Education_Gold_Generation_Notebook

New notebook

Medallion Architecture - Gold Tier

In [ ]:

```
from pyspark.sql import functions as F

SILVER_DB = "silver"
GOLD_DB   = "gold"

def write_gold(df, table_name: str):
    (df.write.mode("overwrite").format("delta")
       .saveAsTable(f"{GOLD_DB}_{table_name}"))

# =======================
# GOLD DIMENSIONS
# =======================

gold_dimDate = spark.table(f"{SILVER_DB}_dimDate")
write_gold(gold_dimDate, "dimDate")

display(gold_dimDate)
```

In [ ]:

```
gold_dimGeography = spark.table(f"{SILVER_DB}_dimGeography")
write_gold(gold_dimGeography, "dimGeography")

gold_dimTrust = spark.table(f"{SILVER_DB}_dimTrust")
write_gold(gold_dimTrust, "dimTrust")

# dimSchool enriched with trust + optional geography
gold_dimSchool = (
    spark.table(f"{SILVER_DB}_dimSchool").alias("s")
    .join(spark.table(f"{SILVER_DB}_dimTrust").alias("t"), on="TrustId", how="left"
    .join(spark.table(f"{SILVER_DB}_dimGeography").select("GeoKey","Ward","Postcode
          on="GeoKey", how="left")
    .select(
        "s.SchoolKey","s.SchoolId","s.SchoolName","s.SchoolType","s.Phase","s.IsSen
        "t.TrustKey","s.TrustId","t.TrustName","t.TrustType","t.LocalAuthority",
        "s.GeoKey","g.Ward","g.PostcodeArea","g.Region"
    )
)
write_gold(gold_dimSchool, "dimSchool")

gold_dimPupil = spark.table(f"{SILVER_DB}_dimPupil")
write_gold(gold_dimPupil, "dimPupil")

gold_dimSendNeed = spark.table(f"{SILVER_DB}_dimSendNeed")
write_gold(gold_dimSendNeed, "dimSendNeed")

gold_dimIntervention = spark.table(f"{SILVER_DB}_dimIntervention")
write_gold(gold_dimIntervention, "dimIntervention")
```

```
# =======================
# GOLD FACTS
# =======================

# gold_factEnrolment (as-is, business-ready)
gold_factEnrolment = spark.table(f"{SILVER_DB}_factEnrolment")
write_gold(gold_factEnrolment, "factEnrolment")

# gold_factAttendance (filter to school days using dimDate)
gold_factAttendance = (
    spark.table(f"{SILVER_DB}_factAttendanceDaily").alias("a")
    .join(spark.table(f"{GOLD_DB}_dimDate").select("DateKey","IsSchoolDay").alias("
          on="DateKey", how="inner")
    .where(F.col("d.IsSchoolDay") == 1)
    .select(
        "a.PupilKey","a.SchoolKey","a.DateKey",
        "a.SessionAM","a.SessionPM","a.AttendanceCode",
        "a.SessionsPossible","a.SessionsPresent",
        "a.IsAbsentDay"
    )
)
write_gold(gold_factAttendance, "factAttendance")

# gold_factAttainment
gold_factAttainment = (
    spark.table(f"{SILVER_DB}_factAttainmentTermly")
    .select(
        "PupilKey","SchoolKey","DateKey","Subject","GradeBand","ScaledScore",
        "ExpectedProgressFlag","AtOrAboveFlag"
    )
)
write_gold(gold_factAttainment, "factAttainment")

# gold_factBehaviour
gold_factBehaviour = spark.table(f"{SILVER_DB}_factBehaviourIncidents")
write_gold(gold_factBehaviour, "factBehaviour")

# gold_factSendProvision
gold_factSendProvision = (
    spark.table(f"{SILVER_DB}_factSendProvision")
    .select(
        "PupilKey","SchoolKey","DateKey","NeedKey",
        "EHCPFlag","ProvisionHoursPerWeek","PlacementType",
        "AnnualTopUpFunding","FundingBand"
    )
)
write_gold(gold_factSendProvision, "factSendProvision")

# gold_factIntervention
gold_factIntervention = (
    spark.table(f"{SILVER_DB}_factInterventionParticipation")
    .select(
        "PupilKey","SchoolKey","InterventionKey",
        "StartDateKey","EndDateKey","TargetOutcome",
        "BaselineValue","EndValue","DeltaValue","ImpactBand"
    )
)
write_gold(gold_factIntervention, "factIntervention")

print("Gold complete.")
```