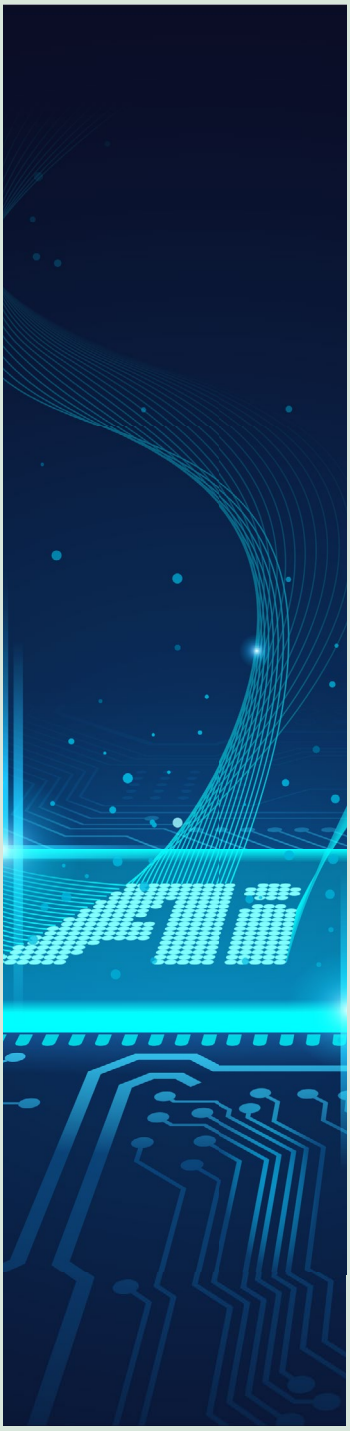
Infosys topaz

WHITE PAPER



IMPLEMENTING GENERATIVE AI FOR DATABASE MIGRATIONS

Abstract

With the advances in technology, open-source databases provide multiple advantages over maintaining traditional legacy databases. However, the process of migration has its challenges due to differences in syntax, data types, and procedural languages and compatibility issues between databases. This is addressed in Infosys Database Migration (IDM) solution using Generative AI chatbot add-on to provide a more accurate last-mile code conversion, especially when traditional utilities like Ora2PG and SQLines face inherent limitations.

Infosys®
Navigate your next

Database migration is the process of moving data, definitions, and stored procedures from one database platform to another and making relevant application changes to align with the new underlying database. Moving data involves selecting, preparing, extracting, transforming, and finally migrating the source database to the target database.

Enterprises are increasingly making the move to open-source databases like PostgreSQL and MySQL for multiple reasons, including cost, efficiency, and maintainability. Detailed advantages are summarized here for reference.

Database migration has multiple phases, including picking the right schema, performing compatibility checks, converting incompatible objects, migrating database and data, and finally, conducting post-migration validation followed by functional and performance testing for the application with a new database.

Commonly used open-source utilities like Ora2pg and SQLines (as an example) are leveraged for traditional database migrations. They are full of in-built features and offer genuine accelerators for migrating databases from traditional databases like Oracle, DB2, MySQL and SQL server, etc.

The below illustration discusses Oracle to Postgres database migration using a sample open-source Ora2PG utility.

The main technical consideration and focus in any migration is the code conversion and reproducing of the underlying PL/SQL code, specifically for Oracle. However, the migration process comes with its challenges.

For instance, Oracle and PostgreSQL databases have differences in syntax, data types, and procedural languages, which can complicate the migration of complex database structures and application code. Additionally, compatibility issues may arise when converting equivalents.

This involves mapping data types, constraints, and indices between the two systems. The key phases are:

1. Deploying and preparing resources:
   - Deploying and configuring Docker images
2. Converting the schema (with Ora2Pg or similar):
   - Converting the source schema
   - Rebuilding the target schema
   - Reviewing and revising the target schema
3. Continuously migrating the data:
   - Migrating data in a consistent fashion
   - Choosing between one-time migration, CDC (Change Data Capture), and transformation
4. Validating the database migration:
   - Ensuring objects and data are translated correctly in the target
5. Cutting over to use PostgreSQL:

- Switching the application to use PostgreSQL

There are 3 approaches currently being leveraged to address these concerns:

1. Traditional ANTLR-based custom scripts developed internally by organizations
2. Traditional open-source utilities for different DB migrations, such as Ora2PG and SQLines
3. Generative AI-based schema migrations: Latest solution leveraging single service integration with ChatGPT to potentially address multiple databases using OpenAI curated queries

In addition, extension wrappers are developed on top of open-source **Ora2PG/SQLines** and Generative AI to increase coverage from ~60% to ~85% in database migration.

While the first two approaches are more amenable for simple to medium DB structural elements, procedures, functions, triggers, and package changes, their coverage is low for Complex SQL statements such as Merge, Bulk Collect, Pivot, Unpivot, and DB vendor-specific packages.

The challenge in traditional database migration approaches (with **ANTLR**-based custom-built utilities or open-source utilities) is the need for multiple expert interventions, maintainability, and reliability in real world deployments when coverage exceeds 60%.

Hence the need to improve this using Generative AI.

Generative AI Queries: How do they work and What is their promise

As we have outlined, the biggest challenge in database migration is reproducing accurately the underlying code, the procedural language used to embrace SQL statements and its syntax.

The IDM (Infosys Database Migration) solution leverages all the approaches mentioned previously:

1. Traditional open-source utilities for different DB migrations, e.g., Ora2PG and SQLines with wrappers as required for various specific tasks
2. Generative AI-based schema migrations: Utilizing single service integration with ChatGPT to address multiple databases using OpenAI curated queries

The USP for IDM (Infosys Database Migration) solution is its leverage of a Generative AI chatbot to provide a more accurate last-mile code conversion, especially when traditional Ora2PG encounters limitations. Additionally, it is enabled with enhanced Apache NiFi processors and can handle complex and high-volume data migrations with nearly 100% coverage.

Complex structural changes can be taken care of by leveraging Generative AI integration services. The only change or customization needed is the corresponding relevant prompt to cater to migration across diverse databases.

This leverages OpenAI's ChatGPT open Azure platform. IDM integrates through API calls to the ChatGPT connector to enable querying and display the resultant PostgreSQL equivalent query in IDM UI.

IDM schema conversion with Generative AI Solution

Source -> Oracle; Target -> PostgreSQL; Install -> Azure OpenAI

Step 1: Use Ora2PG for conversions

Step 2: Use GetAI Query Converter (for selected outliers)

Sample query and Generative AI-based responses:

INPUT	OUTPUT
MERGE INTO employees e USING hr_records h ON (e.id = h.emp_id) WHEN MATCHED THEN UPDATE SET e.address = h.address WHEN NOT MATCHED THEN INSERT (id, address) VALUES (h.emp_id, h.address);	In PostgreSQL, the equivalent statement would be: WITH upsert AS (UPDATE employees e SET address = h.address FROM hr_records h WHERE e.id = h.emp_id RETURNING *) INSERT INTO employees (id, address) SELECT emp_id, address FROM hr_records WHERE (emp_id, address) NOT IN (SELECT id, address FROM upsert);
Oracle unpivot statement: select * from unpivot_test unpivot include nulls (quantity for product_code in (product_code_a as 'A', product_code_b as 'B', product_code_c as 'C', product_code_d as 'D'));	In PostgreSQL, the unpivot operation can be achieved using a combination of UNION ALL and CROSS JOIN. Here's how you can convert the given Oracle unpivot statement to PostgreSQL: ```sql SELECT product_code_a AS product_code, quantity_a AS quantity FROM unpivot_test UNION ALL SELECT product_code_b, quantity_b FROM unpivot_test UNION ALL SELECT product_code_c, quantity_c FROM unpivot_test UNION ALL SELECT product_code_d, quantity_d FROM unpivot_test) ``` This query first selects the product_code and the corresponding quantity for each product_code_a, product_code_b, product_code_c, and product_code_d columns individually. The UNION ALL operator combines the results of these four subqueries. Finally, the WHERE clause filters out any NULL product_code values. Note: PostgreSQL does not have a direct equivalent for the Oracle UNPIVOT statement, so we must use this workaround.
CREATE PROCEDURE remove_emp (employee_id NUMBER) AS tot_emps NUMBER; BEGIN DELETE FROM employees WHERE employees.employee_id = remove_emp.employee_id; tot_emps := tot_emps - 1; END;	In PostgreSQL, you can convert the Oracle procedure to a PostgreSQL function. Here is the equivalent code in PostgreSQL: CREATE OR REPLACE FUNCTION remove_emp (employee_id INTEGER) RETURNS VOID AS $BODY$ DECLARE tot_emps INTEGER; BEGIN DELETE FROM employees WHERE employees.employee_id = remove_emp.employee_id; tot_emps := tot_emps - 1; END; $BODY$ LANGUAGE plpgsql; Note that the data type for the employee_id parameter has been changed to INTEGER to match PostgreSQL's data type. Also, the SEMICOLON at the end of the DELETE statement is removed, as PostgreSQL does not require it.
CREATE FUNCTION get_bal(acc_no IN NUMBER) RETURN NUMBER IS acc_bal NUMBER(11,2); BEGIN SELECT order_total INTO acc_bal FROM orders WHERE customer_id = acc_no; RETURN(acc_bal); END;	CREATE OR REPLACE FUNCTION get_bal(acc_no INTEGER) RETURNS NUMERIC AS $$ DECLARE acc_bal NUMERIC(11,2); BEGIN SELECT order_total INTO acc_bal FROM orders WHERE customer_id = acc_no; RETURN acc_bal; END; $$ LANGUAGE plpgsql;

Table 1: Sample SQL queries and PostgreSQL Equivalents in IDM

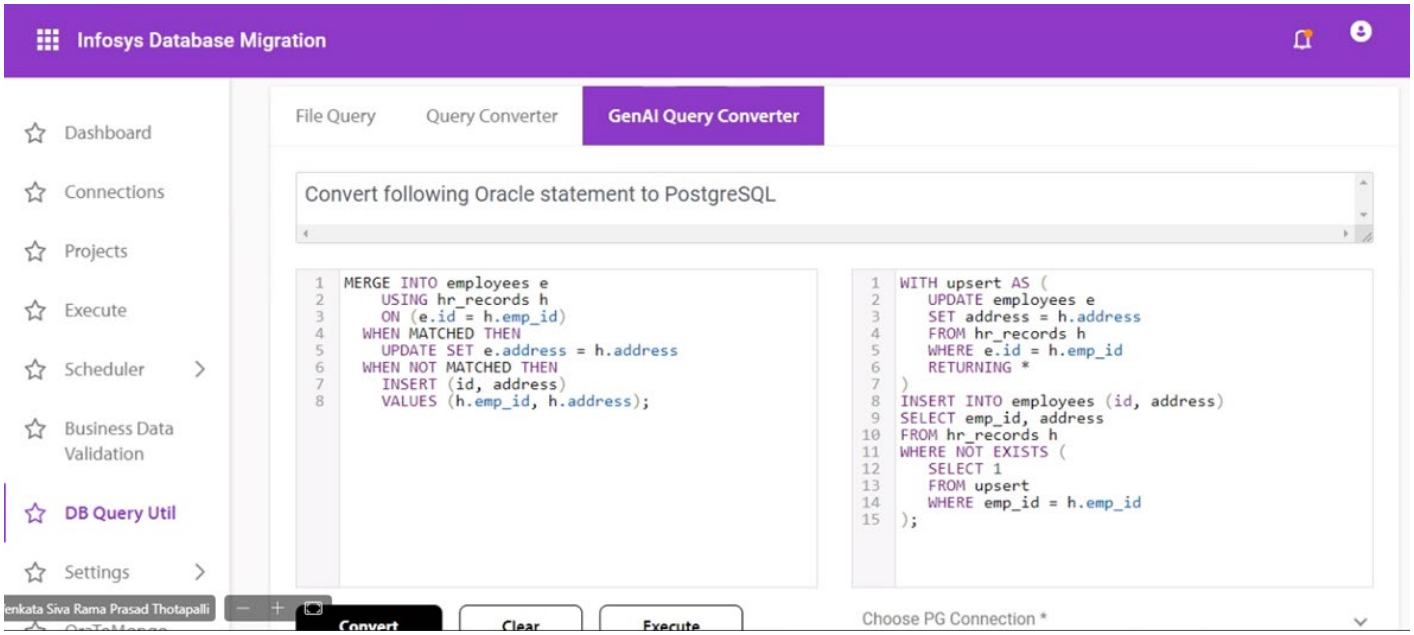


Figure 1: IDM GenAI utility to convert query from Oracle to PostGreSQL

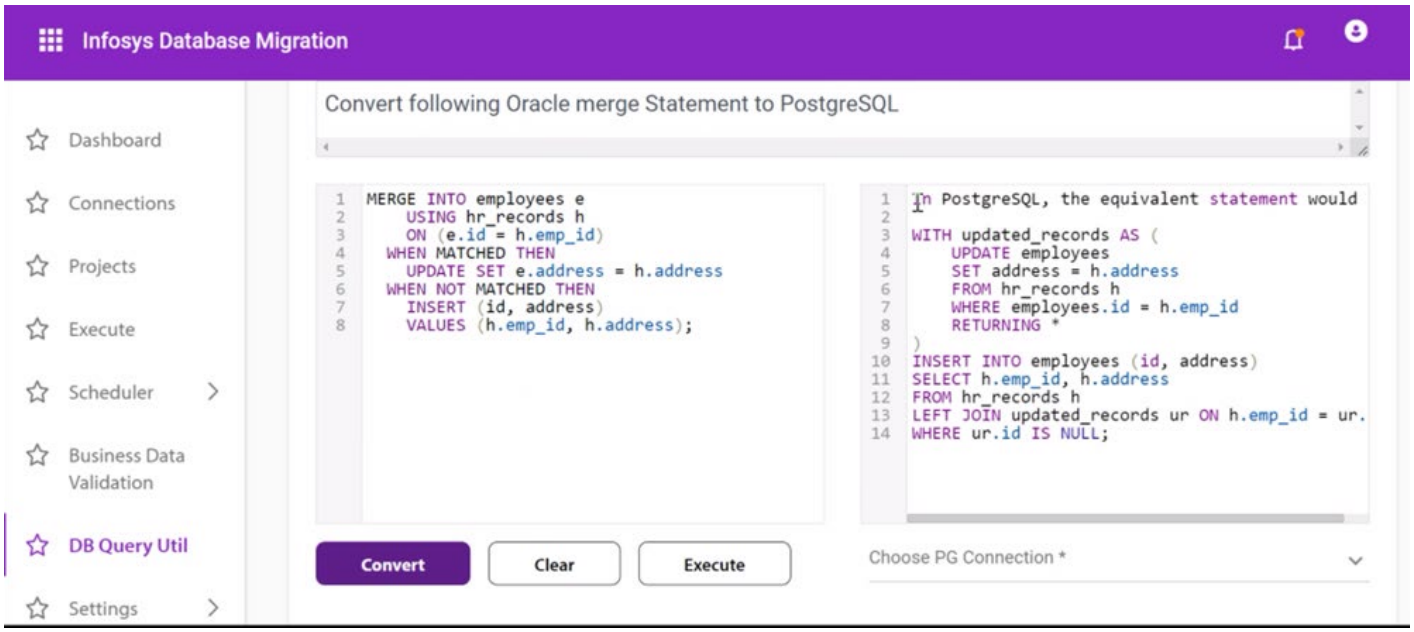


Figure 2: IDM conversion query from Oracle to PostGreSQL

Output in the target database is displayed below for validation.

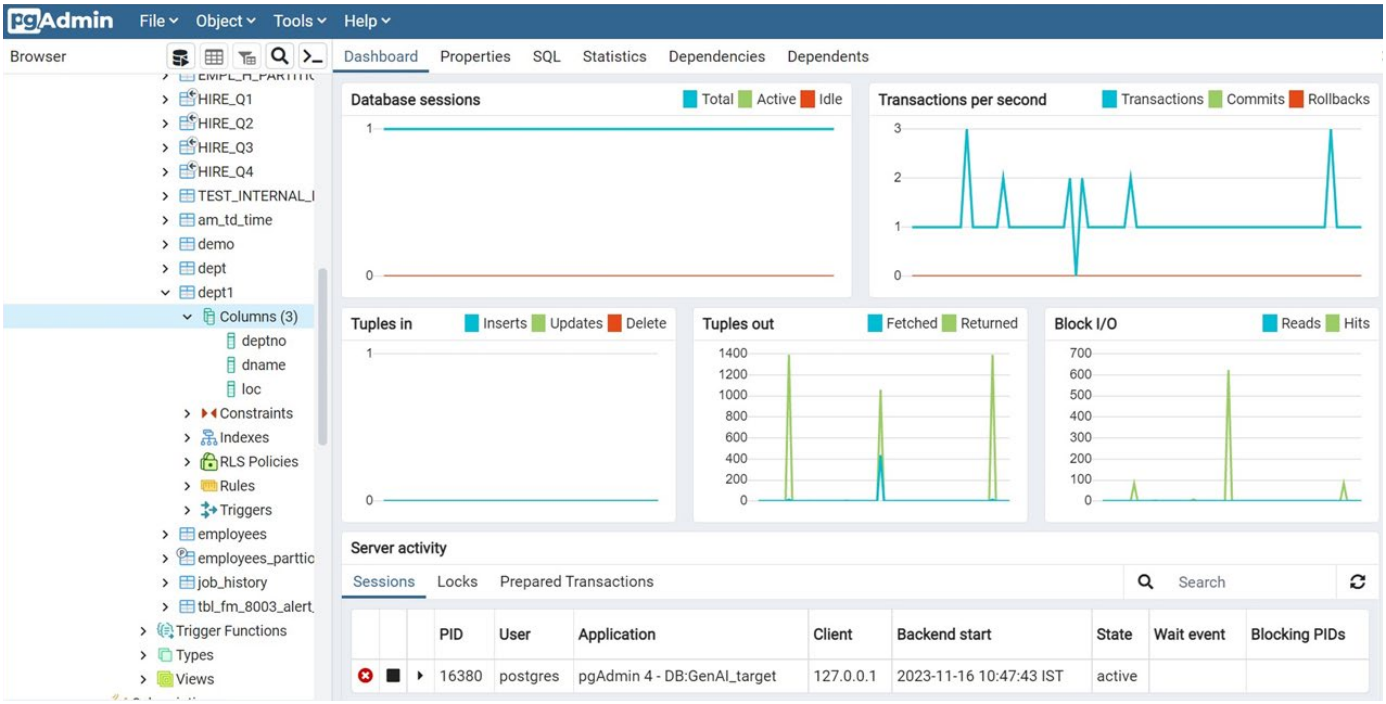


Figure 3: Output in PostgreSQL Database

Advantages

The key capabilities of IDM for database migration are as follows:

1. Pre-Assessment Report
2. Schema Migration and Data Migration
   - ML-based Schema Translation
   - Faster data migration using Apache NiFi-based engine
3. Application DB Refactoring
4. Business Data Validation

Generative AI utility provides customised wrappers for:

- Merge statement
- Pivot
- Unpivot
- Bulk collect
- DB vendor specific packages

ChatGPT simplifies PostgreSQL database migration using Generative AI-based architecture. The Generative AI component adds to IDM's custom built traditional capabilities with the following value adds:

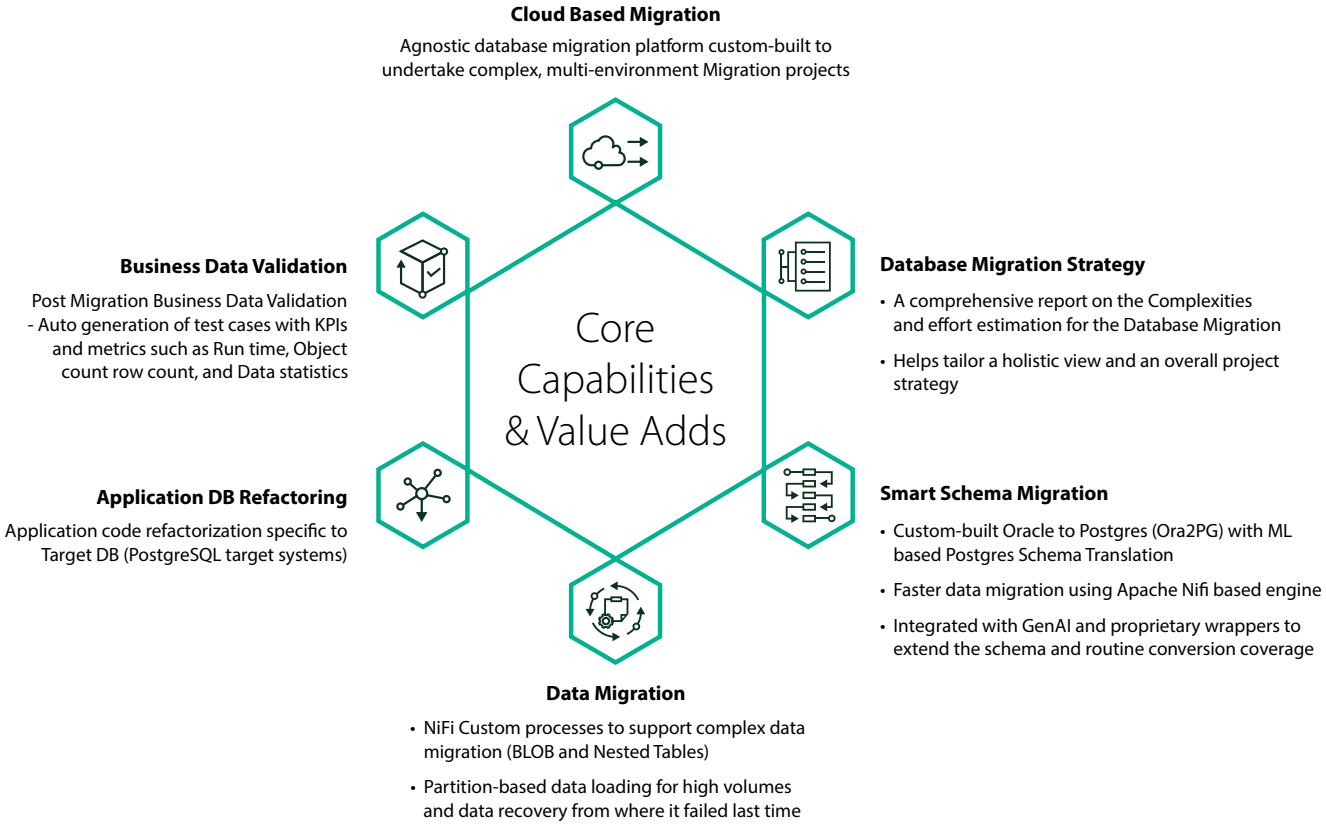
1. It provides ***explanations*** in normal English for specific changes to PostgreSQL.
2. It is a natural PostgreSQL compatible query that can be directly used.
3. Generates business logic explanations.
4. It leverages open-source generative models to increase the fidelity of generated SQL queries, avoid hallucinations, and minimize security risks.
5. Generative AI models generate PostgreSQL as an intermediate step to improve accuracy and minimize security risks.
6. Can be pre-trained for more customised solutions.

Infosys Database Migration (IDM) solution addresses typical data migration issues and challenges and is based on our diverse experience in large data migration projects.

1. Bundled with **database assessment, schema, routines, and data migration, business data validation components in a single tool.**
2. Source database assessment provides incompatibilities and complexity distribution with an approximate effort for schema migration to help plan the migration strategy.
3. High-speed schema and code migration using native DB utilities and offline conversions.
4. Significant number of wrapper functions have been added on top of open-source utilities (Ora2Pg, SQLines) to increase the schema conversion rate.

5. Custom processors have been added to NiFi to support BLOB, CLOB, User Defined Types, and Nested Objects.
6. Partitioned-based data loading to handle high-volume data and recovery from failure points.
7. High volume unstructured data (BLOB as images, documents, etc.) are managed with DB offline utilities and cloud services.
8. Business data validator generates automated test cases to validate the performance of procedures, functions, and queries between source and target databases.
9. Automated generation of test cases to validate key metrics (object counts, row counts, current sequence value, and some data statistics) between source and target DBs.
10. Utilities are included to convert .sql files and queries between databases.

The key capabilities are outlined in the below diagram:



IDM greatly simplifies how legacy databases with expensive maintenance can be migrated to open-source databases using novel Generative AI-based support at enterprise-wide scales.

Conclusion

Infosys Database Migration (IDM) solution addresses typical data migration issues and challenges and is enhanced with Apache NiFi processors and Generative AI integration to handle complex and high-volume data migrations. IDM leverages OpenAI's ChatGPT open Azure platform to enable enhanced accuracy to open-source database migrations.

References

- The Complete Oracle to Postgres Migration Guide: Move and convert Schema, Application & Data (enterprisedb.com)
- Why Replacing Oracle With Postgres is Your Best Bet | EDB (enterprisedb.com)

About the Authors

Eggonu Vengal Reddy

Eggonu Vengal Reddy is a Principal Product Architect with over 20 years of experience in Data Management, specifically Data Warehousing, Data Modeling, Big Data, and Data Science. He has provided architecture and design to develop tools and solutions to handle enterprise-wide database migrations, master data management, data quality and wrangling, explorative analysis, and feature engineering in the Machine Learning life cycle.

Tushar Subhra Das

Tushar Subhra Das is a Senior Business Data Analyst with over 10 years of experience in Data Migration and Governance. He has worked with Europe and Australia-based insurance and logistics clients for Data Migration, MDM and Data Quality, and process governance. In his current role, Tushar is responsible for APAC and EMEA data migration deployments and enhancements, including product developments for iDSS as the next-generation industry-standard data management platform.

Gopinadh Bapatla

Gopinadh Bapatla is a Senior Technology Architect with over 20 years of experience in Application Development, Data Analysis, Data Science, Machine Learning, and Big Data. He has provided architecture and design to develop tools and solutions to handle data quality by data wrangling, explorative analysis, feature engineering, and building Machine Learning models.

Venkata Siva Rama Prasad Thotapalli

Venkata Siva Rama Prasad Thotapalli is a Senior Technical Manager with over 12 years of experience in Data Migration and Governance. He has worked with Europe and UK insurance and Life Sciences clients for Data migration, MDM and Data Quality. In his current role, Venkat is responsible for IDM deployments and enhancements, including product developments for iDM as the next-generation industry-standard data management platform.

Dheeraj Chauhan

Dheeraj Chauhan is a Senior Technology Architect with over 20 years of experience majorly in Application Development, Data Analysis and Management, Risk Management tools, Database update and Imaging Rendering, etc. He has designed architecture to develop tools and solutions to handle/update data, migration of schema and data, feature engineering, and building Machine Learning models.

For more information, contact askus@infosys.com

Infosys®
Navigate your next