# IST769 Homework 9 Submission

## Basic Information

Your Name: Srihari Busam   
Your SUID: sbusam  
Your Email: sbusam@syr.edu  
Date Due: 12/02/2021  
Homework #: 9

## QUESTIONS:

1. Design your own scenario for which a Cassandra table would be a good solution. Make sure to explain the scenario and the specific characteristics of the scenario which would make Cassandra a good fit. Make sure to follow a query first approach and justify how the partition and cluster keys should be setup.

**ANSWER**

I ma thinking of using Cassandra for wish lists for shopping similar to what Amazon ecommerce provides. The idea would be to create custom wish lists based on the categories that customer wants to choose.

I choose Cassandra as Availability is key for the scenario. Eventual consistency is okay as long as the writes to the wish lists do not fail. Generally customer do not care much about the other customer wish lists, eventual consistency is fine and fits right into Cassandra usage in my view.

For the datatypes I choose the customer\_id and list\_id as the primary key as the name of the customer or the list name may confict with users of similar name. For items I choose the “set” data type as we do not expect any duplicates in the set.

The following are the key requirements:

1. Able to add the or update the list. Need availability over consistency
2. Ability get the list by customer.
3. For analytics, ability to get the list of wish lists which contains a specific product.
4. For analytics, ability to fetch all the wish lists for a specific day.

1. Create your Cassandra table in CQL based on your scenario from the previous exercise. You should define the columns and data types to suit your scenario in addition to configuring the partition and cluster keys.

**ANSWER**

|  |
| --- |
| CQL - commands |
| CREATE KEYSPACE shopping with replication = { 'class': 'SimpleStrategy', 'replication\_factor':'3'};  DESCRIBE keyspaces;  USE shopping;  CREATE TABLE shopping.wish\_list ( customer\_id int, list\_id int, created\_on timestamp, cusomter\_name text, wish\_list\_name text, items set<text>, PRIMARY KEY ( customer\_id, list\_id ));  describe TABLE wish\_list; |

Screenshot:

Text

Description automatically generated

1. Write CQL statements to add data to your table. Add at least 9 records consisting of 3 different partition and cluster keys.

**ANSWER**

|  |
| --- |
| Cql commands |
| INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (1,1,'2021-01-01', 'Sam', 'groceries list', {'Milk', 'Bread', 'Coffe'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (1,2,'2021-01-02', 'Sam', 'Electronics', {'TV', 'Play Station', 'Laptop'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (1,3,'2021-01-01', 'Sam', 'Books', {'Dune', 'Titanic'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (2,1,'2021-01-01', 'George', 'groceries list', {'Potato', 'Bread', 'Tea'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (2,2,'2021-01-02', 'George', 'Electronics', {'Boom Box', 'XBOX'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (2,3,'2021-01-06', 'George', 'Books', {'Dune', 'Learn Javascript'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (3,1,'2021-01-04', 'Tina', 'groceries list', {'Cheese'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (3,2,'2021-01-01', 'Tina', 'Electronics', {'XBOX'});  INSERT INTO wish\_list (customer\_id, list\_id, created\_on, cusomter\_name, wish\_list\_name, items) VALUES (3,3,'2021-01-02', 'Tina', 'Books', {'Learn Java', 'Learn Spanish'});  SELECT \* FROM wish\_list; |

Screenshot:

Graphical user interface

Description automatically generated

1. Write a CQL statement to create an index or materialized view on your table so that you can set a different partition key to prevent ALLOW FILTERING. Then write a CQL SELECT statement to demonstrate it works as designed.

**ANSWER**

I choose to create an index for the scenario. The index will be created on the items set columns to quickly find the relevant shopping lists that contains the product.

|  |
| --- |
| CQL commands |
| CREATE INDEX ix\_shopping\_wish\_list\_items ON shopping.wish\_list ( items );  SELECT \* FROM shopping.wish\_list WHERE items CONTAINS 'XBOX'; |

Screenshot:

A screenshot of a computer

Description automatically generated with medium confidence

1. Write a CQL statement to create an index or materialized view on your table so that you can set a different cluster key to prevent ALLOW FILTERING. Then write a CQL SELECT statement to demonstrate it works as designed.

**ANSWER**

I choose to create a materialized view for the scenario to avoid “allow filtering” approach. The new view will contain the created\_on as one of the cluster key. This will help with queries to find wish lists by date.

|  |
| --- |
|  |
| DROP MATERIALIZED VIEW IF EXISTS vw\_shopping\_wish\_list;  CREATE MATERIALIZED VIEW vw\_shopping\_wish\_list  AS SELECT \* FROM shopping.wish\_list WHERE customer\_id IS NOT NULL AND list\_id IS NOT NULL AND created\_on IS NOT NULL AND cusomter\_name IS NOT NULL AND wish\_list\_name IS NOT NULL AND items IS NOT NULL PRIMARY KEY (customer\_id, created\_on, list\_id);  SELECT \* FROM vw\_shopping\_wish\_list WHERE customer\_id = 1 AND created\_on = '2021-01-01' |

Screenshot:

Creation of view:

Text

Description automatically generated

Screenshot: query

A screenshot of a computer

Description automatically generated with medium confidence