Insight Feed Filters — Appointments Dataset

How the Filter Works

When you set up an Insight Feed, the **Filter** field lets you limit which records are included in the emailed report. The filter is written in T-SQL and is applied to the **Appointments** dataset.

Enter only the condition — do not type the `WHERE` keyword.

Correct	`AppointmentType = 'Service Call'`
Incorrect	`WHERE AppointmentType = 'Service Call'`

Leave the filter blank to include every record in the dataset.

Quick Rules

1. **No `WHERE`, no `ORDER BY`, no `SELECT`.** Just the condition.
2. **Text values go in single quotes:** `'Completed'`, not `"Completed"`.
3. **Apostrophes inside text are doubled:** `CustomerGroup = 'O''Brien Properties'`.
4. **Use relative dates, not hard-coded dates.** A feed that runs every morning should say "today," not `'2026-07-31'`. Use `GETDATE()` based expressions so the filter stays correct on every run.
5. **Combine conditions with `AND` / `OR`**, and use parentheses whenever you mix them.
6. **Bracket column names that are also SQL keywords:** `[Exception]`, `[Type]`, `[Group]`, `[Window]`, `[Day]`, `[Stop]`, `[Duration]`.
7. **Blank values are `NULL`**, not empty text. Use `IS NULL` / `IS NOT NULL` rather than `= ''`.
8. Text comparisons are **not case sensitive**, so `'service call'` and `'Service Call'` both match. Spelling and spacing still must match your setup exactly.

The Four Core Examples

1. All Service Call appointments

```
AppointmentType = 'Service Call'
```

2. Installation appointments scheduled for today

```
AppointmentType = 'Installation'
AND AppointmentDate = CAST(GETDATE() AS DATE)
```

3. Appointments marked Completed or Exception, this week

```
AppointmentStatus IN ('Completed', 'Exception')
AND AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()), -1)
AND AppointmentDate <  DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 1, -1)
```

This treats the week as Sunday through Saturday. See the "Week Ranges" section below for a Monday-through-Sunday version.

`AppointmentStatus` and `[Exception]` are two different fields. The filter above matches the appointment **status**. If you also want the exception reason to be filled in, add a line such as:

```
AND [Exception] IS NOT NULL
```

4. Scheduled appointments for next week

```
AppointmentStatus = 'Scheduled'
AND AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 1, -1)
AND AppointmentDate <  DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 2, -1)
```

Date Filters

`AppointmentDate` holds the appointment's date with no time component, which makes it the easiest column to filter on. `StartDateTime` and `EndDateTime` include the time of day.

Single days

Goal	Filter
Today	`AppointmentDate = CAST(GETDATE() AS DATE)`
Tomorrow	`AppointmentDate = CAST(DATEADD(DAY, 1, GETDATE()) AS DATE)`
Yesterday	`AppointmentDate = CAST(DATEADD(DAY, -1, GETDATE()) AS DATE)`

Rolling ranges

Next 7 days, including today:

```
AppointmentDate >= CAST(GETDATE() AS DATE)
AND AppointmentDate < CAST(DATEADD(DAY, 7, GETDATE()) AS DATE)
```

Previous 30 days:

```
AppointmentDate >= CAST(DATEADD(DAY, -30, GETDATE()) AS DATE)
AND AppointmentDate <= CAST(GETDATE() AS DATE)
```

Week ranges

Weeks running Sunday through Saturday

This week

```
AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()), -1)
AND AppointmentDate <  DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 1, -1)
```

Next week

```
AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 1, -1)
AND AppointmentDate <  DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) + 2, -1)
```

Last week

```
AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) - 1, -1)
AND AppointmentDate <  DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()), -1)
```

The pattern is always the same: change the number added to `DATEDIFF(WEEK, 0, GETDATE())` to shift the range. Use `+ 2` and `+ 3` for the week after next, `- 2` and `- 1` for two weeks ago, and so on.

Weeks running Monday through Sunday

This week

```
AppointmentDate >= DATEADD(DAY, -(DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
AND AppointmentDate <  DATEADD(DAY, 7 - (DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
```

Next week

```
AppointmentDate >= DATEADD(DAY, 7 - (DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
AND AppointmentDate <  DATEADD(DAY, 14 - (DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
```

Last week

```
AppointmentDate >= DATEADD(DAY, -7 - (DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
AND AppointmentDate <  DATEADD(DAY, -(DATEDIFF(DAY, 0, GETDATE()) % 7), CAST(GETDATE() AS DATE))
```

Here the pattern is to shift both day offsets by 7 — add 7 to move a week forward, subtract 7 to move a week back.

Month ranges

Goal	Filter
This month	`AppointmentDate >= DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1) AND AppointmentDate < DATEADD(MONTH, 1, DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1))`
Last month	`AppointmentDate >= DATEADD(MONTH, -1, DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1)) AND AppointmentDate < DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1)`
Year to date	`AppointmentDate >= DATEFROMPARTS(YEAR(GETDATE()), 1, 1) AND AppointmentDate <= CAST(GETDATE() AS DATE)`

Time of day

Morning appointments today (before noon):

```
AppointmentDate = CAST(GETDATE() AS DATE)
AND DATEPART(HOUR, StartDateTime) < 12
```

Weekend appointments in the next 30 days:

```
DATENAME(WEEKDAY, AppointmentDate) IN ('Saturday', 'Sunday')
AND AppointmentDate BETWEEN CAST(GETDATE() AS DATE) AND CAST(DATEADD(DAY, 30, GETDATE()) AS DATE)
```

A specific fixed date range

Use this only for one-time or historical reports, since the dates do not move forward:

```
AppointmentDate >= '2026-01-01' AND AppointmentDate < '2026-04-01'
```

Appointment Filters

Several appointment types at once

```
AppointmentType IN ('Service Call', 'Installation', 'Maintenance')
```

Everything except cancelled work

```
AppointmentStatus NOT IN ('Cancelled', 'Rescheduled')
```

Any appointment with an exception

```
[Exception] IS NOT NULL
```

Unassigned appointments in the next two weeks — nothing in the technician field and no crew:

```
AssignedTo IS NULL
AND Crew IS NULL
AND AppointmentDate >= CAST(GETDATE() AS DATE)
AND AppointmentDate < CAST(DATEADD(DAY, 14, GETDATE()) AS DATE)
```

One technician's schedule

```
AssignedTo = 'Mike Sanders'
AND AppointmentDate >= CAST(GETDATE() AS DATE)
```

A technician who may be the assignee *or* a helper on the appointment

```
(AssignedTo = 'Mike Sanders' OR AppointmentAdditionalEmployees LIKE '%Mike Sanders%')
```

A specific crew

```
Crew = 'Install Team A'
```

A specific route and day

```
Route = 'North Route'
AND [Day] = 'Monday'
```

Appointments carrying a tag

```
AppointmentTags LIKE '%Warranty%'
```

Appointments with a keyword in the description or notes

```
(AppointmentDescription LIKE '%furnace%' OR AppointmentNotes LIKE '%furnace%')
```

Long appointments

```
[Duration] >= 240
```

Priority work still open

```
Priority = 'High'
AND AppointmentStatus NOT IN ('Completed', 'Cancelled')
```

Customer & Location Filters

One customer

```
FullName = 'ABC Plumbing'
```

The name has to match what is stored on the customer record exactly, apart from capitalization. If the name contains an apostrophe, double it — `FullName = 'O''Brien Plumbing'`.

Customers in specific cities

```
ServiceAddressCity IN ('Marion', 'Delaware', 'Columbus')
```

By state

```
ServiceAddressState = 'OH'
```

By ZIP code prefix

```
ServiceAddressZipCode LIKE '433%'
```

One sales rep's customers

```
SalesRep = 'Jane Doe'
```

Active customers only

```
Active = 1
```

By customer group, type, or tag

```
CustomerGroup = 'Commercial'
```

```
[Type] = 'Residential'
```

```
CustomerTags LIKE '%VIP%'
```

Customers who came from a particular source

```
Source = 'Google'
```

Customers with an alert message on file

```
AlertMessage IS NOT NULL
```

Customers with an expiring payment profile

```
PrimaryProfileExpDate IS NOT NULL
AND PrimaryProfileExpDate < DATEADD(DAY, 60, GETDATE())
```

By company or division (for multi-company databases)

```
DBA = 'Acme Heating & Cooling'
```

By rollup grouping

```
RollupGrouping1 = 'Midwest'
```

Work Order, Quote & Invoice Filters

Appointments tied to an open work order

```
WorkOrderStatus = 'Open'
```

Recurring maintenance work

```
WorkOrderRecurrenceType IS NOT NULL
```

A specific work order number

```
WorkOrderNumber = '10045'
```

Completed appointments not yet invoiced

```
AppointmentStatus = 'Completed'
AND InvoiceID IS NULL
AND AppointmentDate >= CAST(DATEADD(DAY, -30, GETDATE()) AS DATE)
```

Invoiced appointments with no payment applied

```
InvoiceID IS NOT NULL
AND AppliedPaymentIDs IS NULL
```

Invoices not yet synced to QuickBooks

```
InvoiceID IS NOT NULL
AND QuickBooksInvoiceID IS NULL
```

High-value work orders

```
WorkOrderRevenue >= 5000
```

Low-margin completed jobs

```
AppointmentStatus = 'Completed'
AND WorkOrderProfitability IS NOT NULL
AND WorkOrderProfitability < 100
```

Underperforming production hours

```
RevPerManHourProduction IS NOT NULL
AND RevPerManHourProduction < 75
AND AppointmentDate >= DATEADD(WEEK, DATEDIFF(WEEK, 0, GETDATE()) - 1, 0)
```

Appointments with an open quote attached

```
QuoteID IS NOT NULL
AND InvoiceID IS NULL
```

Quotes above a threshold

```
QuoteRevenue > 10000
```

Line items containing a particular item

```
WorkOrderItems LIKE '%Filter Replacement%'
```

Jobs where labor was recorded

```
TotalWorkTime > 0
```

Jobs where the crew logged more travel than work

```
TotalTravelTime > TotalWorkTime
```

Combining Conditions

Use `AND` when every condition must be true, `OR` when any one is enough. When you mix them, wrap the `OR` group in parentheses — otherwise you will get more records than you expect.

Correct — completed *or* invoiced, but only for this month:

```
(AppointmentStatus = 'Completed' OR InvoiceID IS NOT NULL)
AND AppointmentDate >= DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1)
```

Incorrect — the date limit only applies to the second half, so every completed appointment in history comes through:

```
AppointmentStatus = 'Completed' OR InvoiceID IS NOT NULL
AND AppointmentDate >= DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), 1)
```

A full multi-part example — tomorrow's commercial installs for a specific crew, excluding cancellations:

```
AppointmentDate = CAST(DATEADD(DAY, 1, GETDATE()) AS DATE)
AND AppointmentType = 'Installation'
AND CustomerGroup = 'Commercial'
AND Crew = 'Install Team A'
AND AppointmentStatus <> 'Cancelled'
```

Wildcards with LIKE

Pattern	Matches
`LIKE 'Serv%'`	Values that start with *Serv*
`LIKE '%Call'`	Values that end with *Call*
`LIKE '%Service%'`	Values that contain *Service* anywhere
`NOT LIKE '%Test%'`	Values that do not contain *Test*

`LIKE` with a leading `%` is the right choice for the combined columns such as `AppointmentTags`, `CustomerTags`, `AppointmentCrewMembers`, `AppointmentAdditionalEmployees`, `WorkOrderItems`, `QuoteItems`, and `InvoiceItems`, because those columns hold several values in a single field.

Troubleshooting

The feed came back empty. Check your spelling against the actual values in Smart Service — a status of `'Complete'` will not match a stored value of `'Completed'`. Try loosening the filter to one condition at a time to find the one that is excluding everything.

A filter on a blank field returns nothing. Empty fields are `NULL`. `[Exception] = ''` matches nothing; `[Exception] IS NULL` is what you want. Likewise, `<> 'Cancelled'` will *exclude* rows where the status is blank, since `NULL` is neither equal nor unequal to anything. If blanks should be included, write `(AppointmentStatus <> 'Cancelled' OR AppointmentStatus IS NULL)`.

The feed errors out. The most common causes are a stray `WHERE` at the front, a mismatched parenthesis, a double-quoted string, or an unbracketed keyword column such as `Exception`. Also confirm the column name actually exists in this dataset — column names must match the list below exactly.

Results were right last month but are wrong now. The filter probably contains hard-coded dates. Switch to a `GETDATE()` expression from the Date Filters section.

Commonly Used Columns

Appointment: `AppointmentID`, `AppointmentDescription`, `AppointmentDate`, `StartDateTime`, `EndDateTime`, `AppointmentType`, `AppointmentTags`, `AppointmentStatus`, `[Exception]`, `AssignedTo`, `Crew`, `AppointmentAdditionalEmployees`, `AppointmentCrewMembers`, `[Window]`, `PreNotification`, `Priority`, `Route`, `[Day]`, `[Stop]`, `[Duration]`, `AppointmentNotes`

Time & productivity: `TotalTravelTime`, `TotalWorkTime`, `TotalTime`, `EmployeeCount`, `DirectLaborCost`, `RevPerManHourTravel`, `RevPerManHourProduction`, `RevPerManHourTotal`

Quote: `QuoteID`, `QuoteNumber`, `QuoteDescription`, `QuoteType`, `QuoteTags`, `QuoteRevenue`, `QuoteProfitability`, `QuoteItems`, `QuoteNotes`, `QuoteUser01` – `QuoteUser10`

Work order: `WorkOrderID`, `WorkOrderNumber`, `WorkOrderDescription`, `WorkOrderType`, `WorkOrderTags`, `WorkOrderStatus`, `WorkOrderSubStatus`, `WorkOrderRecurrenceType`, `WorkOrderRecurrencePattern`, `WorkOrderRevenue`, `WorkOrderProfitability`, `WorkOrderItems`, `WorkOrderNotes`, `WorkOrderUser01` – `WorkOrderUser10`

Invoice & payment: `InvoiceID`, `InvoiceNumber`, `InvoiceDescription`, `InvoiceType`, `InvoiceRevenue`, `InvoiceProfitability`, `InvoiceItems`, `InvoiceNotes`, `QuickBooksInvoiceID`, `AppliedPaymentIDs`

Address: `ServiceAddressLine1`, `ServiceAddressCity`, `ServiceAddressState`, `ServiceAddressZipCode`, `ServiceAddressCounty`, `Latitude`, `Longitude`, and the matching `BillingAddress…` columns

Customer: `CustomerID`, `FullName`, `Company`, `FirstName`, `LastName`, `PrimaryPhone`, `Mobile`, `PrimaryEmail`, `SalesRep`, `Source`, `Terms`, `PaymentMethod`, `CustomerSince`, `Active`, `[Type]`, `CustomerGroup`, `CustomerTags`, `Notes`, `User01` – `User10`, `AlertMessage`, `PrimaryProfileExpDate`, `PrimaryProfileStatus`, `PricingOptionName`, `Class`, `SalesTax`, `SalesTaxRate`, `DBA`

Grouping & identifiers: `RollupGrouping1`, `RollupGrouping2`, `RollupGrouping3`, `DatabaseName`, `Dataset`, `EnterpriseAppointmentID`, `EnterpriseCustomerID`, `EnterpriseWorkOrderID`, `EnterpriseQuoteID`, `EnterpriseInvoiceID`