Notes from Study

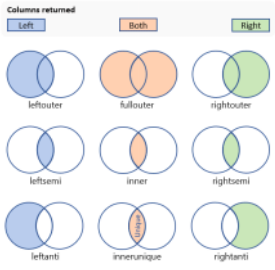
Wednesday, December 18, 2024 12:44 PM

Learning Path	Module	Time	Notes	Links to check
Ingest data with Microsoft Fabric - Training | Microsoft Learn	Ingest Data with Dataflows in Microsoft Fabric - Training | Microsoft Learn	8		Power Query documentation - Power Query | Microsoft Learn

Ingest data with Microsoft Fabric - Training | Microsoft Learn	Orchestrate processes and data movement with Microsoft Fabric - Training | Microsoft Learn	72	Practice PySpak:
			`df.write.format("delta").mode("append").saveAsTable("sales")`
			
			`# Derive FirstName and LastName columns`
			`df =  df.withColumn("FirstName", split(col("CustomerName"), " ").getItem(0)).withColumn("LastName",split(col("CustomerName"), " ").getItem(1))`
			
			`## Add month and year columns`
			`df = df.withColumn("Year", year(col("OrderDate"))).withColumn("Month",month(col("OrderDate")))`
			`display(df)`
			
			`from pyspark.sql.functions import *`
			`# Read the new sales data`
			`df = spark.read.format("csv").option("header","true").load("Files/RawData/Sales")`
			
			Review how to use parameters with notebooks
				Look for PySpark syntax

Ingest data with Microsoft Fabric - Training | Microsoft Learn
 	
and
 	
Implement Real-Time Intelligence with Microsoft Fabric - Training | Microsoft Learn
	Get started with Real-Time Intelligence in Microsoft Fabric - Training | Microsoft Learn	48	Review links for Join types
			
			`Weather`
			`| summarize EventCount = count() by State`
			`| sort by EventCount`
			
			`Weather`
			`| extend damage = DamageProperty + DamageCrops`
			`| summarize sum(damage) by bin(StartTime, 7d)`
			`| render columnchart`
			
			`Weather`
			`| extend damage = DamageProperty + DamageCrops`
			`| summarize sum(damage) by EventType`
			`| render piechart`
				Tutorial: Learn common Kusto Query Language operators - Kusto | Microsoft Learn
				Tutorial: Use aggregation functions in Kusto Query Language - Kusto | Microsoft Learn
				Tutorial: Join data from multiple tables - Kusto | Microsoft Learn
				join operator - Kusto | Microsoft Learn
				Supported sources in to Real-Time hub - Microsoft Fabric | Microsoft Learn
				Supported sources in to Real-Time hub - Microsoft Fabric | Microsoft Learn
				Process event data with the event processor editor - Microsoft Fabric | Microsoft Learn
				Add and manage eventstream destinations - Microsoft Fabric | Microsoft Learn
				
				Review RoundL
				
				Write your first query with Kusto Query Language - Training | Microsoft Learn
				Explore the fundamentals of data analysis using Kusto Query Language (KQL) - Training | Microsoft Learn
				Gain insights from your data by using Kusto Query Language - Training | Microsoft Learn
				Write multi-table queries by using Kusto Query Language - Training | Microsoft Learn

Ingest data with Microsoft Fabric - Training | Microsoft Learn and Implement Real-Time Intelligence with Microsoft Fabric - Training | Microsoft Learn	Use real-time eventstreams in Microsoft Fabric - Training | Microsoft Learn	32	Review window functions 	https://learn.microsoft.com/en-us/training/modules/explore-event-streams-microsoft-fabric/4-route-event-data-to-destinations
Ingest data with Microsoft Fabric - Training | Microsoft Learn and Implement Real-Time Intelligence with Microsoft Fabric - Training | Microsoft Learn	Work with real-time data in a Microsoft Fabric eventhouse - Training | Microsoft Learn	17	Review KQL best practices Functions: getmonth(), getyear(), hourofday(), now(), ago(30min), ago(1d), ingestion_time(), summarize () SummaryColumnN=ame = avg(ValueColumntoSumUp) by ColumToGroupByWith Review Materialized view syntax: .create materialized-view NameOfView on table NameofTable .create async materialized-view with (backfill=true) --> To ingest existing data Review function syntax .create-or-alter function trips_by_min_passenger_count(num_passengers:long) case(isempty(pickup_boroname) or isnull(pickup_boroname), "Unidentified", pickup_boroname)	E

Implement Real-Time Intelligence with Microsoft Fabric - Training | Microsoft Learn	Create a Real-Time Dashboard - Microsoft Fabric | Microsoft Learn	51	arg_max():
			Finds a row in the table that maximizes the specified expression. It returns all columns of the input table or specified columns.
			
			arg_max() (aggregation function) - Kusto | Microsoft Learn
			
			bikes | where ingestion_time() between (ago(30min) .. now()) | summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood | project Neighbourhood, latest_observation, No_Bikes, No_Empty_Docks | order by Neighbourhood asc
			
			`bikes`
			`| where ingestion_time() between (ago(30min)..now())`
			`  and (isempty(['selected_neighbourhoods']) or Neighbourhood  in (['selected_neighbourhoods']))`
			`| summarize latest_observation = arg_max(ingestion_time(),*) by Neighbourhood`
			
			For best performance, if one table is always smaller than the other, use it as the left side of the join operator.
			
			From <https://learn.microsoft.com/en-us/training/modules/multi-table-queries-with-kusto-query-language/2-multi-table-queries>
			The materialize() function caches results within a query execution for subsequent reuse in the query. It's like taking a snapshot of the results of a subquery and using it multiple times within the query. This function is useful in optimizing queries for scenarios where the results:
			Are expensive to compute
			Are nondeterministic
			
			From <https://learn.microsoft.com/en-us/training/modules/multi-table-queries-with-kusto-query-language/2-multi-table-queries>
			 
				Use parameters in Real-Time Dashboards - Microsoft Fabric | Microsoft Learn
				Create real-time dashboards with Microsoft Fabric - Training | Microsoft Learn
				
				arg_max() (aggregation function) - Kusto | Microsoft Learn
				
				Best practices for Kusto Query Language queries - Kusto | Microsoft Learn
				
				Named expressions - Kusto | Microsoft Learn

Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Introduction to end-to-end analytics using Microsoft Fabric - Training | Microsoft Learn	18 minutes	Review admin roles Workspace settings	Workspaces in Microsoft Fabric and Power BI - Microsoft Fabric | Microsoft Learn
Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Get started with lakehouses in Microsoft Fabric - Training | Microsoft Learn	60 minutes	Maybe Review Spark Job Definition	Security in Microsoft Fabric - Microsoft Fabric | Microsoft Learn Create an Apache Spark job definition - Microsoft Fabric | Microsoft Learn Unify data sources with OneLake shortcuts - Microsoft Fabric | Microsoft Learn Workspaces in Microsoft Fabric and Power BI - Microsoft Fabric | Microsoft Learn Roles in workspaces in Microsoft Fabric - Microsoft Fabric | Microsoft Learn

Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Use Apache Spark in Microsoft Fabric - Training | Microsoft Learn	120 mintues	Review code in notebook 1
			`df = spark.read.format("CSV").option("header","true").load("Files/orders/2019.csv")`
			
			`orders_df.write.partitionBy("Year","Month").mode("overwrite").parquet("Files/partitioned_data")`
			
			`df.write.format("delta").saveAsTable("salesorders")`
			
			`from pyspark.sql.functions import *`
			` # Create Year and Month columns`
			` transformed_df = df.withColumn("Year", year(col("OrderDate"))).withColumn("Month", month(col("OrderDate")))`
			` # Create the new FirstName and LastName fields`
			` transformed_df = transformed_df.withColumn("FirstName", split(col("CustomerName"), " ").getItem(0)).withColumn("LastName", split(col("CustomerName"), " ").getItem(1))`
			` # Filter and reorder columns`
			` transformed_df = transformed_df["SalesOrderNumber", "SalesOrderLineNumber", "OrderDate", "Year", "Month", "FirstName", "LastName", "Email", "Item", "Quantity", "UnitPrice", "Tax"]`
			` # Display the first five orders`
			` display(transformed_df.limit(5))`
			
			`sqlQuery = "SELECT CAST(YEAR(OrderDate) AS CHAR(4)) AS OrderYear,\`
			`  SUM((UnitPrice * Quantity) + Tax) AS GrossRevenue\`
			`  FROM salesorders \`
			`  GROUP BY CAST(YEAR(OrderDate) AS CHAR(4))\`
			`  ORDER BY OrderYear"`
			`df_spark = spark.sql(sqlQuery)`
				Master Link: Search for Data Engineering Documentation
				Data Engineering in Microsoft Fabric documentation - Microsoft Fabric | Microsoft Learn
				
				Data engineering and science capacity admin settings - Microsoft Fabric | Microsoft Learn
				Manage settings for data engineering and science capacity - Microsoft Fabric | Microsoft Learn
				Configure and manage starter pools in Fabric Spark. - Microsoft Fabric | Microsoft Learn
				Create custom Apache Spark pools in Fabric - Microsoft Fabric | Microsoft Learn
				Apache Spark runtime in Fabric - Microsoft Fabric | Microsoft Learn
				Create, configure, and use an environment in Fabric - Microsoft Fabric | Microsoft Learn
				High concurrency mode in Apache Spark compute for Fabric - Microsoft Fabric | Microsoft Learn

			df_spark.show()	

Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Work with Delta Lake tables in Microsoft Fabric - Training | Microsoft Learn	1 hour	%%sql
			SET spark.sql.parquet.vorder.enabled=TRUE
			
			%%sql
			CREATE TABLE person (id INT, name STRING, age INT) USING parquet TBLPROPERTIES("delta.parquet.vorder.enabled" = "true");
			
			%%sql
			ALTER TABLE person SET TBLPROPERTIES("delta.parquet.vorder.enabled" = "true");
			
			ALTER TABLE person SET TBLPROPERTIES("delta.parquet.vorder.enabled" = "false");
			
			ALTER TABLE person UNSET TBLPROPERTIES("delta.parquet.vorder.enabled");
			
			--When session level V-Order is not enabled or unset, individual operations need this:
			.option("parquet.vorder.enabled ","true")\\
			
			Merge optimization: for handling unmodified rows
			spark.microsoft.delta.merge.lowShuffle.enabled
			
			Bin-compaction is achieved by the OPTIMIZE command; it merges all changes into bigger, consolidated parquet files.
			Dereferenced storage clean-up is achieved by the VACUUM command.
			
			Control V-Order when optimizing a table
			
			The following command structures bin-compact and rewrite all affected files using V-Order, **independent** of the TBLPROPERTIES setting or session configuration setting
			
			%%sql
			OPTIMIZE <table|fileOrFolderPath> VORDER;
			
			OPTIMIZE <table|fileOrFolderPath> WHERE <predicate> VORDER;
			
			OPTIMIZE <table|fileOrFolderPath> WHERE <predicate> [ZORDER BY (col_name1, col_name2, ...)] VORDER;
			
			Apache Spark performs bin-compaction, ZORDER, VORDER sequentially.
			
			The following commands bin-compact and rewrite all affected files **using the TBLPROPERTIES setting**:
			
			%%sql
			OPTIMIZE <table|fileOrFolderPath>;
			
			OPTIMIZE <table|fileOrFolderPath> WHERE predicate;
			
			OPTIMIZE <table|fileOrFolderPath> WHERE predicate [ZORDER BY (col_name1, col_name2, ...)];
			
			Optimized Write: It dynamically optimizes partitions while generating files with a default 128-MB size.
			Benefits of Optimized Write:
			OPTIMIZE operations will be faster as it will operate on fewer files.
			VACUUM command for deletion of old unreferenced files will also operate faster.
			Queries will scan fewer files with more optimal file sizes, improving either read performance or resource usage.
			
			When to avoid it:
			Non partitioned tables.
			Use cases where extra write latency isn't acceptable.
			Large tables with well defined optimization schedules and read patterns.
			
			spark.conf.set("spark.microsoft.delta.optimizeWrite.enabled", "true")
			
			SET `spark.microsoft.delta.optimizeWrite.enabled` = true
			spark.conf.get("spark.microsoft.delta.optimizeWrite.enabled")
			Using table properties vs. session level: SET TBLPROPERTIES (delta.autoOptimize.optimizeWrite = true)
			
			Get bin size: spark.conf.get("spark.microsoft.delta.optimizeWrite.binSize")
			SET `spark.microsoft.delta.optimizeWrite.binSize`
			
			Optimize: 128 MB, and optimally close to 1 GB
			
			Create table: Use the DeltaTableBuilder API:
			%PySpark
			from delta.tables import *
			DeltaTable.create(spark) \
			.tableName("products") \
			.addColumn("ProductId", "INT") \
			.addColumn("ProductName", "STRING") \
			.addColumn("Category", "STRING") \
			.addColumn("Price", "FLOAT") \
			.execute()
			
			%%sql
			CREATE TABLE salesorders
			(
			Orderid INT NOT NULL,
			OrderDate TIMESTAMP NOT NULL,
			CustomerName STRING,
			SalesTotal FLOAT NOT NULL
			)
			USING DELTA
			
			CREATE EXTERNAL TABLE
			%%sql
			CREATE TABLE
			MyExternalTable
			USING DELTA
			LOCATION 'Files/mydata'
			
			delta_path = "Files/mydatatable"
			df.write.format("delta").save(delta_path)
			
			new_df.write.format("delta").mode("overwrite").save(delta_path)
			new_rows_df.write.format("delta").mode("append").save(delta_path)
			
			In Microsoft Fabric, OptimizeWrite is enabled by default.
			
			# Disable Optimize Write at the Spark session level
			spark.conf.set("spark.microsoft.delta.optimizeWrite.enabled", False)
			
			# Enable Optimize Write at the Spark session level
			spark.conf.set("spark.microsoft.delta.optimizeWrite.enabled", True)
			
			print(spark.conf.get("spark.microsoft.delta.optimizeWrite.enabled"))
			
			In Microsoft Fabric, the Power BI and SQL engines use Microsoft Verti-Scan technology
			V-Order might not be beneficial for write-intensive scenarios such as staging data stores where data is only read once or twice. In these situations, disabling V-Order might reduce the overall processing time for data ingestion.
			
			VACUUM WITH SQL
			%%sql
			VACUUM lakehouse2.products RETAIN 168 HOURS;
			
			%%sql
			DESCRIBE HISTORY lakehouse2.products;
			
			df.write.format("delta").partitionBy("Category").saveAsTable("partitioned_products", path="abfs_path/partitioned_products")
			%%sql
			CREATE TABLE partitioned_products (
			ProductID INTEGER,
			ProductName STRING,
			Category STRING,
			ListPrice DOUBLE
			)
			PARTITIONED BY (Category);
			
			spark.sql("INSERT INTO products VALUES (1, 'Widget', 'Accessories', 2.99)")
			or
			%%sql
			UPDATE products
			SET Price = 2.49 WHERE ProductId = 1;
			
			Use the Delta API:
			from delta.tables import *
			from pyspark.sql.functions import *
			# Create a DeltaTable object
			delta_path = "Files/mytable"
			deltaTable = DeltaTable.forPath(spark, delta_path)
			# Update the table (reduce price of accessories by 10%)
			deltaTable.update(condition = "Category == 'Accessories'", set = { "Price": "Price * 0.9" })
			
			Use *time travel* to work with table versioning
			
			%%sql
			DESCRIBE HISTORY products (Table name or external path)
			
			df = spark.read.format("delta").option("versionAsOf", 0).load(delta_path)
			df = spark.read.format("delta").option("timestampAsOf", '2022-01-01').load(delta_path)
				Use delta tables with streaming data - Training | Microsoft Learn --> Read again
				
				Delta Lake table optimization and V-Order - Microsoft Fabric | Microsoft Learn --> Read again
				
				Using optimize write on Apache Spark to produce more efficient tables - Azure Synapse Analytics | Microsoft Learn
				Low Shuffle Merge optimization on Delta tables - Azure Synapse Analytics | Microsoft Learn
				Delta table maintenance in Microsoft Fabric - Microsoft Fabric | Microsoft Learn
				Compute management in Fabric environments - Microsoft Fabric | Microsoft Learn
				Apache Spark compute for Data Engineering and Data Science - Microsoft Fabric | Microsoft Learn
				
				Interesting but preview, won't be in exam: Native execution engine for Fabric Spark - Microsoft Fabric | Microsoft Learn

Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Ingest Data with Dataflows in Microsoft Fabric - Training | Microsoft Learn		Repeat from Learning Path 1	
Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Orchestrate processes and data movement with Microsoft Fabric - Training | Microsoft Learn	Redid lab part: 20 minutes with some error handling	Repeat from Learning Path 1 Note: Notebook parameterization is in this module!	
Implement a Lakehouse with Microsoft Fabric DP-601T00 - Training | Microsoft Learn	Organize a Fabric lakehouse using medallion architecture design - Training | Microsoft Learn	1 hour because of dimensional model load in the lab	Review pySpark syntax: from pyspark.sql.functions import when, lit, col, current_timestamp, input_file_name # Add columns IsFlagged, CreatedTS and ModifiedTS df = df.withColumn("FileName", input_file_name()) \ .withColumn("IsFlagged", when(col("OrderDate") < '2019-08-01',True).otherwise(False)) \ .withColumn("CreatedTS", current_timestamp()).withColumn("ModifiedTS", current_timestamp()) # Update CustomerName to "Unknown" if CustomerName null or empty df = df.withColumn("CustomerName", when((col("CustomerName").isNull() | (col("CustomerName")=="")),lit("Unknown")).otherwise(col("CustomerName"))) dfdimDate_gold = df.dropDuplicates(["OrderDate"]).select(col("OrderDate"), \ dayofmonth("OrderDate").alias("Day"), \ month("OrderDate").alias("Month"), \	Overview of Fabric Git integration - Microsoft Fabric | Microsoft Learn Review code in DP700Study_TransformDataForSilver for UPSERT statement Review lab : https://microsoftlearning.github.io/mslearn-fabric/Instructions/Labs/03b-medallion-lakehouse.html

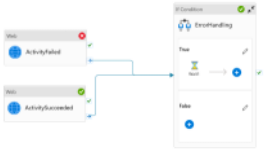
			`year("OrderDate").alias("Year"), \`
			`date_format(col("OrderDate"), "MMM-yyyy").alias("mmmyyyy"), \`
			`date_format(col("OrderDate"), "yyyyMM").alias("yyyymm"), \`
			`).orderBy("OrderDate")`
			
			monotonically_increasing_id()	

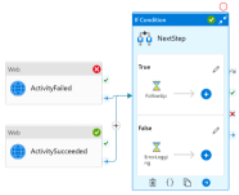
Implement a data warehouse with Microsoft Fabric DP-602T00 - Training | Microsoft Learn	Get started with data warehouses in Microsoft Fabric - Training | Microsoft Learn	15 minutes	Unit 6: Secure and monitor your data warehouse - Training | Microsoft Learn: Read: Allows the user to CONNECT using the SQL connection string. ReadData: Allows the user to read data from any table/view within the warehouse. ReadAll: Allows user to read data the raw parquet files in OneLake that can be consumed by Spark sys.dm_exec_connections: Returns information about each connection established between the warehouse and the engine. sys.dm_exec_sessions: Returns information about each session authenticated between the item and engine. sys.dm_exec_requests: Returns information about each active request in a session. KILL 'SESSION_ID WITH LONG-RUNNING QUERY'; Member, Contributor, and Viewer roles can see their own results within the warehouse, but cannot see other users' results.	Workspaces in Power BI - Power BI | Microsoft Learn Roles in workspaces in Power BI - Power BI | Microsoft Learn
Implement a data warehouse with Microsoft Fabric DP-602T00 - Training | Microsoft Learn	Load data into a Microsoft Fabric data warehouse - Training | Microsoft Learn	1 hour	Unit 2: Explore data load strategies - Training | Microsoft Learn Type 0 SCD: The dimension attributes never change. Type 1 SCD: Overwrites existing data, doesn't keep history. Type 2 SCD: Adds new records for changes, keeps full history for a given natural key. Type 3 SCD: History is added as a new column. Type 4 SCD: A new dimension is added. Type 5 SCD: When certain attributes of a large dimension change over time, but using type 2 isn't feasible due to the dimension's large size. Type 6 SCD: Combination of type 2 and type 3. Unit 4: https://learn.microsoft.com/en-us/training/modules/load-data-into-microsoft-fabric-data-warehouse/4-load-data-using-tsql COPY my_table FROM 'https://myaccount.blob.core.windows.net/myblobcontainer/folder0/*.csv, https://myaccount.blob.core.windows.net/myblobcontainer/folder1/' WITH (FILE_TYPE = 'CSV', CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET='<Your_SAS_Token>') FIELDTERMINATOR = '|') COPY INTO test_parquet FROM 'https://myaccount.blob.core.windows.net/myblobcontainer/folder1/*.parquet' WITH (CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET='<Your_SAS_Token>')) CREATE TABLE AS SELECT: Allows you to create a new table based on the output of a SELECT statement. This operation is often used for creating a copy of a table or for transforming and loading the results of complex queries. INSERT…SELECT Allows you to insert data from one table into another. It's useful when you want to copy data from one table to another without creating a new table. When working with external data on files, we recommend that files are at least 4 MB in size.	
Implement a data warehouse with Microsoft Fabric DP-602T00 - Training | Microsoft Learn	Query a data warehouse in Microsoft Fabric - Training | Microsoft Learn	5 minutes	SELECT ProductCategory, ProductName, ListPrice, ROW_NUMBER() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS RowNumber, RANK() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Rank, DENSE_RANK() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS DenseRank, NTILE(4) OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Quartile FROM dbo.DimProduct ORDER BY ProductCategory;	
Implement a data warehouse with Microsoft Fabric DP-602T00 - Training | Microsoft Learn	Monitor a Microsoft Fabric data warehouse - Training | Microsoft Learn	1 hour 15 minutes	In Spark, one CU translates to two spark vCores of compute. For example, when a customer purchases an F64 SKU, 128 spark vcores are available for Spark experiences.All Spark operations are background operations, and they're smoothed over a 24-hour period. You can view the number of executors allocated to a notebook in the Fabric monitoring hub KQL database CU consumption is calculated based on the number of seconds the database is active and the number of vCores used. For example, when your database uses four vCores and is active for 10 minutes, you'll consume 2,400 (4 x 10 x 60) seconds of CU. All KQL database operations are interactive operations. All Data Factory operations are considered background operations, and they're smoothed over a 24-hour period. "The first phase of throttling begins when a capacity has consumed all its available CU resources for the next 10 minutes. For example, if you purchased 10 units of capacity and then consumed 50 units per minute, you would create a carryforward of 40 units per minute. After two and a half minutes, you would have accumulated a carryforward of 100 units, borrowed from future windows. At this point where all capacity is already exhaustedfor the next 10 minutes, Fabric initiates its first level of throttling, and all new interactive operations are delayed by 20 seconds upon submission. If the carryforward reaches a full hour, interactive requests are rejected, but scheduled background operations continue to run. If the capacity accumulates a full 24 hours of carryforward, the entire capacity is frozen until the carryforward is paid off." In simple terms, 1 Fabric capacity unit = 0.5 Warehouse vCores. For example, a Fabric capacity SKU F64 has 64 capacity units, which is equivalent to 32 Warehouse vCores. From <https://learn.microsoft.com/en-us/fabric/data-warehouse/usage-reporting>	Search term "Fabric Capacity Metrics" in Learn to come to this page: Understand the metrics app compute page - Microsoft Fabric | Microsoft Learn Plan your capacity size - Microsoft Fabric | Microsoft Learn Metrics app calculations - Microsoft Fabric | Microsoft Learn **Evaluate and optimize your Microsoft Fabric capacity - Microsoft Fabric | Microsoft Learn** Service Overload Fabric Capacity Understand your Fabric capacity throttling - Microsoft Fabric | Microsoft Learn Data warehouse billing and utilization reporting - Microsoft Fabric | Microsoft Learn Monitor connections, sessions, and requests using DMVs - Microsoft Fabric | Microsoft Learn Query insights - Microsoft Fabric | Microsoft Learn

Implement a data warehouse with Microsoft Fabric DP-602T00 - Training | Microsoft Learn	Secure a Microsoft Fabric data warehouse - Training | Microsoft Learn	25 minutes	DDM
			-- For Email
			ALTER TABLE Customers
			ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()');
			
			-- For PhoneNumber
			ALTER TABLE Customers
			ALTER COLUMN PhoneNumber ADD MASKED WITH (FUNCTION = 'partial(3,"XXX-XXX-",4)');
			
			-- For CreditCardNumber
			ALTER TABLE Customers
			ALTER COLUMN CreditCardNumber ADD MASKED WITH (FUNCTION = 'partial(4,"XXXX-XXXX-XXXX-",4)');
			
			RLS
			
			--Create a schema
			CREATE SCHEMA [Sec];
			GO
			
			--Create the filter predicate
			CREATE FUNCTION sec.tvf_SecurityPredicatebyTenant(@TenantName AS NVARCHAR(10))
			RETURNS TABLE
			WITH SCHEMABINDING
			AS
			RETURN SELECT 1 AS result
			WHERE @TenantName = USER_NAME() OR USER_NAME() = 'tenantAdmin@contoso.com';
			GO
			
			--Create security policy and add the filter predicate
			CREATE SECURITY POLICY sec.SalesPolicy
			ADD FILTER PREDICATE sec.tvf_SecurityPredicatebyTenant(TenantName) ON [dbo].[Sales]
			WITH (STATE = ON);
			GO
			
			CLS
			-- Create roles
			CREATE ROLE Doctor AUTHORIZATION dbo;
			CREATE ROLE Nurse AUTHORIZATION dbo;
			CREATE ROLE Receptionist AUTHORIZATION dbo;
			CREATE ROLE Patient AUTHORIZATION dbo;
			GO
			
			-- Grant SELECT on all columns to all roles
			GRANT SELECT ON dbo.Patients TO Doctor;
			GRANT SELECT ON dbo.Patients TO Nurse;
			GRANT SELECT ON dbo.Patients TO Receptionist;
			GRANT SELECT ON dbo.Patients TO Patient;
			GO
			
			-- Deny SELECT on the MedicalHistory column to the Receptionist and Patient roles
			DENY SELECT ON dbo.Patients (MedicalHistory) TO Receptionist;
			DENY SELECT ON dbo.Patients (MedicalHistory) TO Patient;
			GO
			
			Always use parameterization methods like sp_executesql or QUOTENAME to sanitize inputs.
			
			From <https://learn.microsoft.com/en-us/training/modules/secure-data-warehouse-in-microsoft-fabric/5-configure-sql-granular-permissions>
			
			CREATE PROCEDURE sp_TopTenRows @tableName NVARCHAR(128)
			AS
			BEGIN
			DECLARE @query NVARCHAR(MAX);
			SET @query = N'SELECT TOP 10 * FROM ' + QUOTENAME(@tableName);
			EXEC sp_executesql @query;
			END;
			
			GRANT UNMASK ON dbo.Customers TO [<username>@<your_domain>.com];
			
			From <https://microsoftlearning.github.io/mslearn-fabric/Instructions/Labs/06d-secure-data-warehouse.html>
				No outside links

Manage a Microsoft Fabric environment - Training | Microsoft Learn	Implement continuous integration and continuous delivery (CI/CD) in Microsoft Fabric - Training | Microsoft Learn	+24 minutes + 18 minutes + 20 minutes		

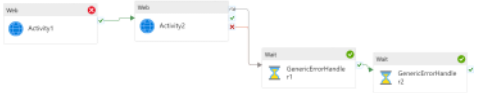
Manage a Microsoft Fabric environment - Training | Microsoft Learn	Monitor activities in Microsoft Fabric - Training | Microsoft Learn	+20 minutes +30 minutes	Column options in Monitor Hub: • Activity name • Status • Item type • Start time • Submitted by • Location • End time • Duration • Refresh type From <https://microsoftlearning.github.io/mslearn-fabric/Instructions/Labs/18-monitor-hub.html>	Activator tutorial using sample data - Microsoft Fabric | Microsoft Learn Apache Spark monitoring overview - Microsoft Fabric | Microsoft Learn
Manage a Microsoft Fabric environment - Training | Microsoft Learn	https://learn.microsoft.com/en-us/training/modules/secure-data-access-in-fabric/	30 minutes	Within each data item, *granular engine permissions* such as Read, ReadData, or ReadAll can be applied. Workspace roles can be assigned to individuals, security groups, Microsoft 365 groups, and distribution lists Admin - Can view, modify, share, and manage all content and data in the workspace, and manage permissions. Member - Can view, modify, and share all content and data in the workspace. Contributor - Can view and modify all content and data in the workspace. Viewer - Can view all content and data in the workspace, but can't modify it.	Roles in workspaces in Microsoft Fabric - Microsoft Fabric | Microsoft Learn Search Roles In Workspacess
Manage a Microsoft Fabric environment - Training | Microsoft Learn	Administer a Microsoft Fabric environment - Training | Microsoft Learn	+15 +15	Tenant is a dedicated space for organizations to create, store, and manage Fabric items. Capacity is a dedicated set of resources that is available at a given time to be used. Domain is a logical grouping of workspaces. Workspace is a collection of items that brings together different functionality in a single tenant.	
The rest of the items	Configure domain workspace settings			https://learn.microsoft.com/en-us/fabric/governance/domains#configure-domain-settings
	Configure data workflow workspace settings			Workspaces in Microsoft Fabric and Power BI - Microsoft Fabric | Microsoft Learn Configuring dataflow storage to use Azure Data Lake Gen 2 - Power BI | Microsoft Learn
	Implement database projects			https://learn.microsoft.com/en-us/fabric/data-warehouse/source-control#database-projects-for-a-warehouse-in-git Fabricators guide to database projects for Microsoft Fabric Data Warehouses - Kevin Chant Three ways to create a Microsoft Fabric Data Warehouse Database Project - Kevin Chant
	Apply sensitivity labels to items			Apply sensitivity labels to Fabric items - Microsoft Fabric | Microsoft Learn How to apply sensitivity labels in Power BI - Power BI | Microsoft Learn Enable sensitivity labels in Fabric - Power BI | Microsoft Learn
	Implement orchestration patterns with notebooks and pipelines, including parameters and dynamic expressions		You can use parameters to pass external values into pipelines. Once the parameter is passed into the resource, it can't be changed. @ is only removed if it is the first character. "@@" returns "@", " @" returns " @". String interpolation: The result is always string @{X} returns the value of X in string format. @{pipeline().parameters.firstName} "@pipeline().parameters.myNumber" Returns 42 as a number. "@{pipeline().parameters.myNumber}" Returns 42 as a string.	Parameters - Microsoft Fabric | Microsoft Learn Expressions and functions - Microsoft Fabric | Microsoft Learn Search for "dynamic expressions fabric pipelines" For Notebook parameters: Develop, execute, and manage notebooks - Microsoft Fabric | Microsoft Learn
	Design and implement full and incremental data loads		select * from data_source_table where LastModifytime > '@{activity('LookupOldWaterMarkActivity').output.firstRow.WatermarkValue}' and LastModifytime <= '@{activity('LookupNewWaterMarkActivity').output.firstRow.NewWatermarkvalue}' To incrementally copy files based on timestamp: In the Copy activity under Advanced choose Filter by Last modified: For every 5 minutes: @formatDateTime(addMinutes(pipeline().TriggerTime, -5), 'yyyy-MM-dd HH:mm:ss') For every x minutes: @formatDateTime(addMinutes(pipeline().TriggerTime, -<your set repeat minute>), 'yyyy-MM-dd HH:mm:ss') AddHours(..., x) AddDays(..., -1) AddDays(..., -7)	Incrementally load data from Data Warehouse to Lakehouse - Microsoft Fabric | Microsoft Learn Incrementally copy new and changed files based on the last modified date - Microsoft Fabric | Microsoft Learn
	Implement mirroring		To successfully configure Mirroring for Azure SQL Database, the principal used to connect to the source Azure SQL Database must be granted the permission ALTER ANY EXTERNAL MIRROR, which is included in higher level permission like CONTROL permission or the db_owner role. When mirroring data from Azure SQL Database or Azure SQL Managed Instance, its System Assigned Managed Identity needs to have"Read and write" permission to the mirrored database. If you create the mirrored database from the Fabric portal, the permission is granted automatically. By default, sharing a mirrored database grants users Read permission to the mirrored database, the associated SQL analytics endpoint, and the default semantic model. In addition to these default permissions, you can grant: Read all SQL analytics endpoint data, Read all OneLake data,Build reports on the default semantic model, Read and write. Currently, you must update your Azure SQL logical server firewall rules to Allow public network access. You must enable the Allow Azure services option to connect to your Azure SQL Database logical server. The SPN for Azure SQL DB Must have contributor role in the workspace that has the mirrored database.	Mirroring - Microsoft Fabric | Microsoft Learn Microsoft Fabric Mirrored Databases From Azure SQL Database - Microsoft Fabric | Microsoft Learn Tutorial: Configure Microsoft Fabric Mirrored Databases From Azure SQL Database - Microsoft Fabric | Microsoft Learn Limitations and Behaviors for Fabric Mirrored Databases From Azure SQL Database - Microsoft Fabric | Microsoft Learn Share Your Mirrored Database and Manage Permissions - Microsoft Fabric | Microsoft Learn
	Denormalize data		All that's known about the dimension member is its natural key. The fact load process needs to create a new dimension memberby using Unknown attribute values. Importantly, it must set the IsInferredMember audit attribute to TRUE. That way, when the late arriving details are sourced, the dimension load process can make the necessary updates to the dimension row. For more information, see Manage historical change in this article.	Modeling dimension tables in Warehouse - Microsoft Fabric | Microsoft Learn Modeling fact tables in Warehouse - Microsoft Fabric | Microsoft Learn Load tables in a dimensional model - Microsoft Fabric | Microsoft Learn
	Handle duplicate, missing, and late-arriving data			Load tables in a dimensional model - Microsoft Fabric | Microsoft Learn
	Optimize a pipeline		Fabric workspace admins can enable the high concurrency mode for pipelines using the workspace settings. Intelligent throughput optimization and Parallel copy. Staging is required when the Copy activity sink is Fabric Warehouse. Options such as Degree of copy parallelism and Intelligent throughput optimization only apply in that case from Source to Staging. Test cases to Lakehouse did not have staging enabled. Dynamic range with a Degree of parallel copies can significantly improve performance. Within the For-Each activity, all of the copy activities run in parallel (up to the batch count maximum of 50) and have degree of copy parallelism set to Auto. From <https://learn.microsoft.com/en-us/fabric/data-factory/copy-performance-sql-databases> Partition option: Specify the data partitioning options used to load data from Azure SQL Database. Allowed values are: None (default), Physical partitions of table, and Dynamic range. When a partition option is enabled (that is, not None), the degree of parallelism to concurrently load data from an Azure SQL Database is controlled by the parallel copy setting on the copy activity. It's recommended to leave Isolation level as None if you want to leave Degree of copy parallelism set to Auto. If the table has a physical partition, then using the Partition option *Physical partitions of table* would be the most balanced approach for transfer duration, capacity units, and compute overhead on the source. This setting is especially ideal if you have more sessions running against the database during the time of data movement. From <https://learn.microsoft.com/en-us/fabric/data-factory/copy-performance-sql-databases> Consider maintainability and developer effort. While leaving the default options take the longest time to move data, runningwith the defaults might be the best option, especially if the source table's DDL is unknown. This also provides reasonable Capacity Units consumption.	Copy activity performance with SQL databases - Microsoft Fabric | Microsoft Learn Configure high concurrency mode for notebooks in pipelines - Microsoft Fabric | Microsoft Learn Copy activity performance and scalability guide - Microsoft Fabric | Microsoft Learn
	Optimize a data warehouse		Consider updating column-level statistics regularly after data changes that significantly change rowcount or distribution of the data. Disabling V-Order can be useful for write-intensive warehouses, such as for warehouses that are dedicated to staging data as part of a data ingestion process. Group INSERT statements into batches (avoid trickle inserts) Consider using CTAS (Transact-SQL) to write the data you want to keep in a table rather than using DELETE. If a CTAS takes the same amount of time, it's safer to run since it has minimal transaction logging and can be canceled quickly if needed. Use integer-based data types if possible. SORT, JOIN, and GROUP BY operations complete faster on integers than on character data. There is no manual way to trigger data compaction. The Warehouse and SQL analytics endpoint have a user session limit of 724 per workspace. The Microsoft Fabric workspace provides a natural isolation boundary of the distributed compute system. OneLake shortcuts can be used to create read-only replicas of tables in other workspaces to distribute load across multiple SQL engines, creating an isolation boundary. This can effectively increase the maximum number of sessions performing read-only queries.	Statistics - Microsoft Fabric | Microsoft Learn Understand V-Order - Microsoft Fabric | Microsoft Learn Caching in Fabric data warehousing - Microsoft Fabric | Microsoft Learn Warehouse performance guidelines - Microsoft Fabric | Microsoft Learn https://blog.fabric.microsoft.com/en-US/blog/announcing-automatic-data-compaction-for-fabric-warehouse// https://learn.microsoft.com/en-us/fabric/data-warehouse/disable-v-order https://learn.microsoft.com/en-us/fabric/data-warehouse/v-order Workload management - Microsoft Fabric | Microsoft Learn
	Optimize Query Performance		SQL: Use Exist() instead of Count() Keep Wild cards at the End of Phrases -- SARGABLE SELECT * FROM TestTable WHERE DATEPART(YEAR, SomeMyDate) = '2021'; --> BAD Avoid using multiple OR in the FILTER predicate USE BETWEEN INSTEAD OF > AND <	SQL Query Optimization: 12 Useful Performance Tuning Tips and Techniques
	Choose between a pipeline and a notebook			Fabric decision guide - copy activity, dataflow, or Spark - Microsoft Fabric | Microsoft Learn
	Choose an appropriate data store			Fabric decision guide - choose a data store - Microsoft Fabric | Microsoft Learn
	Denormalize data			Modeling dimension tables in Warehouse - Microsoft Fabric | Microsoft Learn
	Optimize a data warehouse			Warehouse performance guidelines - Microsoft Fabric | Microsoft Learn
	Optimize query performance			SQL Query Optimization: 12 Useful Performance Tuning Tips and Techniques
	Pipeline error handling			Pipeline failure and error message - Azure Data Factory | Microsoft Learn Operationalize your Azure Data Factory or Azure Synapse Pipeline - Training | Microsoft Learn Monitor Azure Data Factory - Azure Data Factory | Microsoft Learn Add parameters to data factory components - Training | Microsoft Learn Dedupe rows and find nulls by using data flow snippets - Azure Data Factory | Microsoft Learn Orchestrating data movement and transformation in Azure Data Factory - Training | Microsoft Learn Mapping data flow script - Azure Data Factory | Microsoft Learn https://learn.microsoft.com/en-us/azure/data-factory/data-flow-script#distinct-row-using-all-columns





Generic error handling

Or



We determine pipeline success and failures as follows:

Evaluate outcome for all leaves activities. If a leaf activity was skipped, we evaluate its parent activity instead
Pipeline result is success if and only if all nodes evaluated succeed

After an activity ran and completed, you may reference its status with @activity('ActivityName').Status. It's either "Succeeded"_ or "Failed". We use this property to build conditional or logic.

@or(or(equals(activity('ActivityFailed').Status, 'Failed'), equals(activity('ActivitySucceeded1').Status, 'Failed')),equals(activity('ActivitySucceeded1').Status, 'Failed'))
@or(equals(activity('ActivityFailed').Status, 'Succeeded'), equals(activity('ActivitySucceeded').Status, 'Succeeded'))
@and(equals(activity('ActivityFailed').Status, 'Succeeded'), equals(activity('ActivitySucceeded').Status, 'Succeeded'))

The pattern is equivalent to try catch block in coding. For instance, I attempt to run a copy job, moving files into storage. However it might fail half way through. And in that case, I want to delete the partially copied, unreliable files from the storage account (my error handling step). But I'm OK to proceed with other activities afterwards.

	SQL Reference		Learn how to navigate to so that you can use this during the exam if needed.	Transact-SQL Reference (Database Engine) - SQL Server	Microsoft Learn
	Other DP700 exam videos		Will MJ be the first CERTIFIED Fabric Data Engineer? DP700  DP-700 Beta Exam Review: Tips to Pass and Become a Microsoft Certified Fabric Engineer