

Team Project HWS 2025

Using LLM-Agents in Data Integration Pipelines



Hello

- **Prof. Dr. Christian Bizer**
- Professor for Information Systems V
- Research Interests:
 - Web-based Systems
 - Large-scale Data Integration
 - Data and Web Mining
- Room: B6, 26 - B1.15
- eMail: christian.bizer@uni-mannheim.de
- Consultation: Wednesday, 13:30-14:30



Hello

- **M. Sc. Aaron Steiner**
- Graduate Research Associate
- Research Interests
 - Entity Matching using LLMs
 - LLM agents using RAG
 - Data Integration
- Office: B6, 26 - C 1.04
- eMail: aaron.steiner@uni-mannheim.de



Agenda of Today's Kickoff Meeting

1. A round of introductions: You and Your Experience
2. Introduction to Data Integration
3. Project Goals
4. Organization
5. Specific Subtasks
6. Schedule
7. Formal Requirements
8. Related Work

You and Your Experience

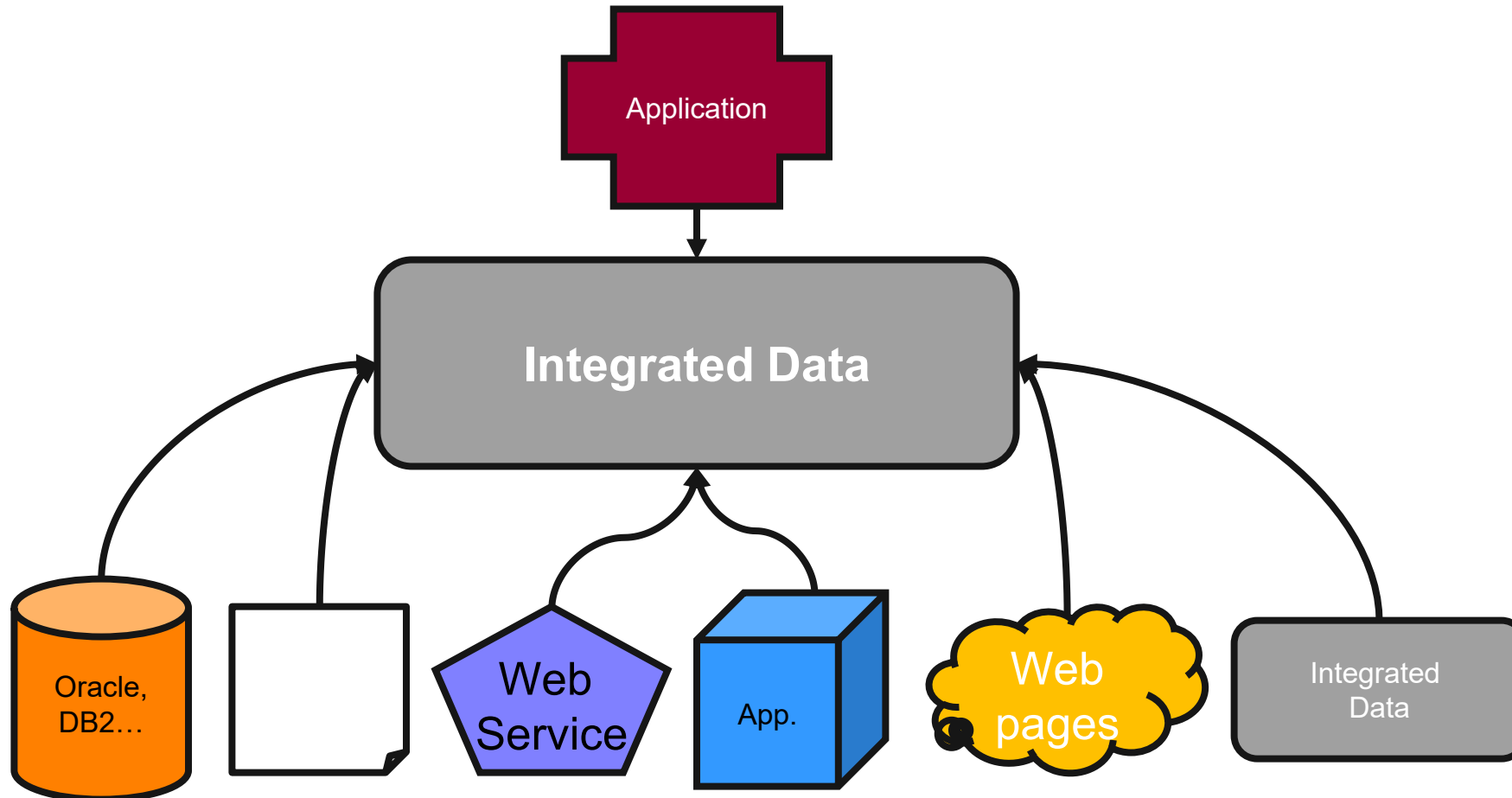
- A Short Round of Introductions
 - What are you studying? Which semester?
 - Which DWS courses did you already attend?
 - What are your programming skills?
 - Do you have any experience with data integration?
 - What experience do you have with LLMs and prompt engineering?
 - Have you worked with cli agent tools? (Claude code, gemini cli, codex)
- Participants
 1. Nicolas Blazek
 2. Abdullah Momin
 3. Gregor Debus
 4. Onur Memiş

2. What is Data Integration?

- Databases and machine learning frameworks are great: They let us manage and analyze huge amounts of data
 1. **assuming** you've put it all into a single schema
 2. **assuming** the database doesn't contain duplicate records
 3. **assuming** that data is current and contains no data conflicts
- In reality, applications often need to work with data from multiple independently created data sources
 1. different sources use different data models
 2. different sources use different schemata
 3. different sources describe the same real-world entity
 4. different sources provide conflicting data about a single entity
 5. different sources provide different access mechanisms

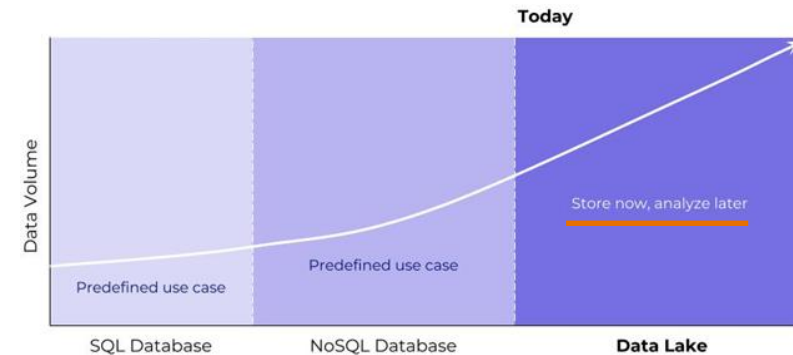


Traditional Data Integration

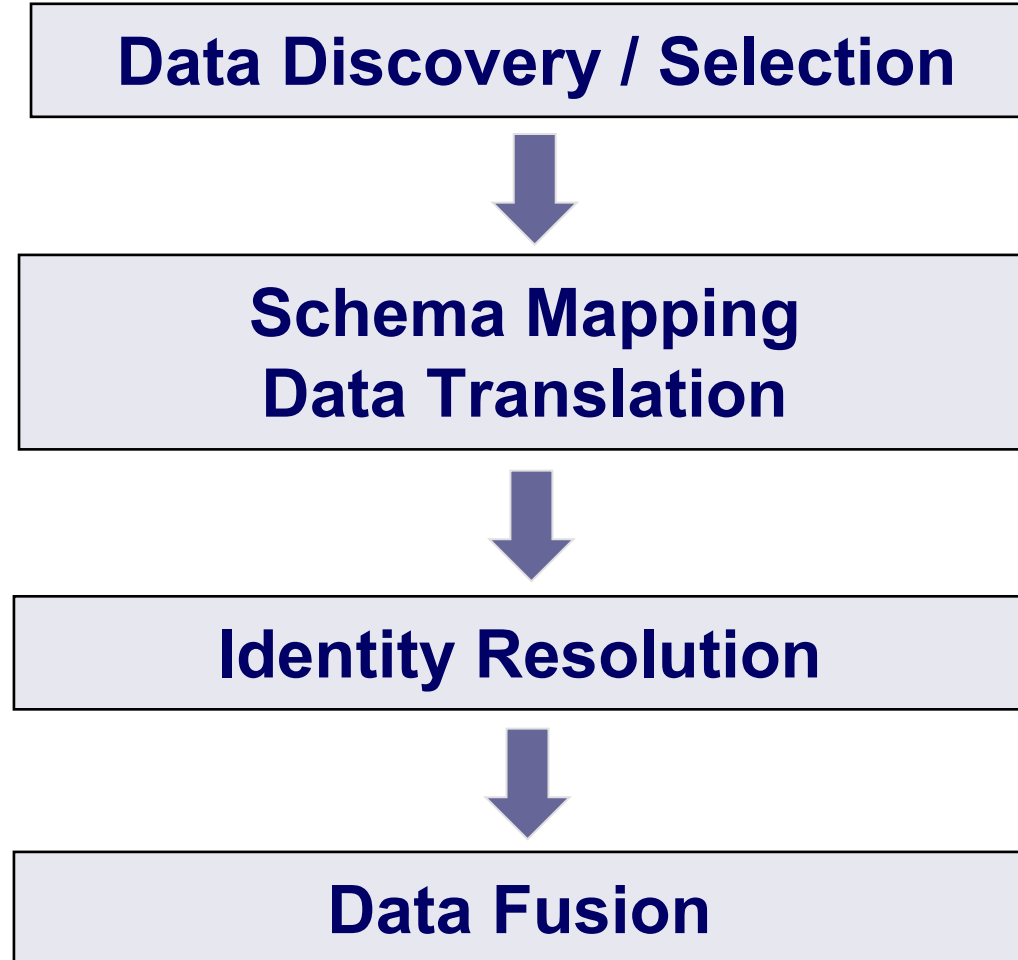


Data Lakes

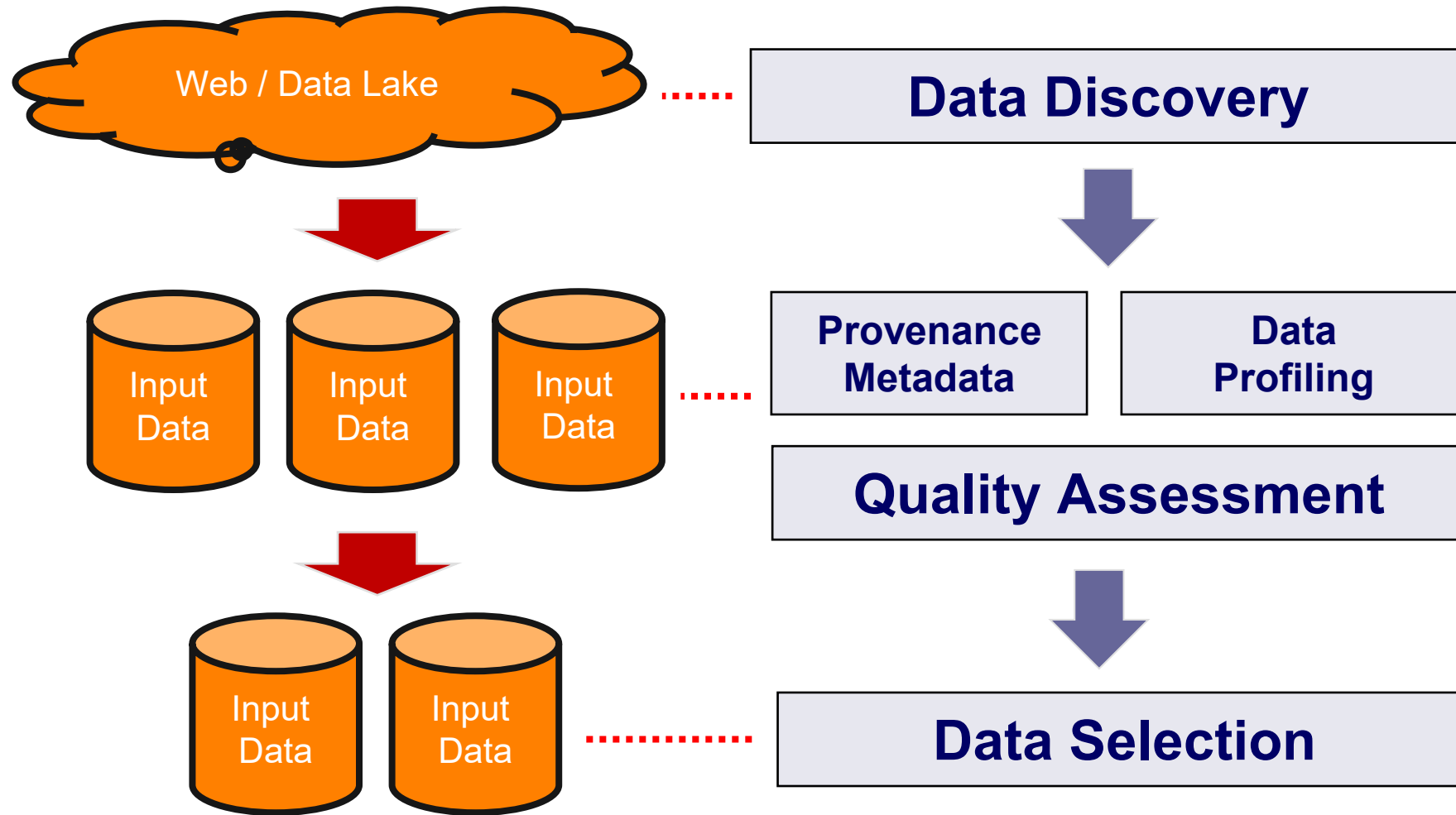
- Data lakes
 - are repositories of raw data in different formats
 - collect or generate metadata about datasets
 - provide a common access interface
- different, not yet known use cases
- are used in a schema-on-read fashion: **Pay-as-you-go integration**
- target users: data scientists



The Data Integration Process

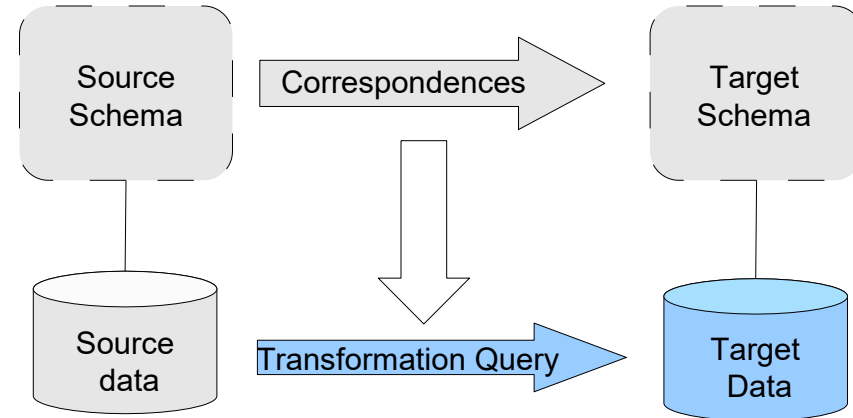
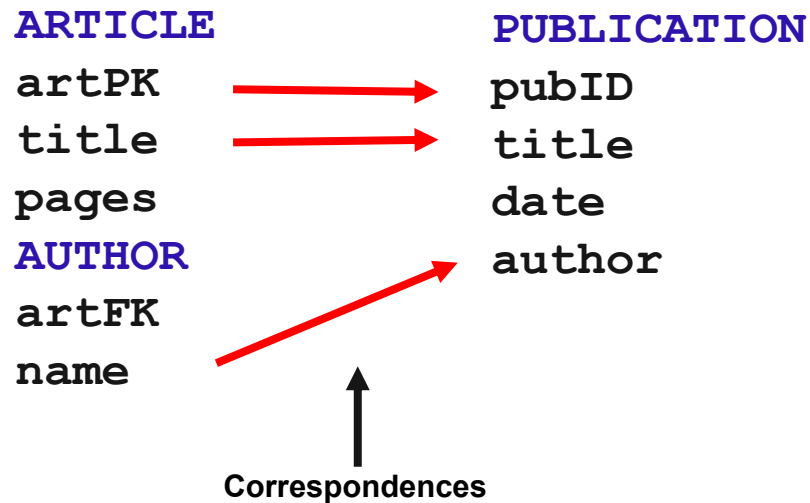


1. Data Discovery and Selection



2. Schema Mapping and Data Translation

- Goal: Resolve structural and schema-related semantic heterogeneity by**
- 1. finding correspondences between elements within different schemata.**
 - 2. translate data to a single target schema based on these correspondences.**



3. Identity Resolution

Goal: Identifying all records in all data sources that describe the same real-world entity.

Basic Approach:

Compare records using a combination of **attribute-specific similarity metrics** or **LLMs for record-level comparison**

If record are similar enough → consider records to describe the same real-world entity

DB1	CID1243	Chris Miller	12/20/1982	Bardon Street, Melville	32 sales
DB2	34	Christian Miller	2/20/1982	7 Bardon St., Melwille	24 sales
DB3	427859	Chris Miller	12/14/1973	7 Bardon St., Madison	13 sales

4. Data Fusion

Goal: Resolve data conflicts by combining attribute values from duplicate records into a single consolidated description of an entity.

Basic Approach:

Assess the **quality** of data sources / records / values

Quality dimensions: timeliness, reputation of source, ...

Apply a **conflict resolution function** to choose most promising values or to correct values

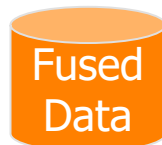
Example functions: highest estimated quality, voting, average, ...



EAN1243	Chris Miller	12/20/1982	Bardon Street, Melville	32 sales
---------	--------------	------------	-------------------------	----------

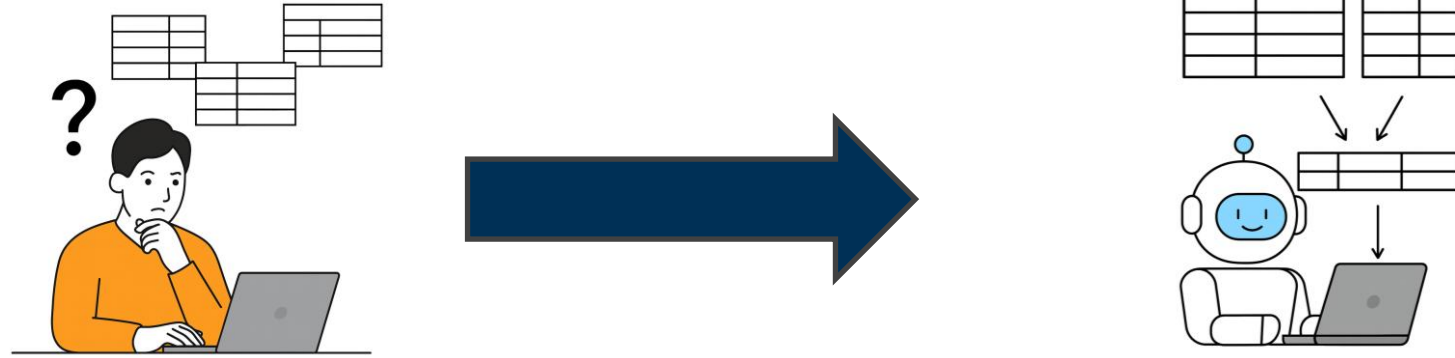


34	Christian Miller	2/20/1982	7 Bardon St., Melville	24 sales
----	------------------	-----------	------------------------	----------



EAN1243	Christian Miller	12/20/1982	7 Bardon Street, Melville	56 sales
---------	------------------	------------	---------------------------	----------

From Manual Data Integration ...To Self-Improving Pipelines



- Data integration = iterative, messy, repetitive, error-prone
- Manual configuration of pipeline
- Labeling many examples
- Manual error analysis
- LLM agents read execution reports and perform error analysis
- Agents improve code of integration pipeline to fix errors
- using code synthesis (the cool stuff)

This project explores how LLM agents can program, evaluate, and improve integration pipelines by themselves

Project Goals – Framework Setting

The project goal is to develop a **multi-agent data integration system** that operates in an automated, unsupervised mode.

- The system should cover the complete data integration pipeline, starting after data selection:
- **Input:** multiple datasets, potentially with metadata (e.g., Kaggle dataset pages)
- **Output:** a single integrated dataset
- The system should use the PyDI framework as its foundation.
- A key component will be an issue tracker, where agents can report and solve problems.



Project Goals - Agent Capabilities to Develop:

The primary goal is to develop a framework that operates in an automated, unsupervised mode.

- Setup and end to end data integration pipeline
- Should monitor the execution of the pipeline using reports
- Derive improvements to the pipeline from output data and reports autonomously
- Possibly generate validation dataset using RAG to further improve the pipeline



Learning Goals

Improve your technical skills

- Gain expertise in Large Language Models and LLM Agents.
- Learn to build a multi-agent system for a complex task.
- Deepen your understanding of the end-to-end data integration pipeline.

Improve your soft skills

- Work as part of a team on a complex, research-oriented project.
- Organize yourself and assign tasks based on your skills.
- Communicate and coordinate your work.

The PyDI Framework

- **Python Data Integration Framework** is an end-to-end data integration framework that covers all steps of the process, from schema matching to data fusion.
- **Key Features:**
 - **Modular Design:** PyDI is a set of independent, composable modules that operate on pandas DataFrames.
 - **Comprehensive Coverage:** The framework includes modules for:
 - Information extraction
 - Schema Matching
 - Data Translation
 - Entity Matching
 - Data Fusion
 - **Modern & Traditional Methods:** It offers a mix of traditional methods and modern techniques using LLMs and embeddings.
- <https://github.com/wbsg-uni-mannheim/PyDI/>
- You can learn more through the [PyDI Tutorial](#) or the code examples in the [Wiki](#).

Team Project Organization

Duration: 6 months (25 September 2025 – 25 March)

Participants: 4 people

Type of work: Team and subgroup based

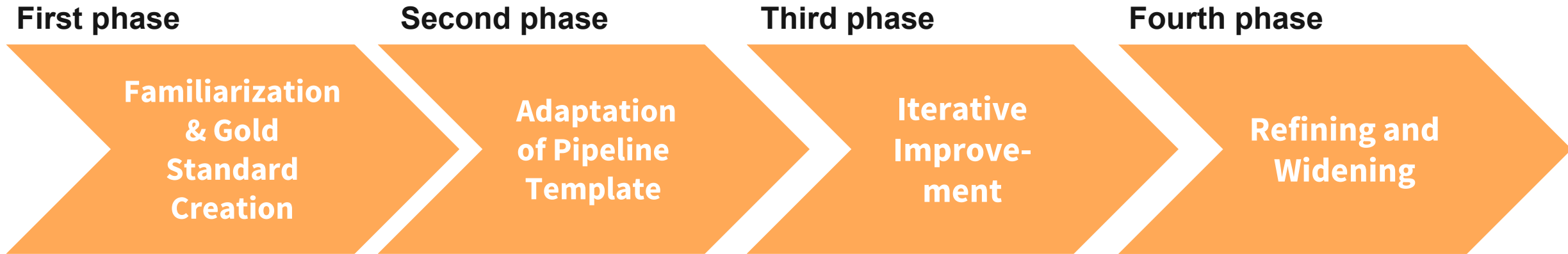
Milestones: 4 project phases

ECTS Points: 12

Evaluation

- Intermediate presentations
- Agents including documentation and code
- Individual contribution to the deliverables is graded
- Personal project diary

Team Project Organization: Project Phases



Phase 1: Familiarization & Gold Standard Creation (4 weeks)

Objective: Get familiar with Data Integration concepts, the PyDI framework, and the landscape of agent development frameworks.

Key Tasks & Deliverables:

– Develop Use Cases & Gold Standards:

- Implement **two interesting data integration use cases** using the existing PyDI framework.
- Crucially, while implementing, you must create high-quality **"gold standards"** for these use cases. This data is essential for the quantitative evaluation of your agent's pipeline performance in later phases.
- See requirements and data sources recommended for [IE683 Student Projects](#)

– Explore Agent Frameworks:

- Make yourselves familiar with the core concepts of coding agents, with a specific focus on **CLI-based agents** (e.g., Gemini-CLI).
- Research and analyze the current state of agent development frameworks.

Phase 1: Familiarization & Gold Standard Creation (4 weeks)

Decide on Your Development Stack:

- Based on your analysis, you must **decide on the development stack** you will use to implement your agent.
- Your analysis should provide an overview of the current landscape and compare at least two primary approaches:
 - **Pure CLI-based:** Using a tool like Gemini-CLI, potentially with sub-agents.
 - **Wrapped CLI:** Using a library like LangChain or a similar framework to wrap and control a CLI-based agent.
- This decision should be well-documented and justified by your findings.

General Note:

- Priority: Agent functionality. The goal is to create and improve data integration pipelines.
- User Interface: A UI is optional and not required.
- Grading: Your grade will be based on the agent's core abilities, not on UI design.

Phase 2: Adaptation of Pipeline Template (6 weeks)

Objective: Develop an agent that can adapt a generic PyDI data integration pipeline to a new use case involving new datasets.

Key Capabilities:

- The agent must be able to understand the logic and steps of the pipeline template.
- It should be able to adapt this pipeline to new, unseen datasets and use cases.

Your Task:

- **Provide the Agent with a Blueprint:** Generalize the existing **PyDI tutorial notebook** and the pipelines you created in Phase 1 to derive a generic process template.
- **Execute the Transfer:** The agent's primary task is to take the pipeline logic from the template and successfully apply it to a **new use case and datasets**.
- **Compare and Analyze:** Manually implement the same pipeline transfer for the new use case yourselves. Compare your own implementation to the one generated by the agent.
- **Identify Improvements:** Based on this comparison, identify and document potential ways the agent's process and output can be improved. This task of implementing the improvements will be the focus for the agent in Phase 3.

Agent's Playbook for Pipeline Transfer

- **Profile Data:** Identify nulls, data types, and value distributions.
- **Sample Strategically:** Use expensive methods on samples to bootstrap cheaper models for full datasets.
- **Schema Matching:** Run a default mapper. Likely LLM based schema matching.
- **Data Normalization:** Apply standard steps (e.g., fix data types, trim whitespace).
- **Entity Matching:** Default to embedding-based blocking. Use a standard matcher and have criteria for when to swap it out based on performance.
- **Data Fusion:** Apply a default fusion strategy based on data type. Adapt based on the frequency of value conflicts.

Phase 3: Autonomous Improvement (6 weeks)

Objective: Create an agent that can autonomously improve the data integration pipeline generated in Phase 2.

Key Capabilities: The agent's core task is to move beyond simple execution and into a cycle of continuous improvement. This will be achieved by:

- **Log & Report Analysis:** The agent will analyze logs and performance reports to identify bottlenecks, errors, and areas with low-quality output.
- **Hypothesize Improvements:** Based on its analysis, the agent will form hypotheses about how to improve specific subtasks within the pipeline.
- **Experimentation:** The agent will then autonomously experiment with different methods and parameters to test its hypotheses.

Phase 3: Autonomous Improvement (6 weeks)

Examples of Agent-Led Improvements:

- Refining Data Normalization: If the agent detects a high number of format outliers for a specific attribute, it could experiment with different normalization rules or even apply an information extraction model to standardize the values.
- Improving Schema Mapping: By analyzing value distributions, the agent could identify and correct mismatched attributes in the schema mapping.
- Optimizing Entity Matching: The agent could analyze the performance of the entity matching step and decide to experiment with different matchers or prompts.

Intermediate Presentation (02.02.2026): At the end of this phase, you will present your progress. You should be prepared to demonstrate the agent's ability to autonomously analyze and improve a data integration pipeline.

High-Level Schedule

- **Phase 1 (approx. 1 month):** Onboarding, overview of the CLI Agent Framework, and completion of 2 WDI projects.
- **Phase 2 (approx. 6 weeks):** Development of the first agent framework to create pipelines for new use cases.
- **Phase 3 (approx. 6 weeks):** Implementation of the agent for iterative pipeline improvement through log analysis.
- **Intermediate Presentation:** With the Prof.
- **Remaining Time (approx. 8 weeks):** Refining and Widening. Details will be decided after Phase 3.
- **Final Presentation:** A final presentation with the Prof. will conclude the project.

Formal Requirements & Consultation

Deliverables

1. On the presentation dates provide us via e-mail with:
 - **Presentation slides**
 - **Task to member report:** excel sheet stating which team member conducted which subtask
 - **Code/Data:** link or zipped folder with your code and data

2. Final Deliverables
 - **Webpage** describing the agents and your experiments
 - **Installer** for setting up the agents locally

All deliverables must be sent to Aaron & Chris!

Formal Requirements & Consultation

Final grade

- 20% for each phase
- 20% for final deliverables
- Late submission: reduces your grade -0.3 per day

Consultation

- Regular progress meetings every **two** weeks. Need to agree on time slot.
- If questions arise in between meetings, feel free to write Aaron an email with questions.

Useful Software & Resources

- [Gemini CLI](#)
 - Free tier: 60 requests/min and 1,000 requests/day
 - Get an in-depth look at the inner workings of a CLI coding agent.
- [LangChain](#)
- [LLM and Agents course](#)
- [Web Data Integration course \(Archive version\)](#)
- Team Cooperation
 - GitHub/Lab for the code base
 - Project Management Tool of your choice
- Processing and GPUs
 - Google Colab: <https://colab.research.google.com/>
 - BwUniCluster2.0: https://wiki.bwhpc.de/e/Category:BwUniCluster_2.0

Related Work: LLM Agents for Data Integration

- Santos, Aécio, et al. "**Interactive Data Harmonization with LLM Agents: Opportunities and Challenges.**" In Novel Optimizations for Visionary AI Systems (NOVAS '25) (2025). arXiv:2502.07132 [cs.AI].
- Tengjun Jin, et al. "ELT-Bench: An End-to-End Benchmark for Evaluating AI Agents on ELT Pipelines." arXiv:2504.04808v2 (2025)
- Mattia Di Profio, et al. "FlowETL: An Autonomous Example-Driven Pipeline for Data Engineering" arXiv:2507.23118 (2025)
- Soorya Shingekar, et al.: "Agentic AI framework for End-to-End Medical Data Inference." arXiv:2507.18115 (2025)
- Wu, Zhenyu, et al. "Schema Inference for Tabular Data Repositories Using Large Language Models." arXiv:2509.04632 (2025).
- Qi, Danrui, et al. "CleanAgent: Automating Data Standardization with LLM-based Agents." arXiv:2403.08291 [cs.LG] (2025).
- Tran, Khanh-Tung, et al. "Multi-Agent Collaboration Mechanisms: A Survey of LLMs." arXiv:2501.06322 [cs.AI] (2025).
- Wang, Peiran, et al. "Large Language Model-based Data Science Agent: A Survey." arXiv:2508.02744 [cs.AI] (2025).
- Schmidgall, Samuel, et al. "Agent Laboratory: Using LLM Agents as Research Assistants." arXiv:2501.04227 [cs.HC] (2025).

Questions?

