# **Question 1**

## a)

Precision is total number of digit in datatype number and scale is the number of digit to the right of decimal point.

### Example

In 45.7, the precision is 3 and scale is 1.

## b)

For column postcode as I see that the size of postcode is fixed to 7 character so using CHAR datatype is more appreciated. If we do not know the exact size of string then we can use VARCHAR2. Now in my case, size of postcode is fixed and it will take 7 byte.

## c)

The constraint will check that whether the type is detached, terraced or semi.

## d)

First statement will add new column to table house.

Second statement is not correct because “table” word cannot be in update statement. So it will give syntax error.

## e)

INSERT INTO houses

(address\_line\_1, postcode, type, cust\_id, sale\_price, sale\_date)

VALUES

('101 ABC Street', 'M54 5DE', 'Semi', 54859, 58500.00, '04-May-2020');

## f)

Column: sale\_price

Create INDEX idx\_price on houses (sale\_price);

## g)

Hash index directly search the data from table without any indexing. It index the data using shorter hashed key instead of using original value which results in fast performance of query. If we have huge data in database, then searching all index value will take a lot of time.

# **Question 2**

## a)

Select dep\_name, city, country

FROM departments

INNER JOIN locations

ON departments.location\_id = locations.location\_id;

## b1)

The given query will return all the rows such as (first\_name, last\_name and dep\_name) from both tables instead of returning (dep\_name) which is associated to that specific employee because there is no join in query.

Here is the right query:

Select first\_name, last\_name, dep\_name

FROM employees

INNER JOIN departments

ON employees.dep\_id = departments.dep\_id;

## B2)

The query will return syntax error because ‘where’ clause always come after ‘from’ clause.

Therefore, the correct query is:

Select first\_name, last\_name, dep\_id

FROM employees

Where dep\_id=5;

## B3)

The following query will return those departments whose name has letter “M” in start, between or end. It will not return those departments whose name starts with “M”.

Therefore, the correct query is:

Select dep\_id, dep\_name

FROM departments

Where dep\_name LIKE ‘M%’;

## B4)

The given query return all the cities with total number of department whether it is less than 3 or greater.

Select city, count(\*)

FROM departments

inner join locations using(location\_id)

group by city

having count(\*)>3;

## B5)

The given query do not return manager based on all department if that manager is associated with more than one department.

Here is the correct query:

Select first\_name, last\_name,dep\_name

FROM departments

inner join employees

ON employees.employee\_id = departments.manager\_id;

## B6)

The query will return syntax error.

The correct query is:

select dep\_name,sum(salary)

FROM departments

inner join employees using(dep\_id)

group by dep\_name

UNION

SELECT 'Grand Total', sum(salary)

FROM employees;

# **Question 3**

## a)

studentID attribute is a Surrogate key because it is a single column primary key with numeric datatype.

## b)

Demoralization end up with fewer table because in normalization we remove redundant columns from table and create new table. In this way, number of tables are increasing and we need more joins to execute a query. On the other hand, in demoralization we are joining tables and adding redundant columns, which results decrease in number of tables.

## c)

The relationship between CourseType and Course is one to many, means one CourseType can have several Courses But one Course cannot belong to many CourseType. It can only have one CourseType.

## d)

unit\_grade attribute is not useful in Student because if a student register the data of student stored in Student table. At that point we do not have any grade of that student so what we store in unit\_grade column it will be empty until we do not have any grade. That is why unit\_grade is not useful in Student entity because of redundancy.

## e)

Foreign keys

## f)

Associative entity

It is preferable to a direct relationship between student and course because both student and course has many to many relationship

## g)

It will be impossible to drop Unit table because both Unit and UnitEnrollement has one to many relationship. There is a foreign of Unit Table in UnitEnrollement Table.

# **Question 4**

## a)

### Select:

It is used to select all the rows from table under given logic. It eliminates rows of the table.

### Project:

It is used to select few columns from table rather than select all. It eliminates columns of the table

## b)

Answer: 1.

## d)

Plan B will run fast because it is not using index scan technique whereas Plan A using index scan, which will take time to execute the query. In this way, performance of the query can also be compromised. Furthermore, Plan A will first retrieve all the data then at the end after joins it will again filter the data whereas Plan B knows what to retrieve at first 2 steps so it will only retrieve that data and execute faster.

## e)

Sequential scan is faster in this case because index scan first need to find the index then retrieve the data that requires several input/output operation so it will take time to process.

# **Question 5**

## a)

A database will be in consistent state if all the data integrity constraints such as (foreign, not null, unique, check etc.) are satisfied.

## b)

It is a shared database among tenant and data storage is isolated logically and physical on per tenant. One tenant cannot access the data of other tenant.

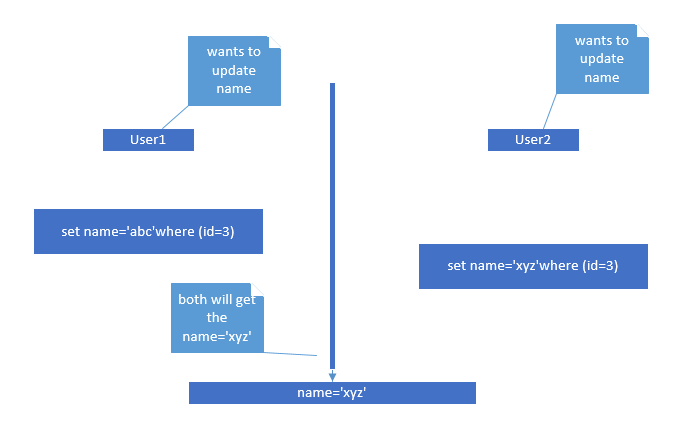
## c)

It access and modifies the content of database in order to maintain ACID properties. It access data using write and read operation and maintains the following properties such as atomicity, consistency, isolation and durability.

Atomicity ensure that any modification happen or not, consistency ensure that database must be consistent before and after modification, isolation ensure that multiple modification/transaction occur independently and durability ensure that modification of database occurs successfully.

## d)

When two different transactional activities occur on a same column of same row at same time. In this case, the first transaction will overwritten by the second. This results in lost update problem.



## e)

Granularity lock means that how much of the data is locked at one time. It range from small to large data item such as database, file, record or field.

### Example 1:

Hierarchy: DB->Area->File->Record

If transacrtion1 lock a file and transacrtion2 wants to acquire record. Transacrtion2 will have to wait because record is locked by transacrtion1.

### Example 2:

If transacrtion1 lock a column and transacrtion2 want to write in another column of same row. Transacrtion2 have to wait.

## f)

Deadlock can happen in database by two or more transactions when each transaction is waiting for another transaction.

### Wait Die:

If transaction1 is greater than transaction2, transacrtion1 have to wait. Otherwise, if transacrtion1 is smaller than transacrtion2, transacrtion1 have to abort and restart later.

### Wound wait:

If transacrtion1 greater than transacrtion2, transacrtion2 have to abort and restart later. Otherwise, if transacrtion1 smaller than transacrtion2, transacrtion2 has to wait.