Interview Date and Time: Friday May 22nd at 9:15am

Interviewer(s):

Rita Edgel (hiring manager),

Mirza Ahmed (Technical Lead)

Alireza Heshmati-Ravazi (QA Manager).

Interview Location: Toll-free number: 1-833-799-0365; Conference ID: 277646150

**Interview questions:**

**-how you test on data mapping**

Data mapping testing can be done by running scripts/queries written base on mapping document. All columns, lookup logics and transformation logics should be covered.

-what are the processes you will follow for checking up data from source to target

Preparations:

1. review mapping document, including source and target structures, transformation notes. Make sure no non-testable, non-reasonable logics or should be clarified by BA. Logics should be clear.
2. Write test scripts (sql/hql). Test scripts should have no syntax errors (dry run query to fix syntax issues). All columns in both source and target side should be included. All look up and transformation logics should be covered. Part of preparation.

Sanity test:

1. tables’ creation and metadata validation. Should match with data model document or mapping document. One-time validation unless there is ddl changes.
2. connection validation and data readiness validation

data mapping ETL testing:

1. Record count validation
2. Data validation by running queries. All transformation logic, lookup logic, hard cod logic etc should be covered and validated.
3. Natural key validation – no null combinations and no duplications
4. Mandatory fields validation
5. Negative testing based on requirement

-how do you test using data model

Data Model usually is architect's output which is somehow like building a plan to set the relationship between data items or tables. Usually, data model is input for BA to write data mapping document.

For QA, mapping document is our major document. But we do, sometimes reference to data model to get clear understanding of relations between tables, primary keys and foreign keys and stored procedures. It helps data validation testing. Also helps us when need to mockup data to make sure data consistency and default values between tables ensuring test data good quality.

**API related questions**

<https://www.restapitutorial.com/lessons/httpmethods.html>

API Testing is to test the Application Programming Interface, or the business logical layer of any application is called API testing.

**API Testing types:**

Functionality testing.

Performance testing

Security testing

**API testing tools:**

POSTMAN: API functionality testing tool.

SOAP-UI: payed tool, provides flexibility of API testing.

REST-Assured:

JMETER: load/performance testing

**http methods**

HTTP methods are POST, GET, PUT, PATCH, and DELETE. These correspond to create, read, update, and delete (or CRUD) operations, respectively.

OPTIONS and HEAD

GET – Retrieve information about the REST API resource

POST – Create a REST API resource

PUT – Update a REST API resource

DELETE – Delete a REST API resource or related component

|  |  |  |  |
| --- | --- | --- | --- |
| **HTTP Verb** | **CRUD** | **Entire Collection (e.g. /customers)** | **Specific Item (e.g. /customers/{id})** |
| POST | Create | 201 (Created), 'Location' header with link to /customers/{id} containing new ID. | 404 (Not Found), 409 (Conflict) if resource already exists. |
| GET | Read | 200 (OK), list of customers. Use pagination, sorting and filtering to navigate big lists. | 200 (OK), single customer. 404 (Not Found), if ID not found or invalid. |
| PUT | Update/Replace | 405 (Method Not Allowed), unless you want to update/replace every resource in the entire collection. | 200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid. |
| PATCH | Update/Modify | 405 (Method Not Allowed), unless you want to modify the collection itself. | 200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid. |
| DELETE | Delete | 405 (Method Not Allowed), unless you want to delete the whole collection—not often desirable. | 200 (OK). 404 (Not Found), if ID not found or invalid. |

**Web services API types:**

SOPA – Simple Object Access Protocol

XML

JSON

REST – Representational State Transfer

|  |  |
| --- | --- |
| SOAP | REST-FUL |
| * Simple Object Access Protocol * SOAP is a protocol * All data passed in SOAP is in XML format * SOAP cannot make sure of REST since SOAP is a protocol and REST is an architectural pattern. * SOAP is a protocol. SOAP was designed with a specification. It includes a WSDL file which has the required information on what the web service does in addition to the location of the web service | * Representational State Transfer * REST is an Architectural type * REST permits different data format such as Plain text, HTML, XML, JSON etc. But the most preferred format for transferring data is JSON. * REST can make use of SOAP as the underlying protocol for web services, because in the end it is just an architectural pattern. * REST is an Architectural style in which an web service can only be treated as a RESTful service. * 70% of all web services are implemented using RESTful |

**API Testing preparations:**

Requirement document

Test scope

Test tools

**-MongoDB overview**

NoSQL database is used for storage and retrieval of data that is modeled other than the tabular relations used in relational databases (like SQL, Oracle, etc.).

MongoDB is Document Oriented database.

|  |  |
| --- | --- |
| **MongoDB** | **SQL Databases** |
| * highly flexible and scalable document structure * flexible structure which can be easily modified and extended. * faster as compared to SQL databases due to efficient indexing and storage techniques. * MongoDB does not support foreign key constraints. We can achieve this concept by embedding one document inside another. | * store data in form of tables, rows, columns and records * not very much flexible |

**MongoDB namespace:** Namespace is the concatenation of the database name and collection name. For e.g. school.students with school as the database and students as the collection

MongoDB uses **BSON** to represent document structures.

**A covered query is the one in which:**

* fields used in the query are part of an index used in the query, and
* the fields returned in the results are in the same index

**ETL**

**-what kind of testing will you perform source to target**

Sanity testing: test environment, connections, table creation, test data readiness

Functional testing: record count, data validation, negative testing( wrong source file format, wrong data type, empty file etc.), day0, day1, day2

Regression testing, end-to-end testing.