author: sfc-gh-drichert id: servicenow\_to\_snowflake\_connector summary: Step-by-step to set up Servicenow connector categories: Connectors environments: web status: Private feedback link: https://github.com/Snowflake-Labs/sfguides/issues tags: Connectors, Data Engineering, Servicenow

# Snowflake Connector for Servicenow Installation

## Overview

Duration: 1

Ingest data from ServiceNow into Snowflake automatically. The connector supports both the initial load of historical data as well as incremental updates. The latest data is regularly pulled from ServiceNow and you control how frequently it is refreshed.

Use this quickstart to configure and understand the Snowflake Connector for Servicenow using the Snowsight wizard, select some tables, ingest data, run some typical usage queries. When you are done stop the connector to avoid costs. You could also do all these steps programmatically, please refer to the documentation.

Note: This quickstart assumes you do not have a Servicenow account, so it guides you through the steps of creating a developer account. Of course, if you do have a Servicenow account, please feel free to try it out, with the caveat that, at the time of writing, the connector is in public preview and should not be used for production.

### Prerequisites

* Servicenow account with administrator’s rights.
* Snowflake account and user with accountadmin’s role.
* Accepting the Terms of Service in the Snowflake Marketplace. ORGADMIN rights are required for this.

### What You’ll Learn

* How to set up the Snowflake Servicenow connector.
* How to ingest table data.
* How to stop the connector to avoid unnecessary costs in a development environment.

### What You’ll Need

* A [Snowflake](https://snowflake.com/) Account
* A [Servicenow](https://developer.servicenow.com/dev.do/) developer account

### What You’ll Build

* A Servicenow to Snowflake ingestion data flow.

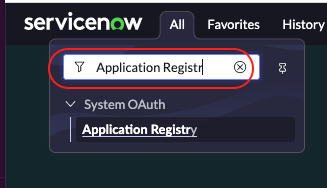
## Servicenow Setup

Duration: 30

1. Go to the [Servicenow developer website](https://developer.servicenow.com), and create a developer user.
2. Log on to the developer website with your newly created user and select **Create an Instance**.
3. Choose an instance type. You receive an email with your instance URL, and admin user and password.

## Servicenow endpoint configuration

The Servicenow endpoint configuration window creates an OAuth client application record and generates a client ID and client secret that Snowflake needs to access the restricted resources on the instance.

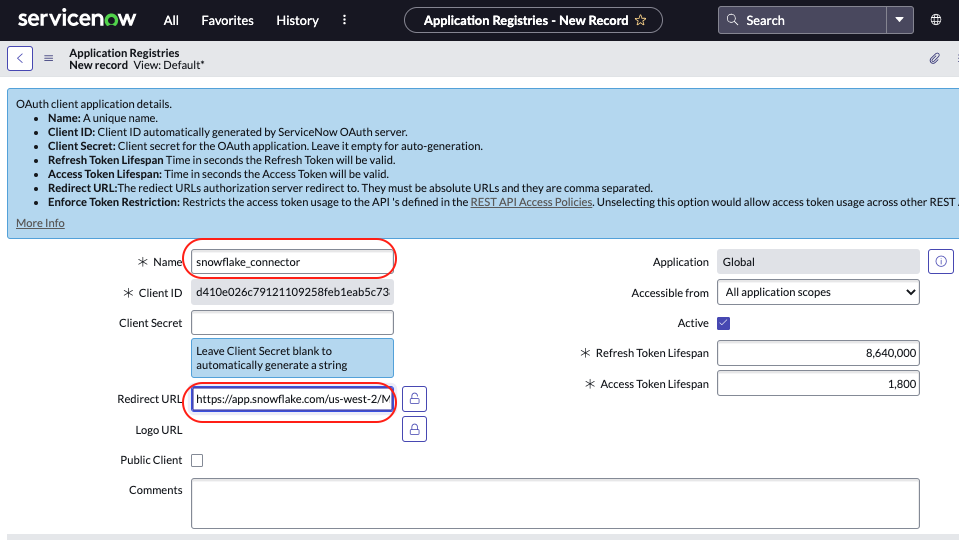
1. Log on to your Servicenow developer instance.
2. From the main page, select **All** and search **Application Registry,** for example this is what it looks like on the Tokyo version. 
3. Select **New** in the upper right-hand side of the window.
4. Select **Create an OAuth API endpoint for external clients**.
5. Give the endpoint a name, such as **Snowflake\_connector**. Leave the client secret blank. This will autofill when you select **Submit** later in the procedure.
6. Fill in the redirect URL with this syntax (Alternatively, Snowflake will generate this in a later step and you can come back and modify the redirect URL).

https:*//apps-api.c1.<cloud\_region\_id>.<cloud>.app.snowflake.com/oauth/complete-secret*

where - **cloud\_region\_id** can be found in the URL of Snowsight, for example:

https://app.snowflake.com/**us-west-2**/MyAccountId/worksheets

* and **cloud** is aws or azure or gcp.

For example, for AWS US WEST 2 would be: javascript https://apps-api.c1.us-west-2.aws.app.snowflake.com/oauth/complete-secret  Select **Submit**.

## Snowflake Configuration

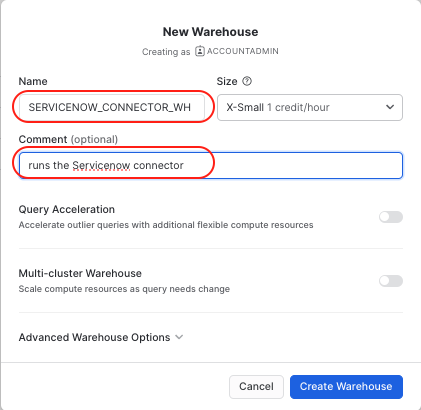
Duration: 10

### Accept Terms & Conditions

1. Log on to your Snowflake account through the Snowsight web interface and change to the **orgadmin** role.
2. Select “Admin » Billing & Terms”.
3. In the “Snowflake Marketplace” section, review the Consumer Terms of Service.
4. If you agree to the terms, select “Accept Terms & Conditions”.

### Set Up Two Virtual Warehouses

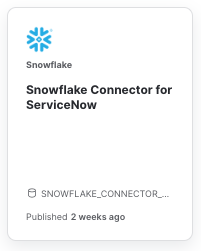
Change to the **accountadmin** role.

1. Navigate to Admin -> Warehouses and select **+ Warehouse**.
2. Name the first vitural warehosue **SERVICENOW\_CONNECTOR\_WH** and, leaving the defaults, select **Create Warehouse**. 
3. Repeat the above two steps to create a second virtual warehouse **SERVICENOW\_WAREHOUSE**.

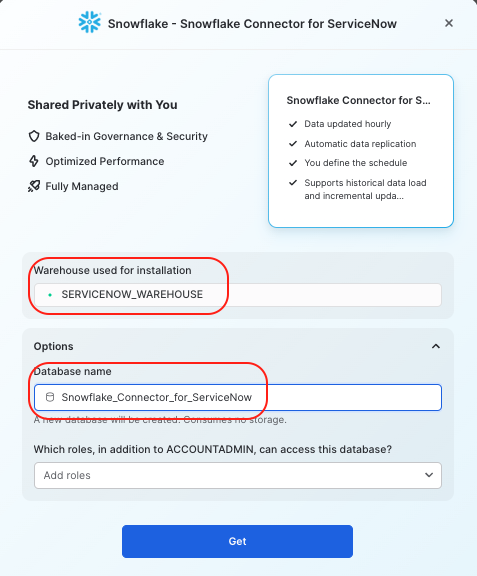
## Get the Servicenow connector

The connector is delivered through the Snowflake native application framework into your account as a database with a couple of schemas, tables, views, and stored procedures.

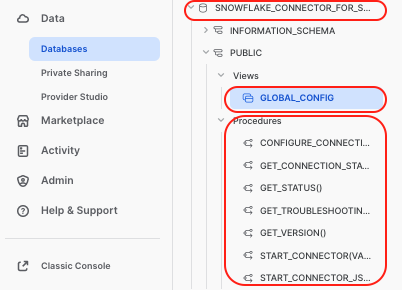
1. From the Snowflake Account Home page, select Marketplace.
2. In the search window, enter **servicenow**. The tile appears:



*Tile*

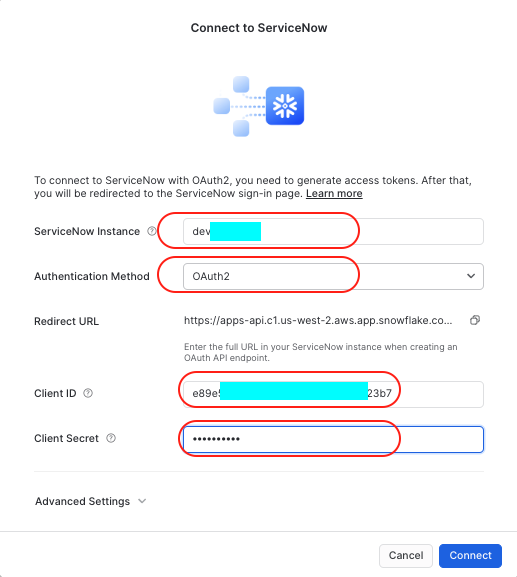
1. Select the **Snowflake Connector for ServiceNow**.
2. Review the business needs and usage samples. Hint: If you want to access the examples after configuring the connector, simply choose a role different from accountadmin.
3. Select **Get**.
4. Select the warehouse you created above, **SERVICENOW\_WAREHOUSE**.
5. For this quickstart, leave the default name for the installation database. The screen should look like the following: 
6. After reading the small print on the bottom of the screen, select **Get**. After 10-20 seconds, you receive the following message, **Snowflake Connector for ServiceNow is now ready to use in your account.**
7. Select **Done**.

If you would like to verify the connector was installed, from Snowsight, you can go to **Data -> Databases**. You will see a new database with the name **SNOWFLAKE\_CONNECTOR\_FOR\_SERVICENOW**. Open the Public schema and views to see the Global\_Config view. Some of the Procedures have also been installed. Others will appear after the installation finishes.



*installed*

## Connect Snowflake to Servicenow

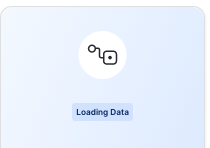
1. From the Snowflake Account Home page, select Marketplace and then select the **Snowflake Connector for Servicenow** tile.
2. In the **Snowflake Connector for ServiceNow** window, select **Manage**.
3. Select **Connect**.
4. Fill in the Servicenow instance details. This is the first part of the Servicenow URL for your Servicenow account, **without** the trailing *service-now.com*.
5. Select **OAuth2** for the Authentication method.
6. Enter the **Client id** from Servicenow that was generated in the Servicenow endpoint configuration.
7. Copy the Client secret from Servicenow and into the Snowflake configure pop-up. *Hint: on ServiceNow unlock the field by clicking on the lock, and then copy the text to Snowflake to make sure you are actually copying the right text.* The screen should look something similar to this: 
8. Select **Connect**. Your Servicenow accounts pops up and requests to connect to Snowflake.
9. Select **Allow**. The connection is established between the two systems.

To verify the connection, select the three dots […] and **View Details**. At the top of the pop-up you will see when **ServiceNow** Authenticated. ## Select Servicenow Tables

1. In Snowsight, select the **Snowflake Connector for Servicenow** tile.
2. In the **Snowflake Connector for ServiceNow** window, select **Select Tables**.
3. From the search window enter **incident** and check the box next to it and choose a 30 minute sync time. **Do not start the ingestion yet!**
4. To choose other tables, clear the search, put the table name and select the checkbox. Do this for the following tables:

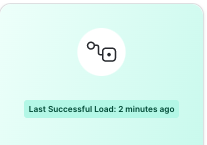
* SYS\_CHOICE
* SYS\_USER
* SYS\_USER\_GROUP
* TASK
* Hint: Select Field title “Status” to sort and show all the tables you selected.

1. Select **Configure** and review the default values for destinations and schemas, roles, a secondary warehouse and journal table.
2. Now select **Start Ingestion**. The select window closes and you get the message “Loading Data” from the main Connector window.



*load*

and, depending on the load time, a success message follows.



*success*

## Connector Monitoring (Query Sync History)

In the connector interface, choose **Query Sync History.** A worksheet opens with several SQL queries you can execute to get monitoring information.

## Setting Permissions to Read

Once you have ingested some data, you probably want to access it. Use the following SQL to create the **servicenow\_reader\_role** and give it the right access.

**USE ROLE** accountadmin; **CREATE** **ROLE** servicenow\_reader\_role;  
**GRANT** **USAGE** **ON** **DATABASE** SERVICENOW\_DEST\_DB **TO** **ROLE** servicenow\_reader\_role;  
**GRANT** **USAGE** **ON** **SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA **TO** **ROLE** servicenow\_reader\_role;

**GRANT** **USAGE** **ON** **WAREHOUSE** SERVICENOW\_WH **TO** **ROLE** servicenow\_reader\_role;

**GRANT** **SELECT** **ON** FUTURE **TABLES** **IN** **SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA **TO** **ROLE** servicenow\_reader\_role;  
**GRANT** **SELECT** **ON** FUTURE VIEWS **IN** **SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA **TO** **ROLE** servicenow\_reader\_role;  
**GRANT** **SELECT** **ON** **ALL** **TABLES** **IN** **SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA **TO** **ROLE** servicenow\_reader\_role;  
**GRANT** **SELECT** **ON** **ALL** VIEWS **IN** **SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA **TO** **ROLE** servicenow\_reader\_role;

**USE ROLE** servicenow\_reader\_role; **USE SCHEMA** SERVICENOW\_DEST\_DB.DEST\_SCHEMA; **WITH** T1 **AS** (  
 **SELECT**  
 **DISTINCT**  
 T.NUMBER **AS** TICKET\_NUMBER  
 ,G1.NAME **AS** PARENT\_ASSIGNMENT\_GROUP  
 ,G.NAME **AS** CHILD\_ASSIGNMENT\_GROUP  
 ,T.SHORT\_DESCRIPTION  
 ,T.DESCRIPTION  
 ,CI.NAME **AS** CONFIGURATION\_ITEM  
 ,SC\_CAT.**LABEL** **AS** **CATEGORY**  
 ,SC\_SUBCAT.**LABEL** **AS** SUBCATEGORY  
 ,T.PRIORITY  
 ,T.SYS\_CREATED\_ON **AS** CREATED\_ON  
 ,SU.NAME **AS** ASSIGNED\_TO  
 ,SU1.NAME **AS** OPENED\_BY  
 ,U2.NAME **AS** INCIDENT\_REQUESTED\_FOR  
 ,T.SYS\_UPDATED\_ON **AS** UPDATED\_ON  
 ,T.CLOSED\_AT  
  
 **FROM**  
 TASK\_\_VIEW T  
 **LEFT** **JOIN**   
 INCIDENT\_\_VIEW I   
 **ON** I.SYS\_ID = T.SYS\_ID *-- ADDITIONAL INCIDENT DETAIL*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM**   
 SYS\_CHOICE\_\_VIEW SC\_CAT   
 **WHERE**   
 ELEMENT = 'U\_T\_CATEGORY'  
 ) SC\_CAT  
 **ON** T.U\_T\_CATEGORY = SC\_CAT.VALUE *-- MAPPING FOR CATEGORY VALUES FROM* **TASK** *TABLE*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM**   
 SYS\_CHOICE\_\_VIEW   
 **WHERE**   
 ELEMENT = 'U\_T\_SUBCATEGORY'   
 **AND** NAME ='SC\_REQ\_ITEM'  
 ) SC\_SUBCAT   
 **ON** T.U\_T\_SUBCATEGORY = SC\_SUBCAT.VALUE *-- MAPPING FOR SUBCATEGORY VALUES FROM* **TASK** *TABLE*  
 **LEFT** **JOIN**   
 CMDB\_CI\_\_VIEW CI   
 **ON** T.CMDB\_CI\_VALUE = CI.SYS\_ID *-- CONFIGURATION ITEM OR APPLICATION NAME*  
 **LEFT** **JOIN**   
 SC\_REQ\_ITEM\_\_VIEW R   
 **ON** T.SYS\_ID = R.SYS\_ID *-- RITM OR SERVICE REQUEST INFORMATION*  
 **LEFT** **JOIN**   
 SC\_REQUEST\_\_VIEW SR   
 **ON** R.REQUEST\_VALUE = SR.SYS\_ID *-- RITM REQUESTED FOR INFORMATION*  
 **LEFT** **JOIN**   
 SYS\_USER\_\_VIEW SU   
 **ON** T.ASSIGNED\_TO\_VALUE = SU.SYS\_ID *-- ASSIGNED TO USERS NAME*  
 **LEFT** **JOIN**   
 SYS\_USER\_\_VIEW SU1   
 **ON** T.OPENED\_BY\_VALUE = SU1.SYS\_ID *-- OPENED BY USERS NAME*  
 **LEFT** **JOIN**   
 SYS\_USER\_\_VIEW U2   
 **ON** I.CALLER\_ID\_VALUE = U2.SYS\_ID *-- INCIDENT REQUESTED FOR NAME*  
 **LEFT** **JOIN**   
 SYS\_USER\_GROUP\_\_VIEW G   
 **ON** NVL(T.ASSIGNMENT\_GROUP\_VALUE, T.ASSIGNMENT\_GROUP) = G.SYS\_ID *-- CHILD GROUP NAME*  
 **LEFT** **JOIN**   
 SYS\_USER\_GROUP\_\_VIEW G1   
 **ON** NVL(G.PARENT\_VALUE, G.**PARENT**) = G1.SYS\_ID *-- PARENT GROUPS*  
 **LEFT** **JOIN**   
 SYS\_AUDIT\_DELETE\_\_VIEW DEL   
 **ON** T.SYS\_ID = DEL.DOCUMENTKEY *-- THIS JOIN HELPS IDENTIFY DELETED TICKETS*  
  
 **WHERE**  
 DEL.DOCUMENTKEY **IS** **NULL** *-- THIS CONDITION HELPS KEEP ALL DELETED RECORDS OUT*  
 **AND**  
 I.SYS\_ID **IS** **NOT** **NULL** *-- THIS CONDITION HELPS KEEP JUST THE INCIDENT TICKETS*  
)  
**SELECT**  
 YEAR(CREATED\_ON) **AS** YEAR\_CREATED  
 ,MONTH(CREATED\_ON) **AS** MONTH\_CREATED  
 ,CONFIGURATION\_ITEM **AS** APPLICATION  
 ,PRIORITY  
 ,COUNT(**DISTINCT** TICKET\_NUMBER)  
**FROM**  
 T1  
**GROUP** **BY**  
 YEAR\_CREATED  
 ,MONTH\_CREATED  
 ,APPLICATION  
 ,PRIORITY  
**ORDER** **BY**  
 YEAR\_CREATED  
 ,MONTH\_CREATED  
 ,APPLICATION  
 ,PRIORITY  
;  
  
  
## **Stop** **the** Ingestion  
> aside positive  
> **If** you do **not** **stop** **the** connector, it will wake up **the** virtual warehouse **at** **the** specified time interval **and** consume credits.  
  
  
1. **In** Snowsight, **select** **the** \*\*Snowflake Connector **for** Servicenow\*\* tile.  
  
1. **In** **the** \*\*Snowflake Connector **for** ServiceNow\*\* window, **select** \*\***Stop** Ingestion\*\*.  
  
  
**Read** **the** warning **and** **select** \*\***Stop** Ingestion\*\*.  
  
  
## **Delete** **the** Connector (but **not** **the** **data**)  
**To** **delete** **the** connector you need **to** **drop** **the** connector **database**:   
```SQL  
**DROP** **DATABASE** SNOWFLAKE\_CONNECTOR\_FOR\_SERVICENOW;

## Conclusion

Duration: 1

Upon successful completion of this Quickstart you were able to setup the Servicenow connector!

## Optional: Query Examples

### CMDB applications count

The CMDB (Configuration Management Database) is the ServiceNow database that stores information about all technical services. Within the CMDB, the support information for each service offering is stored in a Configuration Item (CI) specific to that service. This query provides the CMDB applications count by department, assignment groups, application owner, vendor or their respective status.

Tables to replicate to run this example: CMDB\_CI\_BUSINESS\_APP CMN\_DEPARTMENT CORE\_COMPANY SYS\_USER\_GROUP SYS\_USER SYS\_AUDIT\_DELETE

to add: CALL ENABLE\_TABLES

**WITH** T1 **AS**(  
 **SELECT**  
 **DISTINCT**   
 B.NAME **AS** DEPARTMENT  
 ,D.NAME **AS** ASSIGNMENT\_GROUP  
 ,A.NAME **AS** BUSINESS\_APP\_NAME  
 ,E.NAME **AS** APP\_OWNER  
 ,A.BUSINESS\_CRITICALITY  
 ,C.NAME **AS** VENDOR\_NAME  
 ,C.STATUS **AS** VENDOR\_STATUS  
 **FROM**  
 CMDB\_CI\_BUSINESS\_APP\_\_VIEW A *-- THIS TABLE INCLUDES ALL THE BUSINESS APPS THAT ARE CONFIGURED*  
 **LEFT** **JOIN**  
 CMN\_DEPARTMENT\_\_VIEW B *-- THIS TABLE INCLUDES THE MAPPING THAT PROVIDE LABEL NAMES*  
 **ON** NVL(A.DEPARTMENT\_VALUE,A.DEPARTMENT) = B.SYS\_ID  
 **LEFT** **JOIN**   
 CORE\_COMPANY\_\_VIEW C *-- THIS TABLE INCLUDES VENDOR RELATED DETAILS WHO PROVIDE THE BUSINESS APPS*  
 **ON** NVL(A.VENDOR\_VALUE, A.VENDOR) = C.SYS\_ID  
 **LEFT** **JOIN**   
 SYS\_USER\_GROUP\_\_VIEW D *-- THIS TABLE HELPS MAP IDENTIFIER CODE TO USER ASSIGNMENT GROUPS*  
 **ON** NVL(A.ASSIGNMENT\_GROUP\_VALUE,A.ASSIGNMENT\_GROUP) = D.SYS\_ID  
 **LEFT** **JOIN**   
 SYS\_USER\_\_VIEW E *-- THIS TABLE HELPS MAP IDENTIFIER TO USER NAMES*  
 **ON** NVL(A.OWNED\_BY\_VALUE, A.OWNED\_BY) = E.SYS\_ID  
 **LEFT** **JOIN**  
 SYS\_AUDIT\_DELETE\_\_VIEW DEL  
 **ON** A.SYS\_ID = DEL.DOCUMENTKEY  
 **WHERE**   
 DEL.DOCUMENTKEY **IS** **NULL**  
  
 )  
 **SELECT**  
 DEPARTMENT  
 ,BUSINESS\_CRITICALITY  
 ,COUNT(**DISTINCT** BUSINESS\_APP\_NAME) **AS** APP\_COUNT\_BY\_DEPT  
**FROM**  
 T1  
**GROUP** **BY**  
 DEPARTMENT  
 ,BUSINESS\_CRITICALITY  
**ORDER** **BY**  
 DEPARTMENT  
 ,BUSINESS\_CRITICALITY  
;

### Identify number of problem tickets opened

Problem tickets created by each parent group and respective child group. Details about who is working on the problem ticket, which state the problem is at, and which category and sub category the issue belongs to.

Tables to replicate to run this example:

CMDB\_CI TASK

PROBLEM

SYS\_AUDIT\_DELETE

SYS\_CHOICE SYS\_USER

SYS\_USER\_GROUP

**WITH** T1 **AS**(  
 **SELECT**  
 T.NUMBER **AS** TICKET\_NUMBER  
 ,G1.NAME **AS** PARENT\_ASSIGNMENT\_GROUP  
 ,G.NAME **AS** CHILD\_ASSIGNMENT\_GROUP  
 ,T.SHORT\_DESCRIPTION  
 ,T.DESCRIPTION  
 ,CI.NAME **AS** CONFIGURATION\_ITEM  
 ,CAT.**LABEL** **AS** **CATEGORY**  
 ,SUBCAT.**LABEL** **AS** SUBCATEGORY  
 ,ST.**LABEL** **AS** STATE  
 ,SU.NAME **AS** ASSIGNED\_TO  
 ,SU1.NAME **AS** OPENED\_BY  
 ,SU2.NAME **AS** RESOLVED\_BY  
 ,P.RESOLVED\_AT  
 ,P.FIX\_NOTES  
 ,P.U\_NUMBER\_OF\_USERS\_IMPACTED **AS** USERS\_IMPACTED  
 ,COALESCE(P.WORKAROUND, P.U\_WORKAROUND) **AS** WORKAROUND  
   
 **FROM**  
 TASK\_\_VIEW T  
 **LEFT** **JOIN** PROBLEM\_\_VIEW P   
 **ON** P.SYS\_ID=T.SYS\_ID  
 **LEFT** **JOIN** SYS\_USER\_GROUP\_\_VIEW G   
 **ON** NVL(T.ASSIGNMENT\_GROUP\_VALUE, T.ASSIGNMENT\_GROUP) = G.SYS\_ID *-- CHILD GROUP NAME*  
 **LEFT** **JOIN** SYS\_USER\_GROUP\_\_VIEW G1   
 **ON** NVL(G.PARENT\_VALUE, G.**PARENT**) = G1.SYS\_ID *--PARENT GROUPS*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW SU   
 **ON** T.ASSIGNED\_TO\_VALUE = SU.SYS\_ID *-- ASSIGNED TO USER DETAILS*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW SU1   
 **ON** T.OPENED\_BY\_VALUE = SU1.SYS\_ID *-- OPENED BY USER DETAILS*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW SU2   
 **ON** P.RESOLVED\_BY\_VALUE = SU2.SYS\_ID *-- RESOLVED BY USER DETAILS*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM** SYS\_CHOICE\_\_VIEW   
 **WHERE** NAME = 'PROBLEM'   
 **AND** ELEMENT = 'CATEGORY'  
 ) CAT   
 **ON** P.**CATEGORY** = CAT.VALUE *-- CATEGORY MAPPING*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM** SYS\_CHOICE\_\_VIEW   
 **WHERE** NAME = 'PROBLEM'   
 **AND** ELEMENT = 'SUBCATEGORY'  
 )SUBCAT   
 **ON** P.SUBCATEGORY = SUBCAT.VALUE *-- SUBCATEGORY MAPPING*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*  
 **FROM** SYS\_CHOICE\_\_VIEW   
 **WHERE** NAME = 'PROBLEM'   
 **AND** ELEMENT = 'STATE'  
 )ST   
 **ON** T.STATE = ST.VALUE *-- STATE MAPPING*  
 **LEFT** **JOIN** CMDB\_CI\_\_VIEW CI   
 **ON** T.CMDB\_CI\_VALUE = CI.SYS\_ID *-- CONFIGURATION ITEM*  
 **LEFT** **JOIN** SYS\_AUDIT\_DELETE\_\_VIEW DEL   
 **ON** T.SYS\_ID = DEL.DOCUMENTKEY *--DELETED TICKETS*  
   
 **WHERE** T.SYS\_CLASS\_NAME = 'PROBLEM' *-- THIS FIELD BROADLY IDENTIFIES THE TICKET TYPE*  
 **AND** DEL.DOCUMENTKEY **IS** **NULL**  
)  
  
**SELECT**  
 PARENT\_ASSIGNMENT\_GROUP  
 ,CHILD\_ASSIGNMENT\_GROUP  
 ,CONFIGURATION\_ITEM  
 ,**CATEGORY**  
 ,COUNT(**DISTINCT** TICKET\_NUMBER) **AS** TICKET\_COUNT  
**FROM**   
 T1  
**GROUP** **BY**  
 PARENT\_ASSIGNMENT\_GROUP  
 ,CHILD\_ASSIGNMENT\_GROUP  
 ,CONFIGURATION\_ITEM  
 ,**CATEGORY**  
**ORDER** **BY**  
 PARENT\_ASSIGNMENT\_GROUP  
 ,CHILD\_ASSIGNMENT\_GROUP  
 ,CONFIGURATION\_ITEM  
 ,**CATEGORY**  
;

### First contact resolution percentage by Configuration Item (application)

How many tickets (Incidents + Service Requests) were resolved by Helpdesk right when they were created instead of hopping them onto another department or assignment group.

Tables to replicate to run this example:

CMDB\_CI CMN\_DEPARTMENT INCIDENT SC\_REQUEST SC\_REQ\_ITEM SYS\_AUDIT\_DELETE SYS\_CHOICE SYS\_USER SYS\_USER\_GROUP TASK

**WITH** T1 **AS** (  
 **SELECT**  
 **DISTINCT**  
 T.NUMBER **AS** TICKET\_NUMBER  
 ,G1.NAME **AS** PARENT\_ASSIGNMENT\_GROUP  
 ,G.NAME **AS** CHILD\_ASSIGNMENT\_GROUP  
 ,D1.NAME **AS** DEPARTMENT  
 ,T.SHORT\_DESCRIPTION  
 ,T.DESCRIPTION  
 ,CI.NAME **AS** CONFIGURATION\_ITEM  
 ,SC\_CAT.**LABEL** **AS** **CATEGORY**  
 ,SC\_SUBCAT.**LABEL** **AS** SUBCATEGORY  
 ,SU.NAME **AS** ASSIGNED\_TO  
 ,T.SYS\_CREATED\_ON **AS** CREATED\_ON  
  
 **FROM**  
 TASK\_\_VIEW T  
 **LEFT** **JOIN** INCIDENT\_\_VIEW I   
 **ON** I.SYS\_ID = T.SYS\_ID *-- ADDITIONAL INCIDENT DETAIL*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM** SYS\_CHOICE\_\_VIEW SC\_CAT   
 **WHERE** ELEMENT = 'U\_T\_CATEGORY'  
 ) SC\_CAT  
 **ON** T.U\_T\_CATEGORY = SC\_CAT.VALUE *-- MAPPING FOR CATEGORY VALUES FROM* **TASK** *TABLE*  
 **LEFT** **JOIN**   
 (  
 **SELECT**   
 \*   
 **FROM** SYS\_CHOICE\_\_VIEW   
 **WHERE** ELEMENT = 'U\_T\_SUBCATEGORY'   
 **AND** NAME ='SC\_REQ\_ITEM'  
 )SC\_SUBCAT   
 **ON** T.U\_T\_SUBCATEGORY = SC\_SUBCAT.VALUE *-- MAPPING FOR SUBCATEGORY VALUES FROM* **TASK** *TABLE*  
 **LEFT** **JOIN** CMDB\_CI\_\_VIEW CI   
 **ON** T.CMDB\_CI\_VALUE = CI.SYS\_ID *--CONFIGURATION ITEM OR APPLICATION NAME*  
 **LEFT** **JOIN** SC\_REQ\_ITEM\_\_VIEW R   
 **ON** T.SYS\_ID = R.SYS\_ID *--RITM OR SERVICE REQUEST INFORMATION*  
 **LEFT** **JOIN** SC\_REQUEST\_\_VIEW SR   
 **ON** R.REQUEST\_VALUE = SR.SYS\_ID *--RITM REQUESTED FOR INFORMATION*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW SU   
 **ON** T.ASSIGNED\_TO\_VALUE = SU.SYS\_ID *-- ASSIGNED TO USERS NAME*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW SU1   
 **ON** T.OPENED\_BY\_VALUE = SU1.SYS\_ID *----OPENED BY USERS NAME*  
 **LEFT** **JOIN** SYS\_USER\_\_VIEW U2   
 **ON** I.CALLER\_ID\_VALUE = U2.SYS\_ID *---INCIDENT REQUESTED FOR NAME*  
 **LEFT** **JOIN** CMN\_DEPARTMENT\_\_VIEW D1 *-- DEPARTMENT MAPPING*  
 **ON** D1.SYS\_ID=SU.DEPARTMENT\_VALUE  
 **LEFT** **JOIN** SYS\_USER\_GROUP\_\_VIEW G   
 **ON** NVL(T.ASSIGNMENT\_GROUP\_VALUE, T.ASSIGNMENT\_GROUP) = G.SYS\_ID *---CHILD GROUP NAME*  
 **LEFT** **JOIN** SYS\_USER\_GROUP\_\_VIEW G1   
 **ON** NVL(G.PARENT\_VALUE, G.**PARENT**) = G1.SYS\_ID *--PARENT GROUPS*  
 **LEFT** **JOIN** SYS\_AUDIT\_DELETE\_\_VIEW DEL   
 **ON** T.SYS\_ID = DEL.DOCUMENTKEY *--DELETED TICKETS*  
  
 **WHERE** DEL.DOCUMENTKEY **IS** **NULL**  
 **AND**  
 (  
 I.SYS\_ID **IS** **NOT** **NULL**   
 **OR**   
 R.SYS\_ID **IS** **NOT** **NULL**  
 )  
)  
  
, T2 **AS** (  
 **SELECT**  
 CONFIGURATION\_ITEM  
 ,ASSIGNED\_TO  
 ,YEAR(CREATED\_ON) YR  
 ,MONTH(CREATED\_ON) MO  
 ,TICKET\_NUMBER  
 ,**CASE**   
 **WHEN** LOWER(CHILD\_ASSIGNMENT\_GROUP) = 'IT - SNOWDESK'   
 **AND** LOWER(ASSIGNED\_TO) != 'LIFT AUTOMATION'   
 **AND** LOWER(ASSIGNED\_TO) != 'SNOW BOT'   
 **THEN** 1  
 **ELSE** 0  
 **END** CT  
 ,1 **AS** TCKT\_CT  
   
 **FROM** T1  
)  
  
**SELECT**  
 YR  
 ,MO  
 ,CONFIGURATION\_ITEM  
 ,(SUM(CT) / SUM(TCKT\_CT))\*100 **AS** FCR\_PCT  
**FROM**  
 T2  
**GROUP** **BY**  
 YR  
 ,MO  
 ,CONFIGURATION\_ITEM  
**ORDER** **BY**  
 YR  
 ,MO  
 ,CONFIGURATION\_ITEM  
;