Inmon vs. Kimball – Data Warehouse Development Approaches

|  |  |  |
| --- | --- | --- |
| Aspect | Inmon Approach  (Top-Down) | Kimball Approach  (Bottom-Up) |
| Philosophy | Build Enterprise Data Warehouse (EDW) first | Build Dimensional Data Marts first |
| Data Modeling | Normalized (3NF) for EDW | Dimensional Modeling (Star/Snowflake schema) |
| Architecture | ODS → EDW (3NF) → Data Marts | Staging → Dimensional Data Marts → Integrated DW |
| Data Integration | Centralized at EDW level | Through Conformed Dimensions across marts |
| ETL Process | Complex: Source → ODS → EDW → Marts | Simpler: Source → Staging → Marts |
| Delivery Speed | Slower initial delivery | Faster initial delivery |
| Cost | Higher upfront cost | Lower initial cost |
| Flexibility | Less flexible, more rigid upfront | More flexible, easy to scale |
| Governance | Strong central data governance | Departmental control, less centralized |
| Best For | Large enterprises (Banking, Government) | Fast BI and reporting (Retail, Sales) |
| User Focus | IT-driven | Business-driven |
| Schema Design | EDW in 3NF; marts later in dimensional model | Star/Snowflake schema used directly in marts |
| Use of ODS | Commonly used | Rarely used |
| Maintenance | Centralized and complex | Distributed and simpler per mart |
| Scalability | Strong long-term scalability | Easier to start, may need integration later |

## dataSources analysis PI extract

Reports

Dashboards

…

E-DWH

Inmon approach

Data sources data marts pi

Kimball approach

Full Load vs. Incremental Load

|  |  |  |
| --- | --- | --- |
| Aspect | Full Load | Incremental Load |
| Definition | Loads the entire dataset from source to target each time | Loads only new or changed data since the last load |
| Data Volume | High (entire data set) | Low (only deltas/incremental data) |
| Speed & Performance | Slower, resource-intensive | Faster, more efficient |
| Use Case | Initial load, small datasets, or complete refresh needs | Large datasets, regular updates with minimal data change |
| Data Overwrite | Yes – replaces the existing data | No – appends/updates only changed data |
| ETL Complexity | Simple – no need to track changes | More complex – requires change tracking (e.g., timestamps) |
| Storage Cost | Higher, as all data is moved/stored repeatedly | Lower, as only part of data is moved |
| Risk of Errors | Lower risk of missing data | Higher risk if change capture fails or is misconfigured |
| Maintenance | Easier to implement and maintain | More complex but scales better in the long run |