram', 'Lal', '12-dec-1976'));
```

1 row created.

```sql
SQL> select e.employee.firstname, e.employee.age(e.employee.birthdate) from
 2  new_emp_oo e;
```

```
EMPLOYEE.FIRSTNAME        E.EMPLOYEE.AGE(E.EMPLOYEE.BIRTHDATE)
------------------------- ------------------------------------
Ram                          17321
```

```sql
SQL> create table new_emp1 of emp_ty;
```

Table created.

```sql
SQL> create type emp_ty1 as object
 2  (empt_id varchar2(9),
 3  person person_ty);
 4  /
```

Type created.

```sql
SQL> create table emp_oo1(full_emp emp_ty1);
```

```
Table created.

SQL> insert into emp_oo1 values
  2      (emp_ty1('101',
  3      person_ty('Sham',
  4      addr_ty('1001 TU',
  5   'Patiala', 'PB', '147001'))));

1 row created.

SQL> insert into new_emp1 values ('100', person_ty('raj', addr_ty('1000 TU', 'Pta', 'Pb', '147001')));


1 row created.

SQL> select * from new_emp1;

EMPT_ID
---------
PERSON(NAME, ADDRESS(STREET, CITY, STATE, ZIP))
-------------------------------------------------------------------------
100
PERSON_TY('raj', ADDR_TY('1000 TU', 'Pta', 'Pb', '147001'))


SQL> select ref(p) from new_emp1 p;

REF(P)
-------------------------------------------------------------------------
000028020962310E79DAD541678083F34D04C7597F4FAF0E96224F4E05993B631113268ED200
41B9
810000


SQL> create type new_dept_oo as object
  2  (depno number(3), dname varchar(20));
  3  /

Type created.

SQL> CREATE TABLE dept_table OF new_dept_oo;

Table created.

SQL> insert into dept_table values(10, 'comp');

1 row created.
```

```
SQL> insert into dept_table values(20, 'chem');

1 row created.

SQL> insert into dept_table values(10, 'math');

1 row created.

SQL> select ref(p) from dept_table p;

REF(P)
--------------------------------------------------------------------------------
0000280209E0B2B6CBC62A4509A73B0168855948CE0BD10BC5001F4AD79B080B129E78F1DF0
041B9
990000

00002802091BDD768FBC6E4197B0D94EE374114CD80BD10BC5001F4AD79B080B129E78F1DF0
041B9
990001

0000280209F5B9EBEAEDA94A45A9BF32CFD67DAE7D0BD10BC5001F4AD79B080B129E78F1DF
0041B9
990002


SQL> create table emp_test_fk(
  2  empno number(3),
  3  name varchar(10),
  4  dept ref new_dept_oo);

Table created.

SQL> desc emp_test_fk
 Name                              Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPNO                                      NUMBER(3)
 NAME                                       VARCHAR2(10)
 DEPT                                       REF OF NEW_DEPT_OO

SQL> set desc depth 2
SQL> desc emp_test_fk
 Name                              Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPNO                                      NUMBER(3)
 NAME                                       VARCHAR2(10)
 DEPT                                       REF OF NEW_DEPT_OO
  DEPNO                                     NUMBER(3)
```

```
 DNAME                        VARCHAR2(20)

SQL> insert into emp_test_fk
  2  select 100, 'raj', ref(p) from dept_table p where depno = 10;

2 rows created.

SQL> insert into emp_test_fk
  2  select 101, 'shyam', ref(p) from dept_table p where depno = 20;

1 row created.

SQL> select * from emp_test_fk;

    EMPNO NAME
---------- ----------
DEPT
--------------------------------------------------------------------------------
      100 raj
0000220208E0B2B6CBC62A4509A73B0168855948CE0BD10BC5001F4AD79B080B129E78F1DF

      100 raj
0000220208F5B9EBEAEDA94A45A9BF32CFD67DAE7D0BD10BC5001F4AD79B080B129E78F1DF

      101 shyam
00002202081BDD768FBC6E4197B0D94EE374114CD80BD10BC5001F4AD79B080B129E78F1DF


SQL> select empno, name, deref(e.dept) from emp_test_fk e;

    EMPNO NAME
---------- ----------
DEREF(E.DEPT)(DEPNO, DNAME)
--------------------------------------------------------------------------------
      100 raj
NEW_DEPT_OO(10, 'comp')

      100 raj
NEW_DEPT_OO(10, 'math')

      101 shyam
NEW_DEPT_OO(20, 'chem')


SQL> select empno, name, deref(e.dept), deref(e.dept).depno depno,
  2  deref(e.dept).dname dname from emp_test_fk e;

    EMPNO NAME
```

```
---------- ----------
DEREF(E.DEPT)(DEPNO, DNAME)
-----------------------------------------------------------------------
    DEPNO DNAME
---------- -------------------
     100 raj
NEW_DEPT_OO(10, 'comp')
      10 comp


     100 raj
NEW_DEPT_OO(10, 'math')
      10 math


    EMPNO NAME
---------- ----------
DEREF(E.DEPT)(DEPNO, DNAME)
-----------------------------------------------------------------------
    DEPNO DNAME
---------- -------------------

     101 shyam
NEW_DEPT_OO(20, 'chem')
      20 chem


SQL> create table emp_table_fk
  2  (employee emp_ty,
  3  dept ref new_dept_oo);

Table created.

SQL> set describe depth 1
SQL> desc emp_table_fk
 Name                            Null?   Type
 ---------------------------------------- ------- ---------------------------
 EMPLOYEE                                EMP_TY
 DEPT                                    REF OF NEW_DEPT_OO

SQL> set describe depth 2
SQL> desc emp_table_fk
 Name                     Null?   Type
 ---------------------------------------- ------- ---------------------------
 EMPLOYEE                                EMP_TY
  EMPT_ID                       VARCHAR2(9)
  PERSON                        PERSON_TY
 DEPT                                    REF OF NEW_DEPT_OO
  DEPNO                         NUMBER(3)
  DNAME                         VARCHAR2(20)
```

```
SQL> set describe depth 3
SQL> desc emp_table_fk
 Name                          Null?   Type
 ----------------------------------------- ------- ----------------------------
 EMPLOYEE                              EMP_TY
  EMPT_ID                       VARCHAR2(9)
  PERSON                        PERSON_TY
   NAME                         VARCHAR2(25)
   ADDRESS                      ADDR_TY
 DEPT                          REF OF NEW_DEPT_OO
  DEPNO                         NUMBER(3)
  DNAME                         VARCHAR2(20)


SQL> set describe depth 4
SQL> desc emp_table_fk
 Name                          Null?   Type
 ----------------------------------------- ------- ----------------------------
 EMPLOYEE                              EMP_TY
  EMPT_ID                        VARCHAR2(9)
  PERSON                         PERSON_TY
   NAME                          VARCHAR2(25)
   ADDRESS                       ADDR_TY
    STREET                       VARCHAR2(60)
    CITY                       VARCHAR2(30)
    STATE                      CHAR(2)
    ZIP                      VARCHAR2(9)
 DEPT                            REF OF NEW_DEPT_OO
  DEPNO                          NUMBER(3)
  DNAME                          VARCHAR2(20)


SQL> INSERT INTO emp_table_fk
 2  VALUES (
 3    emp_ty(
 4       100,
 5       person_ty('ram', addr_ty('10 tu', 'pat', 'pb', '147001'))
 6    ),
 7    (SELECT REF(P)
 8     FROM dept_table P
 9     WHERE depno = 10
 10     AND ROWNUM = 1)
 11  );

1 row created.

SQL> select * from emp_table_fk;

EMPLOYEE(EMPT_ID, PERSON(NAME, ADDRESS(STREET, CITY, STATE, ZIP)))
```

```
-------------------------------------------------------------------------
DEPT
-------------------------------------------------------------------------
EMP_TY('100', PERSON_TY('ram', ADDR_TY('10 tu', 'pat', 'pb', '147001')))
0000220208E0B2B6CBC62A4509A73B0168855948CE0BD10BC5001F4AD79B080B129E78F1DF


SQL> select e.employee.empt_id id, e.employee.person.name name,
  2  deref(e.dept), deref(e.dept).depno depno,
  3  deref(e.dept).dname dname from emp_table_fk e;

ID        NAME
--------- ------------------------
DEREF(E.DEPT)(DEPNO, DNAME)
-------------------------------------------------------------------------
    DEPNO DNAME
---------- -------------------
100       ram
NEW_DEPT_OO(10, 'comp')
      10 comp    print(row)
```