BSCCS2001: Practice Assignment with Solutions
Week 2

Modules covered:

1. Attribute Types, Relation Schema and Instance, Keys, Relational Query Languages
2. Operations, Select, Project, Union, Difference, Intersection, Cartesian Product
3. Natural Join, Aggregate Operations
4. Introduction to SQL - History of SQL, Data Definition Language (DDL), Basic Query Structure (DML)
5. Additional Basic Operations, Set Operations, Null Values, Aggregate Functions

1. Consider the table **taskAssignment** (in Figure 1) which represents tasks assigned to each employee for a given day. [MCQ: 2 points]

taskAssignment

employee_num	task_num	task_duration	date_of_assignment	supervisor_num	location
101	P103	7	10-01-2020	112	Block-C
102	P103	5	10-01-2020	112	Block-C
101	P103	4	11-01-2020	112	Block-C
101	P103	6	12-01-2020	112	Block-C
104	P102	6	10-01-2020	111	Block-B
105	P101	7	10-01-2020	110	Block-A
104	P101	7	11-01-2020	110	Block-A
105	P101	6	11-01-2020	110	Block-A
102	P102	6	11-01-2020	111	Block-B

Figure 1: Table **taskAssignment**

Select the appropriate compound key for the table.

- ◯ { *employee_num* }
- ◯ { *employee_num*, *task_num* }
- √ { *employee_num*, *task_num*, *date_of_assignment* }
- ◯ { *employee_num*, *supervisor_num* }

Solution: A compound key is a set of more than one attribute which uniquely identifies all rows of a relation.
The entries for { *employee_num* } are not unique for all the tuples. It is neither a set of more than one attribute nor uniquely identifies all rows of the given relation. Thus, it cannot be a compound key.
The entries for { *employee_num*, *task_num* } are not unique for all the tuples. Thus, it cannot be a compound key.
The entries for {*employee_num*, *task_num*, *date_of_assignment*} are unique for all the rows. So, **it is a valid candidate key for the given relation**.
The entries for { *employee_num*, *supervisor_num* } are not unique for all the tuples. Thus, it cannot be a compound key.

2. Consider the following relational schema on students of a school. [MCQ: 2 points]
studentInfo *(enrollment_num, class, section, roll, name).*
{*enrollment_num*} and {*class, section, roll*} are two possible candidate keys. What is the maximum number of possible superkeys of **studentInfo**?

- ◯ 16
- √ 18
- ◯ 20
- ◯ 22

Solution: This question will be discussed in the *Solve with the Instructor* session.

3. Consider the tables **vendor** and **component** as shown in Figure 2. The table **component** has attribute *vendor_num* which is a foreign key that refers to table **vendor(*vendor_num*)**.

vendor

vendor_num	vendor_name	vendor_location
10	YADAV	CHENNAI
11	AKHTAR	KOLKATA
12	PRASAD	TRICHY
13	SHARMA	BENGALURU

component

item_num	name	cost	vendor_num
1011	RAM	2500.00	11
1012	CPU	8000.50	12
1013	MONITOR	5000.00	10
1014	KEYBOARD	500.50	13
1013	MONITOR	2250.00	13
1014	KEYBOARD	450.50	11
1011	RAM	3300.00	10

Figure 2: Tables **vendor** and **component**

Identify the appropriate "CREATE TABLE" statement for table **component**. [MCQ: 2 points]

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, name),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, vendor_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(item_num, vendor_num));
```

√
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
```

```
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, vendor_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

Solution:
Option 1: since the table **component** has duplicate values in the column *item_num*, the attribute set {*item_num*} cannot be a primary key. Hence, option 1 is incorrect.
Option 2: since table **component** has duplicate values in columns {*item_num, name*}, the attribute set {*item_num, name*} cannot be a primary key. Hence, option 2 is incorrect.
Option 3: since the number of columns in the foreign key does not match the number of columns in the referenced table, option 3 is incorrect.
Option 4: since table **component** has unique entries in columns {*item_num, vendor_num*}, attribute set {*item_num, vendor_num*} is the proper primary key and hence, option 4 is the correct option.

4. Identify the appropriate "ALTER TABLE" statement for table **vendor** such that we can add a column named *vendor_phone* to store the phone number of the vendors. [MSQ: 2 points]

◯ `ALTER TABLE vendor APPEND COLUMN vendor_phone numeric(10);`

√ `ALTER TABLE vendor ADD COLUMN vendor_phone numeric(10);`

√ `ALTER TABLE vendor ADD vendor_phone numeric(10);`

◯ `ALTER TABLE vendor INSERT COLUMN vendor_phone numeric(10);`

Solution: To add a new column in the given table, we use **"ADD"** or **"ADD COLUMN"** commands. Thus, option 2 and option 3 are correct.

5. Identify the correct statement(s) about a Foreign Key. [MSQ: 2 points]

- ◯ A FOREIGN KEY is a data element/attribute within a data field of a data record that is not unique, and cannot be used to distinguish one data record in a database from another data record within a database table.
- ◯ A FOREIGN KEY constraint prevents data from being inserted into the foreign key column.
- √ A FOREIGN KEY is a data element/attribute within a data field of a data record within a database table that refers to either a primary key or an attribute with unique constraint.
- √ A FOREIGN KEY can be added at the time of ALTER TABLE query.

Solution: Option 1 - A FOREIGN KEY is a data element/attribute within a data field of a data record that is unique, and this can be used to distinguish one data record in a database from another data record within a database table.
Option 2 - The FOREIGN KEY constraint prevents invalid data from being inserted into the foreign key column.
Option 3, 4 are factual statements about Foreign Key.

6. Consider the table **Employee** given in Figure 3.

Employee

ID	Name	Salary
1	MIKE	35
2	KYLE	50
3	JAMES	50
4	JONES	NULL
5	LIMA	70
6	PACY	50

Figure 3: Table Employee

Determine the correct relation based on the three queries given below: [MCQ: 2 points]

- `SELECT COUNT(DISTINCT Salary) as A FROM Employee;`
- `SELECT COUNT(*) as B FROM Employee;`
- `SELECT COUNT(Salary) as C FROM Employee;`

◯ value of B>A and B=C

◯ value of A>B and B>C

◯ value of A>B and B=C

√ value of B>A and B>C

Solution: This question will be discussed in the *Solve with the Instructor* session.

7. Consider the table **Employee** given in Figure 4.

Employee

ID	Name	Salary
1	MIKE	35
2	KYLE	50
3	JAMES	50
4	JONES	NULL
5	LIMA	70
6	PACY	50

Figure 4: Table Employee

Determine the suitable query that returns the following table: [MCQ: 2 points]

AVG_SAL
55

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND ID>5;
```

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND Salary>50;
```

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND ID>3;
```

√
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND Salary>35;
```

Solution: From the table **Employee**, NOT NULL values that are greater than 35 are 50, 50, 70, 50. The average of 50, 50, 70, 50 is 55.

8. Identify the output for the following SQL statement. [MCQ: 2 points]

```
SELECT max(temperature) - min(temperature)
FROM weatherReport
WHERE state='Karnataka';
```

- ◯ 4
- √ 5
- ◯ 2
- ◯ 0

Solution: The part of the statement:

```
WHERE state='Karnataka';
```

extracts all rows having *state* as "Karnataka".
`max(temperature)` returns the maximum *temperature* from among the given cities in "Karnataka", i.e. 36 (for "Bellary").
`min(temperature)` returns the minimum *temperature* from among the given cities in "Karnataka", i.e. 31 (for "Bengaluru").
Therefore, the output is 36 - 31 = 5;

9. Using the table **Citizen** given in Figure 5, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 5: Table Citizen

Identify the correct SQL statement(s) to create the given table. [MSQ: 2 points]

◯
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID));
```

√
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID, lastname));
```

√
```
CREATE TABLE Citizen (ID int,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (firstname, lastname));
```

√
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID, profession));
```

◯
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (profession));
```

Solution: Only an attribute, or a set of attributes that can uniquely identify a row, can be chosen as the PRIMARY KEY. Here (ID, lastname), (firstname, lastname) and (ID, profession) can do so. Hence, options 2, 3, 4 are correct.

10. Using the table **Citizen** given in Figure 6, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 6: Table Citizen

Let the table **Citizen** be created using the SQL statement given below: [MSQ: 2 points]

- `CREATE TABLE Citizen (ID int, profession varchar(255), lastname varchar(255) NOT NULL, firstname varchar(255));`

Identify the correct INSERT INTO statement for this table.

√ `INSERT INTO Citizen (ID, profession, lastname, firstname) VALUES (23,'clerk', 'Holmes', 'Mark');`

◯ `INSERT INTO Citizen TABLE VALUES (23,'clerk', 'Holmes', 'Mark');`

√ `INSERT INTO Citizen VALUES (23,'clerk', 'Holmes', 'Mark');`

◯ `INSERT INTO Citizen VALUES ('23','clerk', 'Holmes', 'Mark');`

Solution: To insert values in a table, the generic format is -
INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);
OR
INSERT INTO table_name VALUES (value1, value2, ...);
Also, varchar type attributes need to be in ' ' and INT type attributes without ' '.

11. Using the table **Citizen** given in Figure 7, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 7: Table Citizen

Let the table **Citizen** be created with *firstname* as the primary key, and the values be inserted as per the schema in the table. [MCQ: 1 points]

Choose the SQL statement to remove the primary key constraint Citizen_pkey of the table **Citizen**.

◯ `ALTER TABLE Citizen`
`DROP Citizen CONSTRAINT Citizen_pkey;`

◯ `MODIFY TABLE Citizen`
`DROP CONSTRAINT Citizen_pkey;`

◯ `MODIFY TABLE Citizen`
`DROP Citizen CONSTRAINT;`

√ `ALTER TABLE Citizen`
`DROP CONSTRAINT Citizen_pkey;`

Solution: The syntax to remove an existing primary key constraint is -
ALTER TABLE table_name DROP CONSTRAINT primary_key_constraint_name;
Hence option 4 is correct.
ALTER TABLE Citizen DROP CONSTRAINT Citizen_pkey;

12. Let $\{sup\_num\}$ be the primary key of table **suppliers** and $\{part\_num, sup\_num\}$ be the primary key of table **parts**. [NAT: 2 points]
Consider the SQL query given below:

```
SELECT s.sup_num, sum(p.part_qty)
FROM suppliers s, parts p WHERE s.sup_num = p.sup_num
GROUP BY s.sup_num
HAVING SUM(p.part_qty) > 70
```

How many rows will be returned by the above SQL query?
Answer: 2

Solution: As per the given SQL statement, it first performs a Cartesian product between **suppliers** and **parts**, which output all possible combinations from both the tables.
The part of the statement:
`s.sup_num = p.sup_num`
eliminates the rows which do not satisfy the condition. The output is as shown below:

s.sup_num	s.sup_name	p.part_num	p.sup_num	p.part_qty
1001	Able	301	1001	32
1001	Able	302	1001	16
1002	Peter	301	1002	41
1002	Peter	302	1002	11
1002	Peter	304	1002	35
1003	Molina	302	1003	36
1003	Molina	304	1003	40
1004	Nikki	301	1004	17
1004	Nikki	303	1004	25

The part of the statement:
`SELECT s.sup_num, sum(p.part_qty) ...GROUP BY s.sup_num`
results in:

s.sup_num	SUM(p.part_qty)
1001	48
1002	87
1003	76
1004	42

Finally, the part of the statement: `HAVING SUM(p.part_qty) > 70`
results in:

s.sup_num	SUM(p.part_qty)
1002	87
1003	76

Thus, the result has 2 rows.