SYRACUSE UNIVERSITY
SCHOOL OF INFORMATION STUDIES
PROJECT IMPLEMENTATION REPORT

SERVICE TICKET MANAGEMENT

- AIYAPPA UTHAIAH

ABSTRACT: The project focuses on assisting the IT services industry specifically in the field of service ticket management. Prior to the project tickets were handled using MS Excel to maintain all the data in a tabular format. Different MS excel functions such as filter and pivot table are used to get an idea about the tasks and how to go about handling them. This method was error prone since the data maintenance was manual and also not timely. There are risks of duplication or loss of data which will have severe consequences. The project addresses these issues and will be able to simplify the process and also automate some tasks to a certain extent.

SYRACUSE UNIVERSITY
School of Information Studies

Contents

SERVICE TICKET MANAGEMENT SYSTEM

PROJECT SUMMARY

The 'IT Service Ticket Management System' focuses on assisting the IT services industry in the process of creation of service requests and also to enable better IT management. IT services can include activities such as consulting, application support, technology support, enterprise software etc. Majority of the service provided is concentrated towards application development and maintenance which makes up a large chunk of the revenue generated. In an IT service industry any request for the application development and maintenance is handled as a ticket (different organizations have different nomenclature). It can be a request for changing existing functionality or to create a new functionality or to flag an issue that has cropped up in the system.

Currently there are various ways in which the client submits a request which can be via e-mail, phone or over live-chat. The implementation of the ticket management system will eliminate the use of multiple modes and unify it under one system. This will allow us to centralize all the requests from the customer of various teams such as Finance, Accounting, and Marketing etc. on one platform. Another data management problem being faced currently is the effective collection, maintenance and delegation of tasks that are received by the clients. At present, to handle this the company is using MS Excel to maintain all the data in a tabular format. Different MS excel functions such as filter and pivot table are used to get an idea about the tasks and how to go about handling them.

The risks of continuing with the existing methodology of service ticket management are:

- It is error prone since the data maintenance is manual and also not timely. There are risks of duplication or loss of data which will have severe consequences.
- Secondly the data maintained is not secure and can be viewed or modified without the necessary approvals.

DESIRED SOLUTION

- There are two types of users who use the ticket management system: Clients and employees of the service company.
- Clients will use the ticket management system to raise tickets for any work that has to be delegated to the service company.
- The Client will assign the ticket to the specified individual. They also classify the ticket as either a ‘Change Request’ or as an ‘Incident’.
- Change requests tickets consist of addition of new business processes in the system or modification of business processes, changes to the look and feel of the system etc. Change requests will require that there be a time, scope and cost estimate done for the change before start.
- Incidents are setbacks in the system which will or have severely affected the system causing impact to the business. An incidents require that the task be picked up and resolved with utmost priority since it affects the clients’ business process.
- Apart from the authorization to create the ticket, the clients will only have display access once they have created the ticket.
- Employee from the service company will be able to view, modify and edit tickets that are assigned to them.
- Employees will also have display access for other tickets that they are not authorized to edit.
- Employees will have to provide an estimate on the number of hours of work the request involves and record it in the ticket.
- At every stage of ticket resolution the employee has to update the status so that the client is informed about it.
- On completion of the request the employee closes the task i.e. the status of the task is updated to close for change requests and resolved for incidents.
- Both the users will be able to run reports to fetch ticket details form various tables for reporting and managerial purposes.

TABLES AND ATTRIBUTES:

Data Object	Explanation
Users	**Contains all user information**
User_ID (PK)	PRIMARY KEY – User_ID identifies all the users uniquely
Team_ID	FOREIGN KEY – Team_ID contains the ID of the team to which the user belongs to. Associated with the primary key of Team Table.
User_FName	
User_LName	
User_MobNo	
User_Email	
User_Role	User role can be of two types: Clients or Employees. Based on the input 'C' or 'E' the role is decided as Client and Employee respectively.
Team	**Contains Team Description**
Team_ID	PRIMARY KEY – Each team in the client and service company has an ID associated with it.
Team_Name	Description of the team.
Login Cred	**Contains User Login info**
User_ID	PIMARY KEY, FOREIGN KEY - Associated with the primary key of User Table.
Username	Username of the user required for login
User_Pwd	Password of the user required for login
Client	**Child Entity of User. Contains Client user information**
User_ID	PRIMARY KEY, FOREIGN KEY – Associated with the primary key of User table.
User_Team	FOREIGN KEY – Associated with the primary key 'Team_ID' of Team table
User_Location	Identifies where the user works out of.
CRQ_Count	Count of Change Requests raised by Client
INC_Count	Count of Incidents raised by Client
Employee	**Child Entity of User. Contains Employee user information**
User_ID	PRIMARY KEY, FOREIGN KEY – Associated with the primary key of User table.
User_Team	FOREIGN KEY – Associated with the primary key 'Team_ID' of Team table
User_Location	Identifies where the user works out of.
CRQ_Count	Count of Change Requests assigned to employee
INC_Count	Count of Incidents assigned to employee

Ticket	Contains information of ticket raised by the Client
Ticket_ID	PRIMARY KEY - ID generated when the client creates a new ticket
Owner_ID	FOREIGN KEY – Associated with the User ID in the Client table specifically. Only the Client user can be the owner of a ticket or only the client as creation rights.
Ticket_Type	Ticket type can be of two types: Change Request and Incident. Based on the input 'C' or 'I' the ticket type is decided as Change request or Incident respectively.
Change Request	**Contains the details of the Change request ticket created by the Client. It is also accessed by the Employee for view and editing purposes.**
CRQ_Number	PRIMARY KEY – Unique key to identify the Ticket as a change request
Ticket_ID	FOREIGN KEY – Associated with the Primary Key 'Ticket_ID' in Ticket Table
Assignee_ID	FOREIGN KEY - Associated with the User_ID in the table employee.
Request_Type	FOREIGN KEY – Associated with primary key "Request_Type" of Change Request type table
Status	FOREIGN KEY – Associated with primary key "Status" of Change Status table
Priority	FOREIGN KEY – Associated with primary key "Priority_Scale" of Change Priority table
Summary	Summary of the task. This is mandatory.
Notes Change_Description	Detailed description of the change.
Future_Enhancements	Instructions to the client/ future users as to how to carry out tests
Test_Instructions	Special instructions
Knowledge_Transfer	Date on which the Change request has been created.
Creation_Date	The employee performs a time estimate for the request and maintains an estimated start and end date.
Estimated_Start_Date	Date on which the work has started
Estimated_End_Date	Date on which the work ends and the status is set to close.
Actual_Start_Date	
Actual_End_Date	
Change Req Type	**Contains the details of type of change request initiated by the Client**
Request_Type	PRIMARY KEY – Type defines the kind of request raised by the Client Eg:- New development, enhancement, R&D etc.
Request_Typ_Description	Describes more about the type of request raised

Change Status	**Contains the current status of the change request at any given time**
Status Status_Description	PRIMARY KEY – Describes status of unique request Illustrates the status in detail
Change Priority	**Stores the priority assigned to a change request based on the criticality of the request**
Priority_Scale Priority_Description Priority_MaxTime	PRIMARY KEY – Defines the priority of the ticket (High/Medium/Low) Provides a description on the importance of the priority given to the request Estimates the time-frame required to resolve the request
Incident	**Contains the details of the Incident ticket created by the Client. It is also accessed by the Employee for view and editing purposes.**
INC_Number Ticket_ID Assignee Impact_Scale Status Summary Notes INC_Reason Resolution Creation_Date Start_Date End_Date	PRIMARY KEY – Unique key to identify the Ticket as an Incident FOREIGN KEY – Associated with the Primary Key 'Ticket_ID' in Ticket Table FOREIGN KEY - Associated with the User_ID in the table employee. FOREIGN KEY – Associated with primary key "Impact_Scale" of Incident Impact table. It will consist of a rating scale from 1-5 with 1 being severe and 5 being low. FOREIGN KEY – Associated with primary key "Status" of Incident Status table Summary of the task. This is mandatory Reason as to why the incident was raised and the impact description. Description as to how the incident was resolved. Date on which the Change request has been created. Date on which the work has started Date on which the work ends and the status is set to close.
Incident Impact	**Contains the rating of the impact that the Incident has on the functioning of the business.**
Impact_Scale Impact_Description Priority_Description Priority_MaxTime	PRIMARY KEY - It will consist of a rating scale from 1-5 with 1 being severe and 5 being low for the Incident. Description of each rating Provides a description on the importance of the priority given to the request Estimates the time-frame required to resolve the request

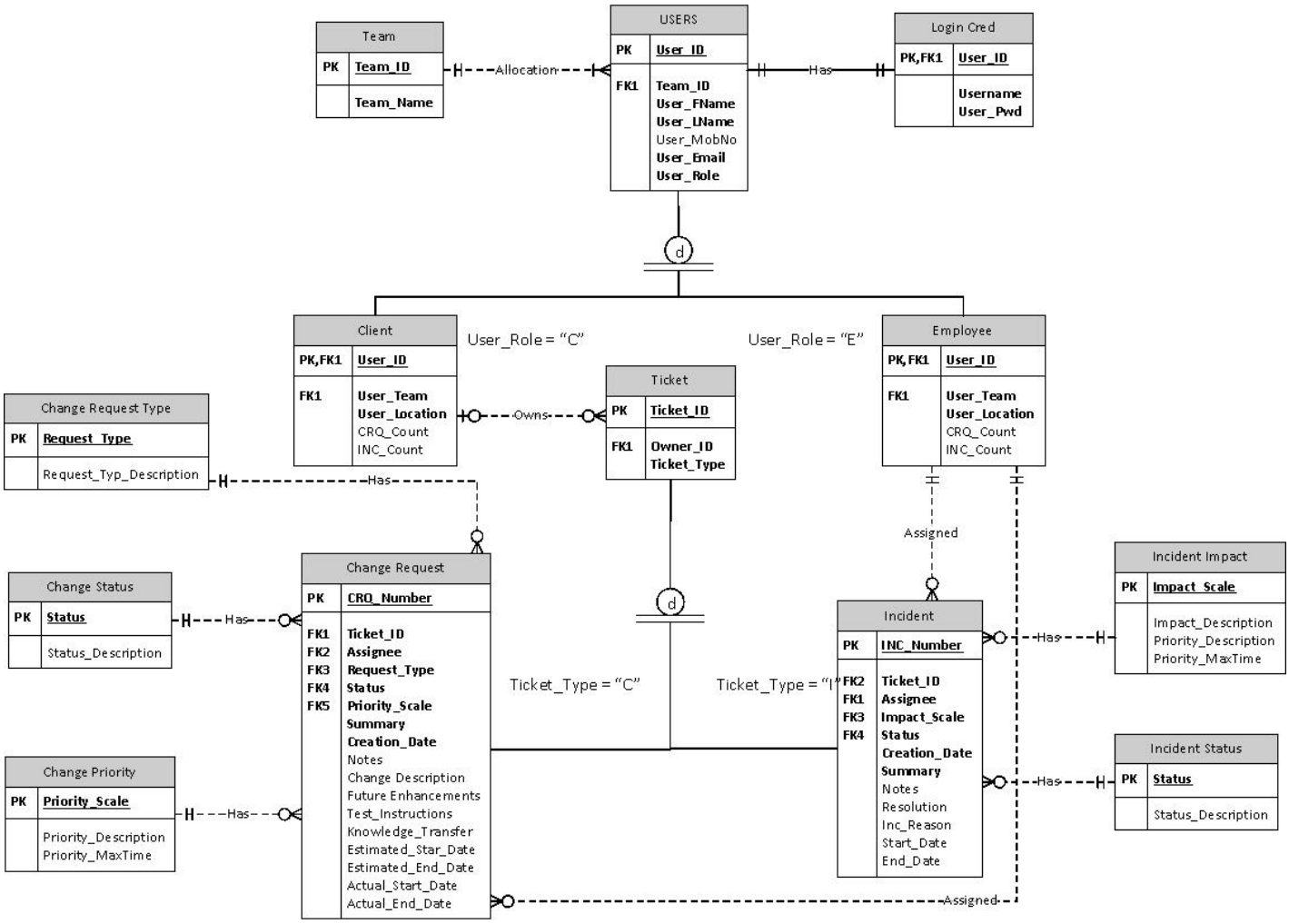
Incident Status	**Contains the current status of the Incident at any given time**
Status Status_Description	PRIMARY KEY – Describes status of unique request Illustrates the status in detail
Users	**Contains all user information**
User_ID (PK) Team_ID User_FName User_LName User_MobNo User_Email User_Role	PRIMARY KEY – User_ID identifies all the users uniquely FOREIGN KEY – Team_ID contains the ID of the team to which the user belongs to. Associated with the primary key of Team Table. User role can be of two types: Clients or Employees. Based on the input 'C' or 'E' the role is decided as Client and Employee respectively.
Team	**Contains Team Description**
Team_ID Team_Name	PRIMARY KEY – Each team in the client and service company has an ID associated with it. Description of the team.
Login Cred	**Contains User Login info**
User_ID Username User_Pwd	PIMARY KEY, FOREIGN KEY - Associated with the primary key of User Table. Username of the user required for login Password of the user required for login
Client	**Child Entity of User. Contains Client user information**
User_ID User_Team User_Location	PRIMARY KEY, FOREIGN KEY – Associated with the primary key of User table. FOREIGN KEY – Associated with the primary key 'Team_ID' of Team table Identifies where the user works out of.
Employee	**Child Entity of User. Contains Employee user information**
User_ID User_Team User_Location	PRIMARY KEY, FOREIGN KEY – Associated with the primary key of User table. FOREIGN KEY – Associated with the primary key 'Team_ID' of Team table Identifies where the user works out of.
Ticket	**Contains information of ticket raised by the Client**
Ticket_ID Owner_ID Ticket_Type	PRIMARY KEY - ID generated when the client creates a new ticket PRIMARY KEY, FOREIGN KEY – Associated with the User ID in the Client table specifically. Only the Client user can be the owner of a ticket or only the client as creation rights.

	Ticket type can be of two types: Change Request and Incident. Based on the input 'C' or 'I' the ticket type is decided as Change request or Incident respectively.
Change Request	**Contains the details of the Change request ticket created by the Client. It is also accessed by the Employee for view and editing purposes.**
Ticket_ID CRQ_Number Owner_ID Assignee_ID Request_Type Status Priority	PRIMARY KEY, FOREIGN KEY – Associated with the Primary Key 'Ticket_ID' in Ticket Table PRIMARY KEY – Unique key to identify the Ticket as a change request FOREIGN KEY - Associated with the Owner_ID in the table ticket. FOREIGN KEY - Associated with the User_ID in the table employee. FOREIGN KEY – Associated with primary key "Request_Type" of Change Request type table FOREIGN KEY – Associated with primary key "Status" of Change Status table FOREIGN KEY – Associated with primary key "Priority_Scale" of Change Priority table
Change Req Type	**Contains the details of type of change request initiated by the Client**
Request_Type Request_Typ_Description	PRIMARY KEY – Type defines the kind of request raised by the Client Eg:- New development, enhancement, R&D etc. Describes more about the type of request raised
Change Status	**Contains the current status of the change request at any given time**
Status Status_Description	PRIMARY KEY – Describes status of unique request Illustrates the status in detail
Change Priority	**Stores the priority assigned to a change request based on the criticality of the request**
Priority_Scale Priority_Description Priority_MaxTime	PRIMARY KEY – Defines the priority of the ticket (High/Medium/Low) Provides a description on the importance of the priority given to the request Estimates the time-frame required to resolve the request
Change Request Notes	**Comprises of the additional details associated with the change request**
CRQ_Number Summary Notes	PRIMARY KEY, FOREIGN KEY – Associated with the CRQ_Number in the Change request table. It connects the change request with all notes. Summary of the task. This is mandatory.

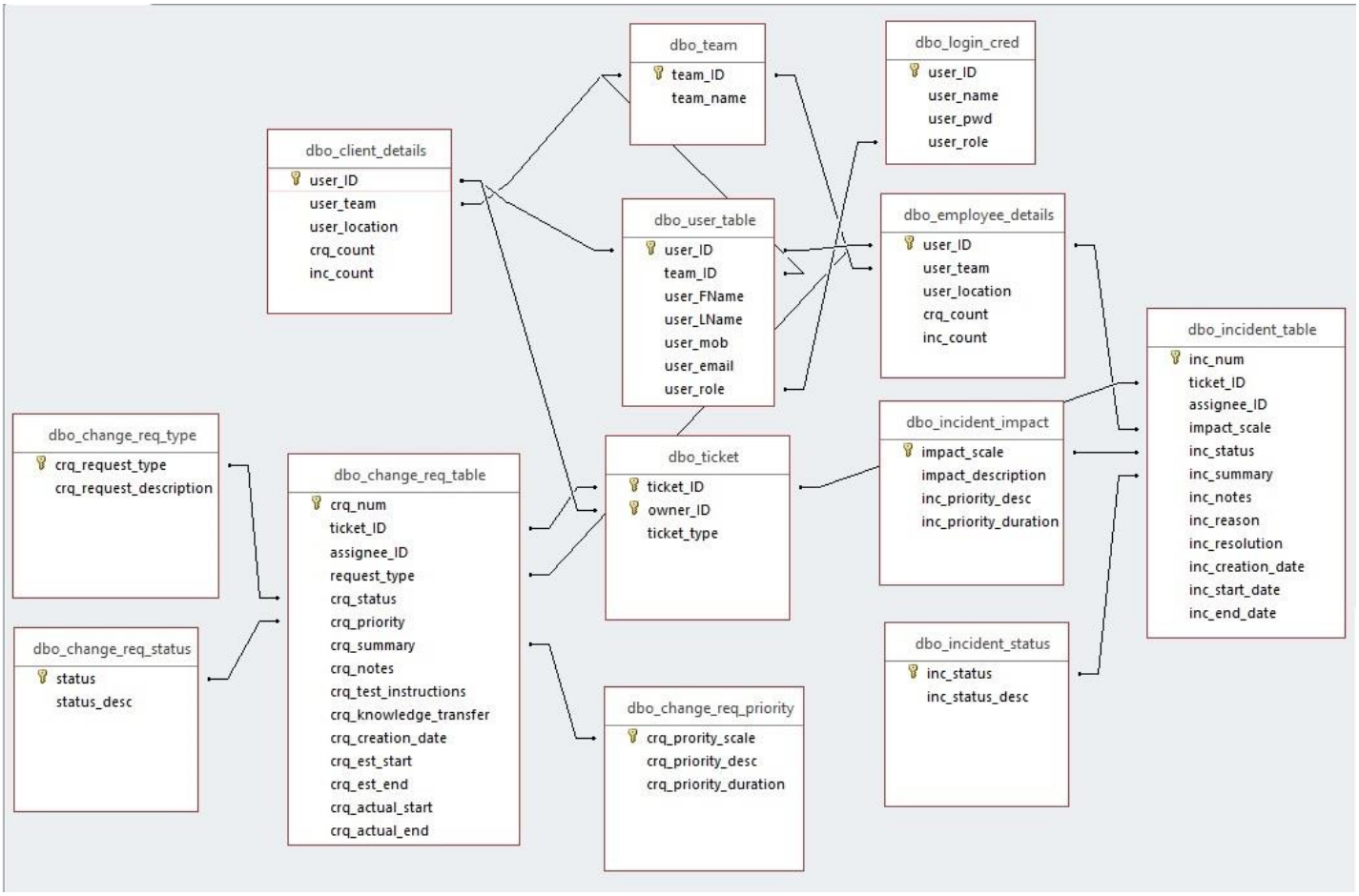
Change_Description Future_Enhancements Test_Instructions Knowledge_Transfer	Detailed description of the change. Instructions to the client/ future users as to how to carry out tests Special instructions
Change Req Date	**Comprises of all the dates associated with the change request**
CRQ_Number Creation_Date Estimated_Start_Date Estimated_End_Date Actual_Start_Date Actual_End_Date	PRIMARY KEY, FOREIGN KEY - Associated with the CRQ_Number in the Change request table. It connects the change request with all the dates associated with this ticket. Date on which the Change request has been created. The employee performs a time estimate for the request and maintains an estimated start and end date. Date on which the work has started Date on which the work ends and the status is set to close.
Incident	**Contains the details of the Incident ticket created by the Client. It is also accessed by the Employee for view and editing purposes.**
Ticket_ID INC_Number Owner_ID Assignee Impact_Scale Status Priority_Scale	PRIMARY KEY, FOREIGN KEY – Associated with the Primary Key 'Ticket_ID' in Ticket Table PRIMARY KEY – Unique key to identify the Ticket as an Incident FOREIGN KEY - Associated with the Owner_ID in the table ticket. FOREIGN KEY - Associated with the User_ID in the table employee. FOREIGN KEY – Associated with primary key "Impact_Scale" of Incident Impact table. It will consist of a rating scale from 1-5 with 1 being severe and 5 being low. FOREIGN KEY – Associated with primary key "Status" of Incident Status table FOREIGN KEY – Associated with primary key "Priority_Scale" of Incident Priority table
Incident Impact	**Contains the rating of the impact that the Incident has on the functioning of the business.**
Impact_Scale Impact_Description	PRIMARY KEY - It will consist of a rating scale from 1-5 with 1 being severe and 5 being low for the Incident. Description of each rating
Incident Status	**Contains the current status of the Incident at any given time**
Status Status_Description	PRIMARY KEY – Describes status of unique request Illustrates the status in detail
Incident Priority	**Stores the priority assigned to an Incident based on the criticality of the request**
Priority_Scale Priority_Description	PRIMARY KEY – Defines the priority of the ticket (High/Medium/Low) Provides a description on the importance of the priority given to the request

Priority_MaxTime	Estimates the time-frame required to resolve the request
Incident Notes	**Comprises of the additional details associated with the Incident**
INC_Number	PRIMARY KEY, FOREIGN KEY – Associated with the CRQ_Number in the Change request table. It connects the change request with all notes.
Summary	Summary of the task. This is mandatory
Notes	
INC_Reason	Reason as to why the incident was raised and the impact description.
Resolution	Description as to how the incident was resolved.
Incident Date	**Comprises of all the dates associated with the Incident**
INC_Number	PRIMARY KEY, FOREIGN KEY - Associated with the CRQ_Number in the Change request table. It connects the change request with all the dates associated with this ticket.
Creation_Date	Date on which the Change request has been created.
Start_Time	Time on which the work has started
End_Time	Time on which the work has ended
End_Date	Date on which the work ends and the status is set to close.

ENTITY RELATIONSHIP DIAGRAM



ACCESS RELATIONSHIP DIAGRAM



BUSINESS RULES

- One user will have only one login credentials. System cannot have duplicate login credentials.
- One user will be a part of only one team. A team will have at least one user.
- User is a Super type which has two Subtypes: Client and Employee.
- A client can create multiple tickets. A ticket is created by only one client
- Ticket is a Super type which has two subtypes: Change Request & Incident.
- An employee can be assigned multiple change requests. A change request is assigned to only one employee.
- An employee can be assigned multiple incidents. An incident is resolved by only one employee.
- A change request has only one request type. A request type can be given to multiple change request.
- A change request has only one status at a given time. A particular status can be assigned to multiple requests.
- A change request is assigned only one priority. Priority scales can be assigned to multiple change requests.
- A change request will contain one change request note. Every change request note will be associated with only one change request.
- Each incident will have an impact on the system. An impact can be caused by multiple incidents.
- Each incident is assigned a status. A status can be assigned to multiple incidents.
- An incident will have only one incident note. Each incident note will be associated with only one incident.

DATABASE INFRASTRUCTURE

The database infrastructure is based on client-server model. SQL server is used as the database engine and access is used as the interface design tool. Data is inserted, deleted, updated and queried from the SQL server database with the help of forms on Access. Useful data stored on SQL database can also be viewed with the help of reports generated through access.

SQL SCRIPTS FOR CREATING AND INSERTING SAMPLE DATA

CREATE: team

```
--Team Table Creation
CREATE TABLE team
(
    team_ID VARCHAR(10) PRIMARY KEY,
    team_name VARCHAR(20) NOT NULL,
);
```

CREATE: user_table

```
--User Table Creation
CREATE TABLE user_table
(
    user_ID VARCHAR(10),-- PRIMARY KEY,
    team_ID VARCHAR (10) NOT NULL,
    user_FName VARCHAR(20) NOT NULL,
    user_LName VARCHAR(20) NOT NULL,
    user_mob VARCHAR(15),
    user_email VARCHAR(50) NOT NULL,
    user_role CHAR(1) NOT NULL,

CONSTRAINT user_table_PK PRIMARY KEY (user_ID),

CONSTRAINT user_table_FK FOREIGN KEY (team_ID) REFERENCES team(team_ID),
);
```

CREATE: login_cred

```
--Login Credentials Table Creation
CREATE TABLE login_cred
(
    user_ID VARCHAR(10) PRIMARY KEY,
    user_name VARCHAR(20) NOT NULL,
    user_pwd VARCHAR(20) NOT NULL,

CONSTRAINT login_cred_FK FOREIGN KEY(user_ID) REFERENCES user_table(user_ID),
);
```

CREATE: client_details

```
--Client Details Table Creation
CREATE TABLE client_details
(
    user_ID VARCHAR(10) PRIMARY KEY,
    user_team VARCHAR(10) NOT NULL,
    user_location VARCHAR(30) NOT NULL,
    crq_count INTEGER,
    inc_count INTEGER,

CONSTRAINT client_details_FK FOREIGN KEY(user_ID) REFERENCES user_table(user_ID),
CONSTRAINT client_details_FK_user_team FOREIGN KEY(user_team) REFERENCES team(team_ID),
);
```

CREATE: employee_details

```
--Employee Table Creation
CREATE TABLE employee_details
(
    user_ID VARCHAR(10) PRIMARY KEY,
    user_team VARCHAR(10) NOT NULL,
    user_location VARCHAR(30) NOT NULL,
    crq_count INTEGER,
    inc_count INTEGER,

CONSTRAINT employee_FK FOREIGN KEY(user_ID) REFERENCES user_table(user_ID),
CONSTRAINT employee_FK_user_team FOREIGN KEY(user_team) REFERENCES team(team_ID),
);
```

CREATE: ticket

```
--Ticket Table Creation
CREATE TABLE ticket
(
    ticket_ID VARCHAR(10),
    owner_ID VARCHAR(10) NOT NULL,
    ticket_type CHAR(01) NOT NULL,


CONSTRAINT ticket_PK PRIMARY KEY(ticket_ID),
CONSTRAINT ticket_FK FOREIGN KEY(owner_ID) REFERENCES client_details(user_ID),
);
```

CREATE: change_req_type

```
--Change Request Type Table Creation
CREATE TABLE change_req_type
(
    crq_request_type CHAR(02) PRIMARY KEY,
    crq_request_description VARCHAR(25),
);
```

CREATE: change_req_status

```
--Change Request Status Table Creation
CREATE TABLE change_req_status
(
    status CHAR(02) PRIMARY KEY,
    status_desc VARCHAR(25),
);
```

CREATE: change_req_priority

```
--Change Request Priority Table Creation
CREATE TABLE change_req_priority
(
    crq_prority_scale CHAR(02) PRIMARY KEY,
    crq_priority_desc VARCHAR(25),
    crq_priority_duration INTEGER,
);
```

CREATE: change_req_table

```
--Change Request Table Creation
CREATE TABLE change_req_table
(
    crq_num VARCHAR(10),
    ticket_ID VARCHAR(10) NOT NULL,
    assignee_ID VARCHAR(10) NOT NULL,
    request_type CHAR(02) NOT NULL,
    crq_status CHAR(02) NOT NULL,
    crq_priority CHAR(02) NOT NULL,
    crq_summary VARCHAR(50) NOT NULL,
    crq_notes VARCHAR(150),
    crq_test_instructions VARCHAR(250),
    crq_knowledge_transfer VARCHAR(250),
    crq_creation_date DATETIME DEFAULT GETDATE() NOT NULL,
    crq_est_start DATETIME,
    crq_est_end DATETIME,
    crq_actual_start DATETIME,
    crq_actual_end DATETIME,

CONSTRAINT change_req_PK PRIMARY KEY(crq_num),
CONSTRAINT change_req_FK_ticket_id FOREIGN KEY(ticket_ID) REFERENCES ticket(ticket_ID),
CONSTRAINT change_req_FK_assignee_ID FOREIGN KEY(assignee_ID) REFERENCES employee_details(user_ID),
CONSTRAINT change_req_FK_request_type FOREIGN KEY(request_type) REFERENCES change_req_type(crq_request_type),
CONSTRAINT change_req_FK_crq_status FOREIGN KEY(crq_status) REFERENCES  change_req_status(status),
CONSTRAINT change_req_FK_crq_priority FOREIGN KEY(crq_priority) REFERENCES change_req_priority(crq_prority_scale),
);
```

CREATE: incident_impact

```
--Incident Impact Table Creation
CREATE TABLE incident_impact
(
    impact_scale CHAR(02) PRIMARY KEY,
    impact_description VARCHAR(40),
    inc_priority_desc CHAR(20),
    inc_priority_duration INTEGER,
);
```

CREATE: incident_status

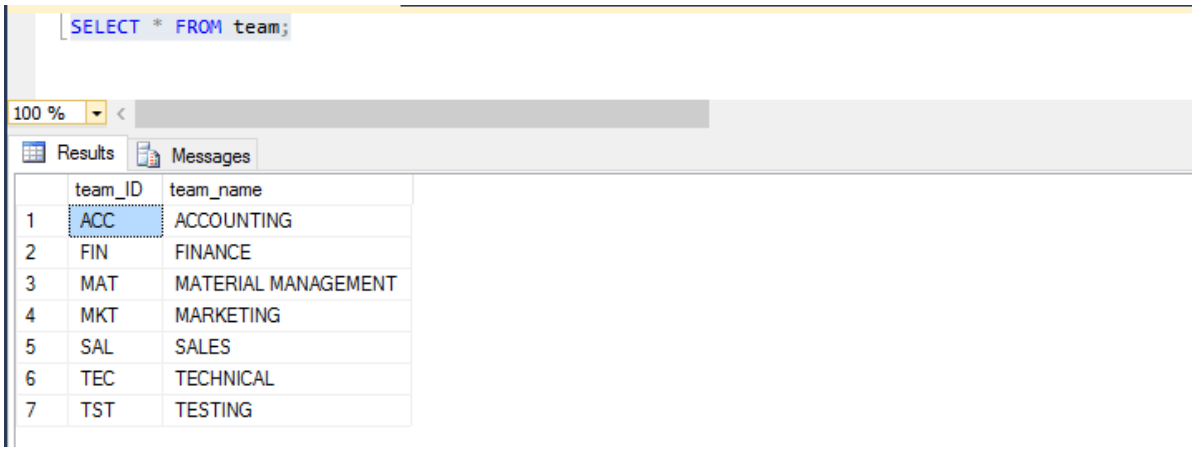
```
--Incident Status Table Creation
CREATE TABLE incident_status
(
    inc_status CHAR(02) PRIMARY KEY,
    inc_status_desc CHAR(20),
);
```

CREATE: incident_table

```
--Incident Table Creation
CREATE TABLE incident_table
(
    inc_num VARCHAR(10) PRIMARY KEY,
    ticket_ID VARCHAR(10),
    assignee_ID VARCHAR(10),
    impact_scale CHAR(02) NOT NULL,
    inc_status CHAR(02),
    inc_summary CHAR(50) NOT NULL,
    inc_notes CHAR(250),
    inc_reason VARCHAR(250),
    inc_resolution VARCHAR(250),
    inc_creation_date DATETIME DEFAULT GETDATE(),
    inc_start_date DATETIME,
    inc_end_date DATETIME,

CONSTRAINT incident_table_FK FOREIGN KEY(ticket_ID) REFERENCES ticket(ticket_ID),
CONSTRAINT incident_table_FK_assignee_ID FOREIGN KEY(assignee_ID) REFERENCES employee_details(user_ID),
CONSTRAINT incident_table_FK_impact_scale FOREIGN KEY(impact_scale) REFERENCES incident_impact(impact_scale),
CONSTRAINT incident_table_FK_inc_status FOREIGN KEY(inc_status) REFERENCES  incident_status(inc_status),
);
```

INSERT DATA: team

```
INSERT INTO team (team_ID, team_name)
VALUES ('FIN','FINANCE');

INSERT INTO team (team_ID, team_name)
VALUES ('ACC','ACCOUNTING');

INSERT INTO team (team_ID, team_name)
VALUES ('MKT','MARKETING');

INSERT INTO team (team_ID, team_name)
VALUES ('MAT','MATERIAL MANAGEMENT');

INSERT INTO team (team_ID, team_name)
VALUES ('SAL','SALES');

INSERT INTO team (team_ID, team_name)
VALUES ('TEC','TECHNICAL');

INSERT INTO team (team_ID, team_name)
VALUES ('TST','TESTING');
```



```
SELECT * FROM team;
```

100 %

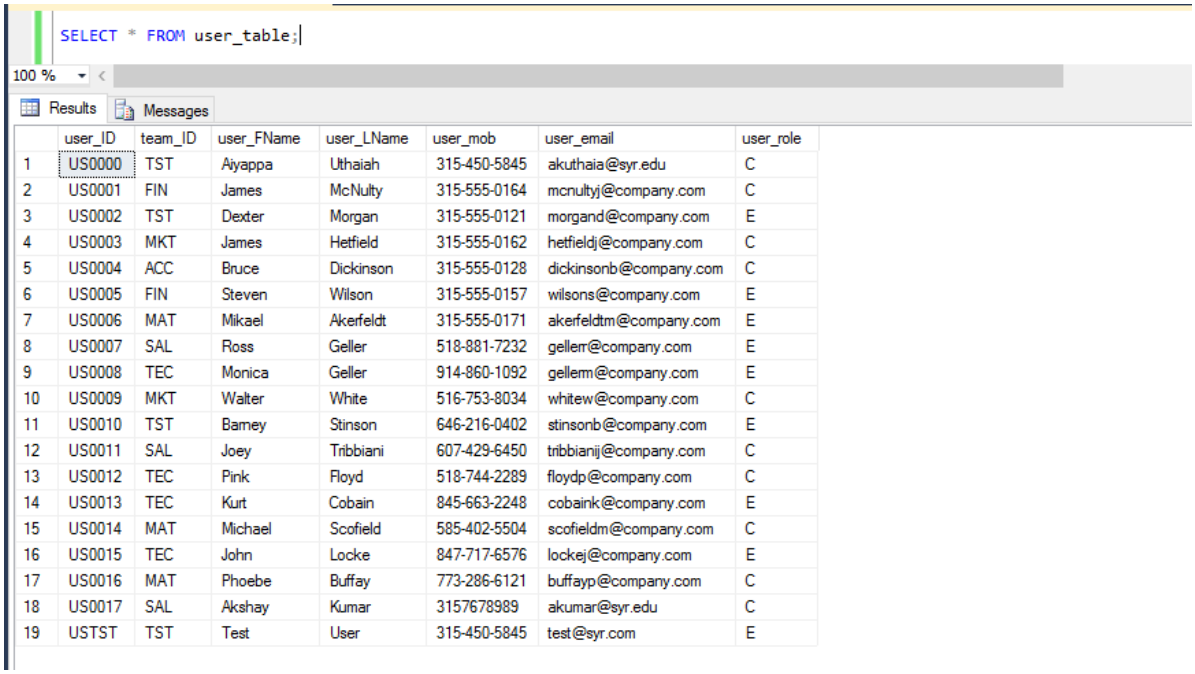
Results | Messages

	team_ID	team_name
1	ACC	ACCOUNTING
2	FIN	FINANCE
3	MAT	MATERIAL MANAGEMENT
4	MKT	MARKETING
5	SAL	SALES
6	TEC	TECHNICAL
7	TST	TESTING

INSERT DATA: user_table

```
INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0001','FIN','James','McNulty','315-555-0164','mcnultyj@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0002','TST','Dexter','Morgan','315-555-0121','morgand@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0003','MKT','James','Hetfield','315-555-0162','hetfieldj@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0004','ACC','Bruce','Dickinson','315-555-0128','dickinsonb@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0005','FIN','Steven','Wilson','315-555-0157','wilsons@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0006','MAT','Mikael','Akerfeldt','315-555-0171','akerfeldtm@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0007','SAL','Ross','Geller','518-881-7232','gellerr@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0008','TEC','Monica','Geller','914-860-1092','gellerm@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0009','MKT','Walter',   'White','516-753-8034','whitew@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0010','TST','Barney','Stinson','646-216-0402','stinsonb@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0011','SAL','Joey','Tribbiani','607-429-6450','tribbianij@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0012','TEC','Pink','Floyd','518-744-2289','floydp@company.com','C');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0013','TEC','Kurt','Cobain','845-663-2248','cobaink@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0014','MAT','Michael','Scofield','585-402-5504','scofieldm@company.com','C');
```

```
INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0015','TEC','John','Locke','847-717-6576','lockej@company.com','E');

INSERT INTO user_table (user_ID,team_ID,user_FName,user_LName,user_mob,
user_email,user_role)
VALUES ('US0016','MAT','Phoebe','Buffay','773-286-6121','buffayp@company.com','C');
```



```
SELECT * FROM user_table;
```

100 %

Results | Messages

	user_ID	team_ID	user_FName	user_LName	user_mob	user_email	user_role
1	US0000	TST	Aiyappa	Uthaiah	315-450-5845	akuthaia@syr.edu	C
2	US0001	FIN	James	McNulty	315-555-0164	mcnultyj@company.com	C
3	US0002	TST	Dexter	Morgan	315-555-0121	morgand@company.com	E
4	US0003	MKT	James	Hetfield	315-555-0162	hetfieldj@company.com	C
5	US0004	ACC	Bruce	Dickinson	315-555-0128	dickinsonb@company.com	C
6	US0005	FIN	Steven	Wilson	315-555-0157	wilsons@company.com	E
7	US0006	MAT	Mikael	Akerfeldt	315-555-0171	akerfeldtm@company.com	E
8	US0007	SAL	Ross	Geller	518-881-7232	gellerr@company.com	E
9	US0008	TEC	Monica	Geller	914-860-1092	gellerm@company.com	E
10	US0009	MKT	Walter	White	516-753-8034	whitew@company.com	C
11	US0010	TST	Barney	Stinson	646-216-0402	stinsonb@company.com	E
12	US0011	SAL	Joey	Tribbiani	607-429-6450	tribbianij@company.com	C
13	US0012	TEC	Pink	Floyd	518-744-2289	floydp@company.com	C
14	US0013	TEC	Kurt	Cobain	845-663-2248	cobaink@company.com	E
15	US0014	MAT	Michael	Scofield	585-402-5504	scofieldm@company.com	C
16	US0015	TEC	John	Locke	847-717-6576	lockej@company.com	E
17	US0016	MAT	Phoebe	Buffay	773-286-6121	buffayp@company.com	C
18	US0017	SAL	Akshay	Kumar	3157678989	akumar@syr.edu	C
19	USTST	TST	Test	User	315-450-5845	test@syr.com	E

INSERT DATA: login_cred

```
INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0001','mcnultyj','mcnultyjus0001');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0002','morgand','morgandus0002');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0003','hetfieldj','hetfieldjus0003');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0004','dickinsonb','dickinsonbus0004');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0005','wilsons','wilsonsus0005');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0006','akerfeldtm','akerfeldtmus0006');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0007','gellerr','gellerrus0007');
```

```
INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0008','gellerm','gellermus0008');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0009','whitew','whitewus0009');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0010','stinsonb','stinsonbus0010');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0011','tribbianij','tribbianijus0011');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0012','floydp','floydpus0012');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0013','cobaink','cobainkus0013');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0014','scofieldm','scofieldmus0014');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0015','lockej','lockejus0015');

INSERT INTO login_cred (user_ID,user_name,user_pwd)
VALUES ('US0016','buffayp','buffaypus0016');
```

```
Select * from login_cred;
```

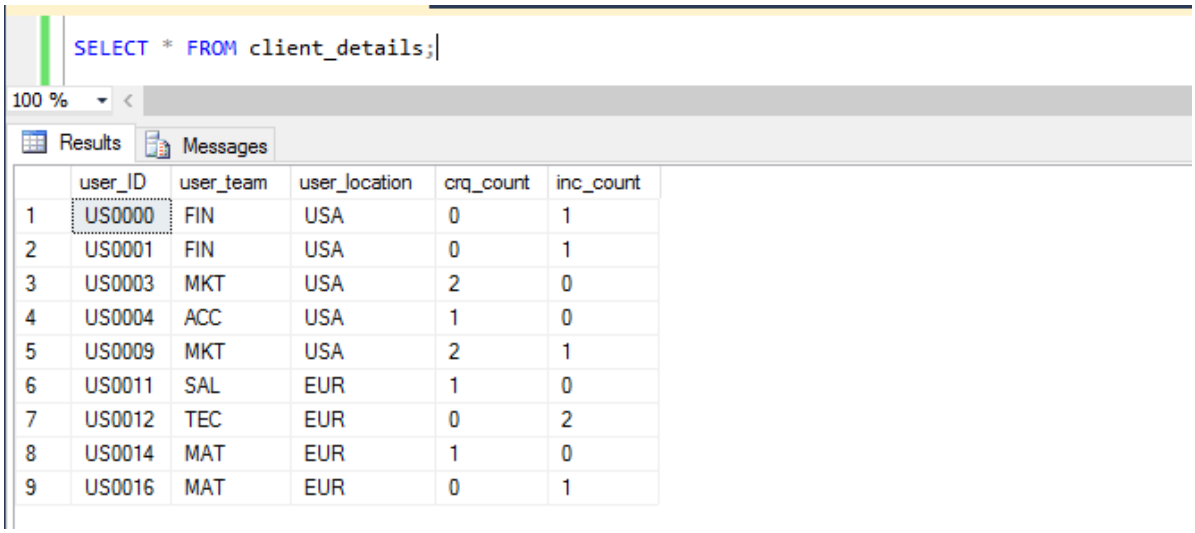
100 %

Results | Messages

	user_ID	user_name	user_pwd
1	US0000	aiyappa	aiyappa123
2	US0001	mcnultyj	mcnultyjus0001
3	US0002	morgand	morgandus0002
4	US0003	hetfieldj	hetfieldjus0003
5	US0004	dickinsonb	dickinsonbus0004
6	US0005	wilsons	wilsonsus0005
7	US0006	akerfeldtm	akerfeldtmus0006
8	US0007	gellerr	gellerrus0007
9	US0008	gellerm	gellermus0008
10	US0009	whitew	whitewus0009
11	US0010	stinsonb	stinsonbus0010
12	US0011	tribbianij	tribbianijus0011
13	US0012	floydp	floydpus0012
14	US0013	cobaink	cobainkus0013
15	US0014	scofieldm	scofieldmus0014
16	US0015	lockej	lockejus0015
17	US0016	buffayp	buffaypus0016
18	USTST	Test	Test

INSERT DATA: client_details

```
INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0001','FIN','USA');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0003','MKT','USA');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0004','ACC','USA');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0009','MKT','USA');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0011','SAL','EUR');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0012','TEC','EUR');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0014','MAT','EUR');

INSERT INTO client_details (user_ID,user_team,user_location)
VALUES ('US0016','MAT','EUR');
```



```
SELECT * FROM client_details;
```

100 %

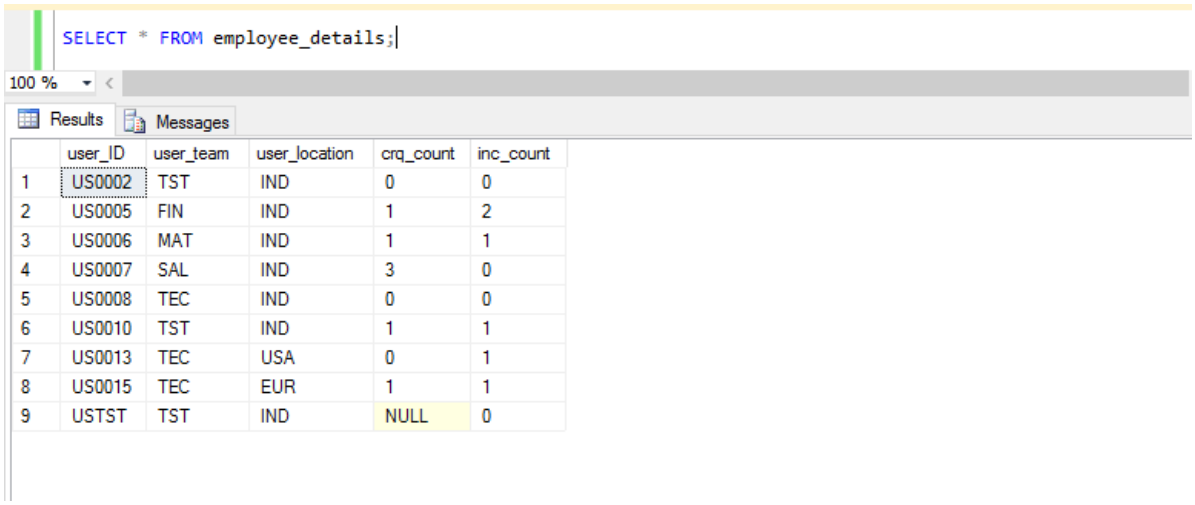
Results | Messages

	user_ID	user_team	user_location	crq_count	inc_count
1	US0000	FIN	USA	0	1
2	US0001	FIN	USA	0	1
3	US0003	MKT	USA	2	0
4	US0004	ACC	USA	1	0
5	US0009	MKT	USA	2	1
6	US0011	SAL	EUR	1	0
7	US0012	TEC	EUR	0	2
8	US0014	MAT	EUR	1	0
9	US0016	MAT	EUR	0	1

INSERT DATA: employee_details

```
INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0002','TST','IND');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0005','FIN','IND');
```

```
INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0006','MAT','IND');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0007','SAL','IND');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0008','TEC','IND');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0010','TST','IND');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0013','TEC','USA');

INSERT INTO employee_details (user_ID,user_team,user_location)
VALUES ('US0015','TEC','EUR');
```



```
SELECT * FROM employee_details;
```

100 %

Results | Messages

	user_ID	user_team	user_location	crq_count	inc_count
1	US0002	TST	IND	0	0
2	US0005	FIN	IND	1	2
3	US0006	MAT	IND	1	1
4	US0007	SAL	IND	3	0
5	US0008	TEC	IND	0	0
6	US0010	TST	IND	1	1
7	US0013	TEC	USA	0	1
8	US0015	TEC	EUR	1	1
9	USTST	TST	IND	NULL	0

INSERT DATA: ticket

```
INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000001','US0001','I');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000002','US0003','C');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000003','US0004','C');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000004','US0009','C');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000005','US0011','C');
```

```
INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000006','US0012','I');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000007','US0014','C');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000008','US0016','I');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000009','US0009','I');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000010','US0003','C');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000011','US0012','I');

INSERT INTO ticket (ticket_ID,owner_ID,ticket_type)
VALUES ('TKT0000012','US0009','C');
```

```
SELECT * FROM ticket;
```

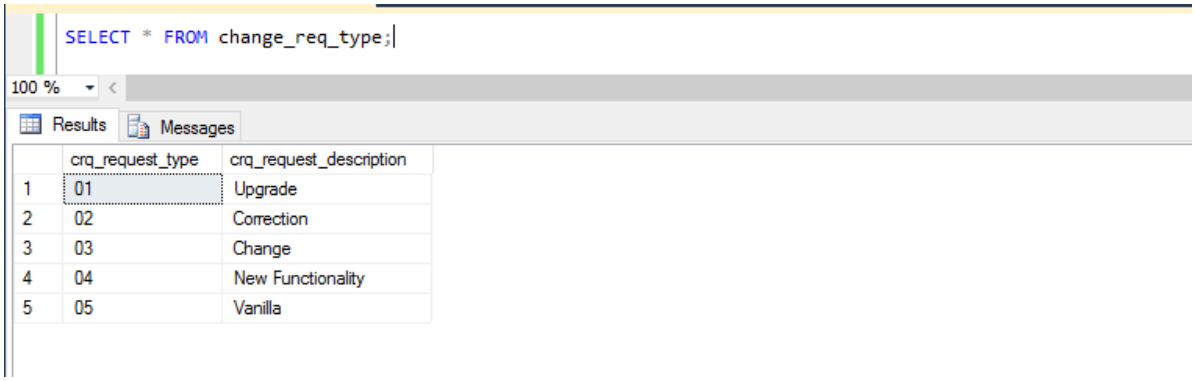
100 %

Results | Messages

	ticket_ID	owner_ID	ticket_type
1	TKT0000001	US0001	I
2	TKT0000002	US0003	C
3	TKT0000003	US0004	C
4	TKT0000004	US0009	C
5	TKT0000005	US0011	C
6	TKT0000006	US0012	I
7	TKT0000007	US0014	C
8	TKT0000008	US0016	I
9	TKT0000009	US0009	I
10	TKT0000010	US0003	C
11	TKT0000011	US0012	I
12	TKT0000012	US0009	C
13	TKT0000013	US0000	I

INSERT DATA: change_req_type

```
INSERT INTO change_req_type (crq_request_type,crq_request_description)
VALUES ('01','Upgrade');

INSERT INTO change_req_type (crq_request_type,crq_request_description)
VALUES ('02','Correction');

INSERT INTO change_req_type (crq_request_type,crq_request_description)
 VALUES ('03','Change');

INSERT INTO change_req_type (crq_request_type,crq_request_description)
VALUES ('04','New Functionality');

INSERT INTO change_req_type (crq_request_type,crq_request_description)
VALUES ('05','Vanilla');
```



	crq_request_type	crq_request_description
1	01	Upgrade
2	02	Correction
3	03	Change
4	04	New Functionality
5	05	Vanilla

INSERT DATA: change_req_status

```
INSERT INTO change_req_status (status, status_desc)
VALUES ('01','Created');

INSERT INTO change_req_status (status, status_desc)
VALUES ('02','Assigned');

INSERT INTO change_req_status (status, status_desc)
VALUES ('03','In Development');

INSERT INTO change_req_status (status, status_desc)
VALUES ('04','In Test');

INSERT INTO change_req_status (status, status_desc)
VALUES ('05','Delivered');

INSERT INTO change_req_status (status, status_desc)
VALUES ('06','Approved');

INSERT INTO change_req_status (status, status_desc)
VALUES ('07','Closed');
```

```
SELECT * FROM change_req_status;
```

100 %

Results | Messages

	status	status_desc
1	01	Created
2	02	Assigned
3	03	In Development
4	04	In Test
5	05	Delivered
6	06	Approved
7	07	Closed

INSERT DATA: change_req_priority

```
INSERT INTO change_req_priority
(crq_prority_scale,crq_priority_desc,crq_priority_duration)
VALUES ('01','Low',5);

INSERT INTO change_req_priority
(crq_prority_scale,crq_priority_desc,crq_priority_duration)
VALUES ('02','Medium',10);

INSERT INTO change_req_priority
(crq_prority_scale,crq_priority_desc,crq_priority_duration)
VALUES ('03','High',20);

INSERT INTO change_req_priority
(crq_prority_scale,crq_priority_desc,crq_priority_duration)
VALUES ('04','Very High',30);
```

```
SELECT * FROM change_req_priority;
```

100 %

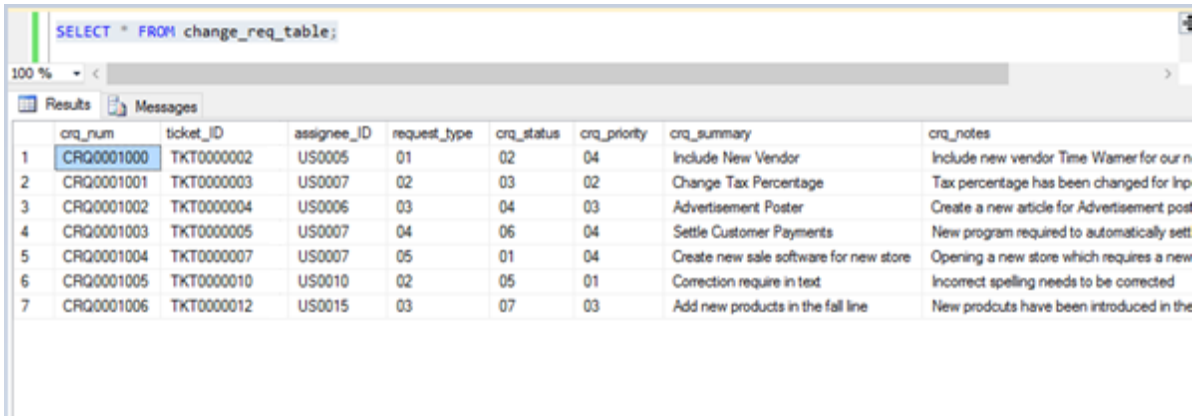
Results | Messages

	crq_prority_scale	crq_priority_desc	crq_priority_duration
1	01	Low	5
2	02	Medium	10
3	03	High	20
4	04	Very High	30

INSERT DATA: change_req_table

```
INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start)
VALUES ('CRQ0001000','TKT0000002','US0005','01','02','04','Include New
Vendor','Include new vendor Time Warner for our new office internet connection','Test
by creating a Purchase Order','NA',11/10/2016,11/15/2016,12/15/2016,11/20/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start)
VALUES ('CRQ0001001','TKT0000003','US0007','02','03','02','Change Tax
Percentage','Tax percentage has been changed for Input Tax from 10% to 5%','Test by
posting a new Invoice','NA',11/5/2016,11/10/2016,11/20/2016,11/12/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start)
VALUES ('CRQ0001002','TKT0000004','US0006','03','04','03','Advertisement
Poster','Create a new article for Advertisement posters','Test by creating a purchase
order with the net article','NA',10/25/2016,10/30/2016,11/19/2016,11/10/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start)
VALUES ('CRQ0001003','TKT0000005','US0007','04','06','04','Settle Customer
Payments','New program required to automatically settle customer payments','Test
Document to be provided','NA',10/10/2016,10/15/2016,11/14/2016,10/10/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end)
VALUES ('CRQ0001004','TKT0000007','US0007','05','01','04','Create new sale software
for new store','Opening a new store which requires a new software for the sale from
that store','Run software','NA',11/25/2016,11/30/2016,12/30/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start)
VALUES ('CRQ0001005','TKT0000010','US0010','02','05','01','Correction require in
text','Incorrect spelling needs to be corrected','Test by runnuing the script after
correction','NA',11/15/2016,11/15/2016,11/20/2016,11/15/2016);

INSERT INTO change_req_table
(crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_n
otes,crq_test_instructions,crq_knowledge_transfer,crq_creation_date,crq_est_start,crq
_est_end,crq_actual_start,crq_actual_end)
VALUES ('CRQ0001006','TKT0000012','US0015','03','07','03','Add new products in the
fall line','New prodcuts have been introduced in the fall line which have to be
```

```
added','Test by creating sale
orders','NA',11/5/2016,11/5/2016,11/25/2016,11/5/2016,11/25/2016);
```

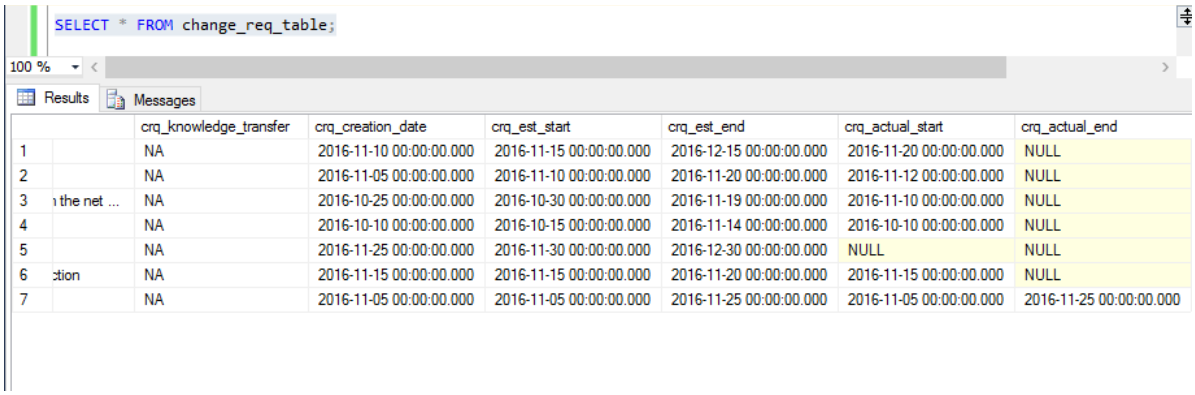


```
SELECT * FROM change_req_table;
```

100 %

Results | Messages

	crq_num	ticket_ID	assignee_ID	request_type	crq_status	crq_priority	crq_summary	crq_notes
1	CRQ0001000	TKT0000002	US0005	01	02	04	Include New Vendor	Include new vendor Time Warner for our n
2	CRQ0001001	TKT0000003	US0007	02	03	02	Change Tax Percentage	Tax percentage has been changed for Inp
3	CRQ0001002	TKT0000004	US0006	03	04	03	Advertisement Poster	Create a new article for Advertisement post
4	CRQ0001003	TKT0000005	US0007	04	06	04	Settle Customer Payments	New program required to automatically sett
5	CRQ0001004	TKT0000007	US0007	05	01	04	Create new sale software for new store	Opening a new store which requires a new
6	CRQ0001005	TKT0000010	US0010	02	05	01	Correction require in text	Incorrect spelling needs to be corrected
7	CRQ0001006	TKT0000012	US0015	03	07	03	Add new products in the fall line	New prodcuts have been introduced in the



```
SELECT * FROM change_req_table;
```

100 %

Results | Messages

		crq_knowledge_transfer	crq_creation_date	crq_est_start	crq_est_end	crq_actual_start	crq_actual_end
1		NA	2016-11-10 00:00:00.000	2016-11-15 00:00:00.000	2016-12-15 00:00:00.000	2016-11-20 00:00:00.000	NULL
2		NA	2016-11-05 00:00:00.000	2016-11-10 00:00:00.000	2016-11-20 00:00:00.000	2016-11-12 00:00:00.000	NULL
3	ı the net ...	NA	2016-10-25 00:00:00.000	2016-10-30 00:00:00.000	2016-11-19 00:00:00.000	2016-11-10 00:00:00.000	NULL
4		NA	2016-10-10 00:00:00.000	2016-10-15 00:00:00.000	2016-11-14 00:00:00.000	2016-10-10 00:00:00.000	NULL
5		NA	2016-11-25 00:00:00.000	2016-11-30 00:00:00.000	2016-12-30 00:00:00.000	NULL	NULL
6	ction	NA	2016-11-15 00:00:00.000	2016-11-15 00:00:00.000	2016-11-20 00:00:00.000	2016-11-15 00:00:00.000	NULL
7		NA	2016-11-05 00:00:00.000	2016-11-05 00:00:00.000	2016-11-25 00:00:00.000	2016-11-05 00:00:00.000	2016-11-25 00:00:00.000

INSERT DATA: incident_impact

```
INSERT INTO incident_impact
(impact_scale,impact_description,inc_priority_desc,inc_priority_duration)
VALUES ('01','Business Affected and not functioning','Very High',5);

INSERT INTO incident_impact
(impact_scale,impact_description,inc_priority_desc,inc_priority_duration)
VALUES ('02','Business Affected but functioning','High',10);

INSERT INTO incident_impact
(impact_scale,impact_description,inc_priority_desc,inc_priority_duration)
VALUES ('03','Process Affected','Medium',12);

INSERT INTO incident_impact
(impact_scale,impact_description,inc_priority_desc,inc_priority_duration)
VALUES ('04','Low Effect','Low',15);
```

```
SELECT * FROM incident_impact;
```

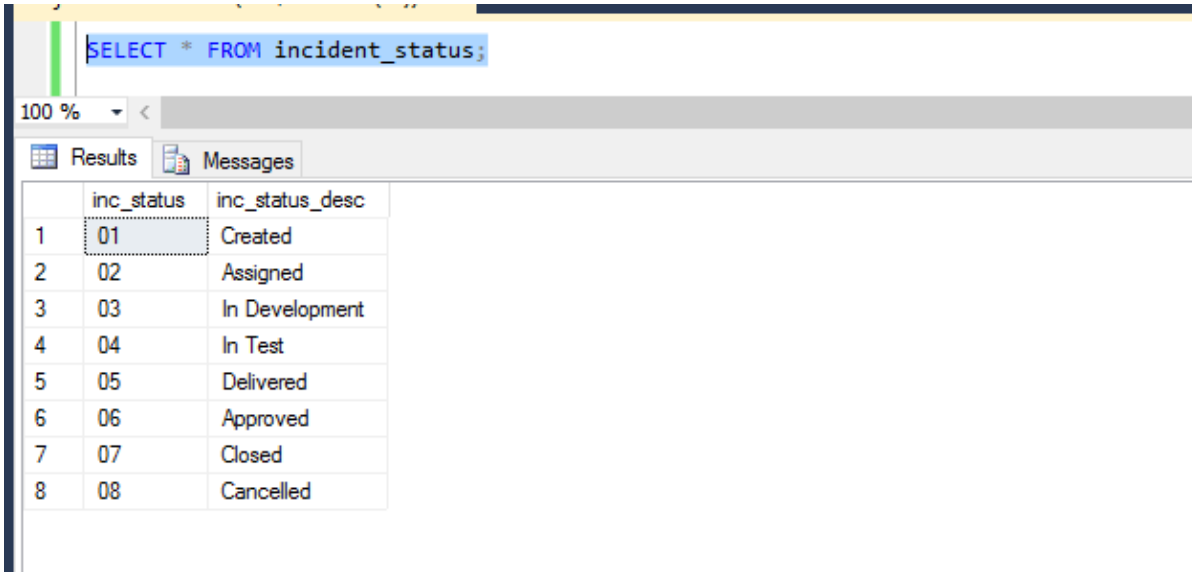
100 %

Results | Messages

	impact_scale	impact_description	inc_priority_desc	inc_priority_duration
1	01	Business Affected and not functioning	Very High	5
2	02	Business Affected but functioning	High	10
3	03	Process Affected	Medium	12
4	04	Low Effect	Low	15

INSERT DATA: incident_status

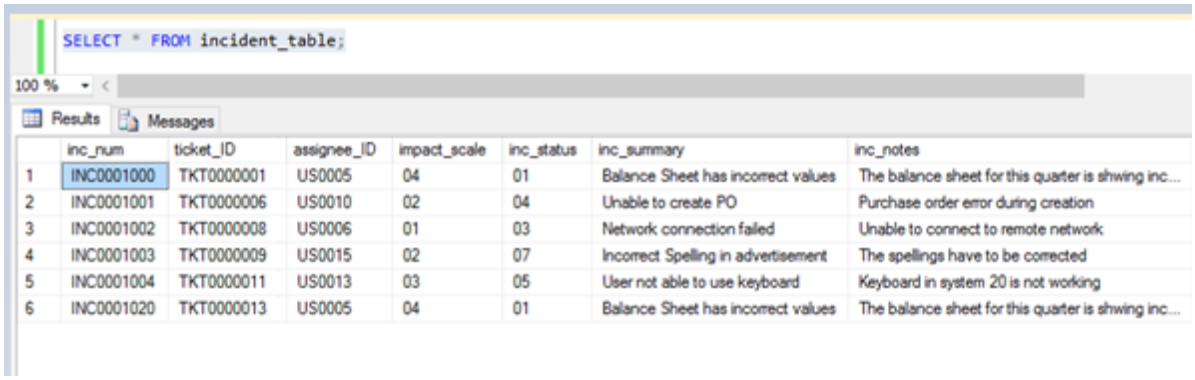
```
INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('01','Created');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('02','Assigned');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('03','In Development');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('04','In Test');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('05','Delivered');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('06','Approved');

INSERT INTO incident_status (inc_status,inc_status_desc)
VALUES ('07','Closed');
```



	inc_status	inc_status_desc
1	01	Created
2	02	Assigned
3	03	In Development
4	04	In Test
5	05	Delivered
6	06	Approved
7	07	Closed
8	08	Cancelled

INSERT DATA: incident_table

```
INSERT INTO incident_table
(inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_crea
tion_date)
VALUES ('INC0001000','TKT0000001','US0005','04','01','Balance Sheet has incorrect
values','The balance sheet for this quarter is shwing incorrect values',11/25/2016);

INSERT INTO incident_table
(inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_crea
tion_date,inc_start_date)
VALUES ('INC0001001','TKT0000006','US0010','02','04','Unable to create PO','Purchase
order error during creation',11/23/2016,11/23/2016);

INSERT INTO incident_table
(inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_crea
tion_date,inc_start_date)
VALUES ('INC0001002','TKT0000008','US0006','01','03','Network connection
failed','Unable to connect to remote network',11/26/2016,11/26/2016);

INSERT INTO incident_table
(inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_reas
on,inc_resolution,inc_creation_date,inc_start_date,inc_end_date)
VALUES ('INC0001003','TKT0000009','US0015','02','07','Incorrect Spelling in
advertisement','The spellings have to be corrected','Incorrect spelling while
creating the advertisement','Spelling corrected in the
script',11/15/2016,11/16/2016,11/16/2016);

INSERT INTO incident_table
(inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_crea
tion_date,inc_start_date)
VALUES ('INC0001004','TKT0000011','US0013','03','05','User not able to use
keyboard','Keyboard in system 20 is not working',11/20/2016,11/21/2016);
```

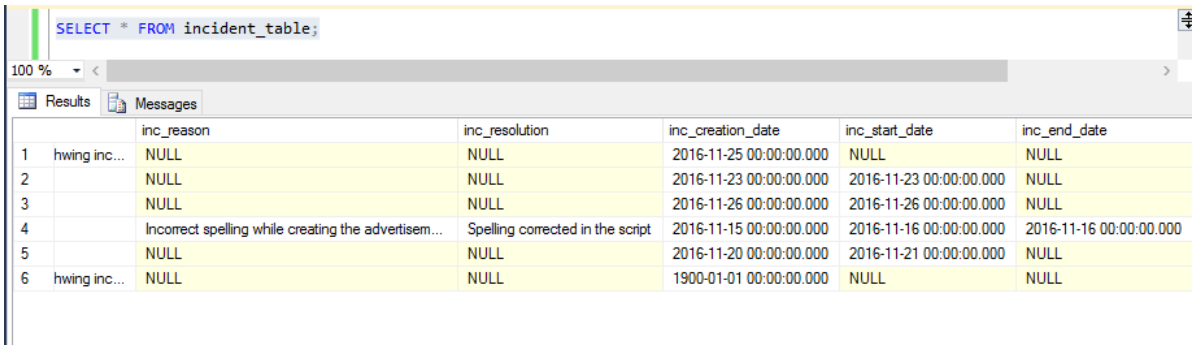


```
SELECT * FROM incident_table;
```

100 %

Results | Messages

	inc_num	ticket_ID	assignee_ID	impact_scale	inc_status	inc_summary	inc_notes
1	INC0001000	TKT0000001	US0005	04	01	Balance Sheet has incorrect values	The balance sheet for this quarter is shwing inc...
2	INC0001001	TKT0000006	US0010	02	04	Unable to create PO	Purchase order error during creation
3	INC0001002	TKT0000008	US0006	01	03	Network connection failed	Unable to connect to remote network
4	INC0001003	TKT0000009	US0015	02	07	Incorrect Spelling in advertisement	The spellings have to be corrected
5	INC0001004	TKT0000011	US0013	03	05	User not able to use keyboard	Keyboard in system 20 is not working
6	INC0001020	TKT0000013	US0005	04	01	Balance Sheet has incorrect values	The balance sheet for this quarter is shwing inc...



```
SELECT * FROM incident_table;
```

100 %

Results | Messages

		inc_reason	inc_resolution	inc_creation_date	inc_start_date	inc_end_date
1	hwing inc...	NULL	NULL	2016-11-25 00:00:00.000	NULL	NULL
2		NULL	NULL	2016-11-23 00:00:00.000	2016-11-23 00:00:00.000	NULL
3		NULL	NULL	2016-11-26 00:00:00.000	2016-11-26 00:00:00.000	NULL
4		Incorrect spelling while creating the advertisem...	Spelling corrected in the script	2016-11-15 00:00:00.000	2016-11-16 00:00:00.000	2016-11-16 00:00:00.000
5		NULL	NULL	2016-11-20 00:00:00.000	2016-11-21 00:00:00.000	NULL
6	hwing inc...	NULL	NULL	1900-01-01 00:00:00.000	NULL	NULL

MAJOR DATA QUESTIONS

The users of my database consist primarily of:

- Clients
- Employees

Clients will use the ticket management system to raise tickets for any work that has to be delegated to the service company. Employee from the service company will work on the tickets raised by the client and based on the status they will be able to view, modify and edit tickets that are assigned to them.

Why the Client and Employee Queries the Database

- The Clients must have the rights to create the two types of tickets

  The clients will be able to create new incidents and change requests. They will only be able to view created tickets but they will not be able to modify them.

  Whenever the client created a ticket it is stored in either the Change Request table ‘change_req_table’ or the Incident table ‘incident_table’.

  The login form will help determine what operations that client can perform.

- The Employee will have the rights to edit the two types of tickets

  The employee will be able to edit incidents and change requests that the client has created. They will only be able to edit and view created tickets but they will not be able to create new ones.

  Whenever the employee updates a ticket it is stored in either the Change Request table ‘change_req_table’ or the Incident table ‘incident_table’.

  The login form will help determine what operations that employee can perform.

Count of Incidents and Change Requests

The count will give an idea as to the volume of tickets that each employee has and the number of tickets raised by the client

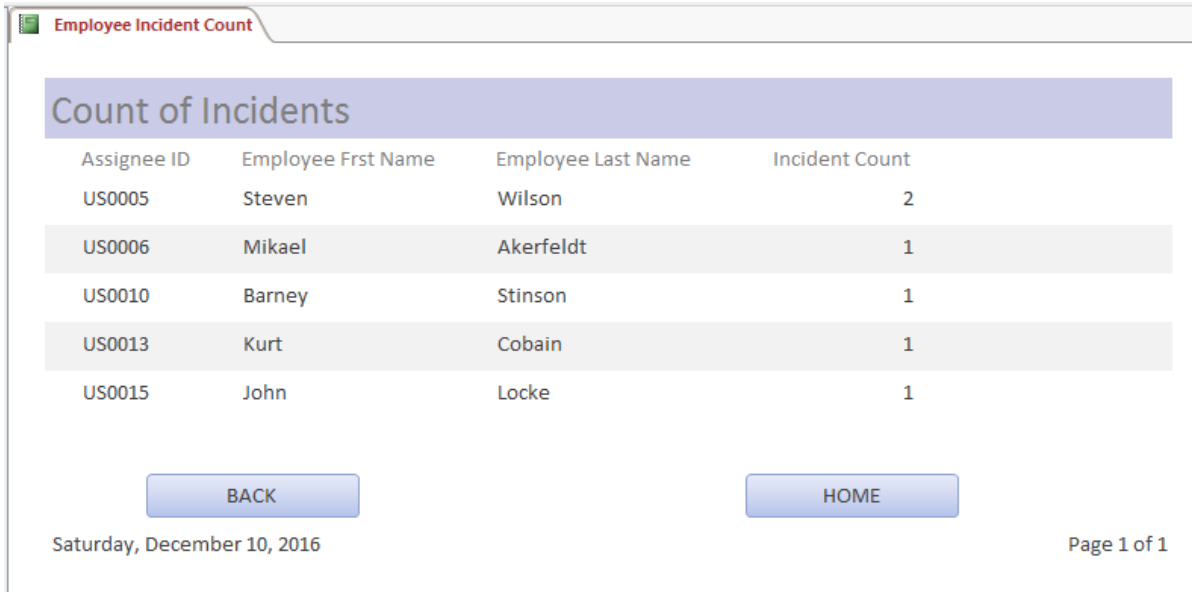
Employee Incident Count

```
--Employee Incident Count
SELECT incident_table.assignee_ID, Count(incident_table.inc_num) AS CountOfinc_num, user_table.user_FName, user_table.user_LName
FROM user_table INNER JOIN incident_table ON user_table.user_ID = incident_table.assignee_ID
GROUP BY incident_table.assignee_ID, user_table.user_FName, user_table.user_LName;
```

94 %

Results | Messages

	assignee_ID	CountOfinc_num	user_FName	user_LName
1	US0005	2	Steven	Wilson
2	US0006	1	Mikael	Akerfeldt
3	US0010	1	Barney	Stinson
4	US0013	1	Kurt	Cobain
5	US0015	1	John	Locke



Employee Change Request Count

```
--Employee Change Request Count
SELECT user_table.user_FName, user_table.user_LName, change_req_table.assignee_ID, Count(change_req_table.crq_num) AS CountOfcrq_num
FROM user_table INNER JOIN change_req_table ON user_table.user_ID = change_req_table.assignee_ID
GROUP BY user_table.user_FName, user_table.user_LName, change_req_table.assignee_ID;
```

90 %

Results | Messages

	user_FName	user_LName	assignee_ID	CountOfcrq_num
1	Steven	Wilson	US0005	1
2	Mikael	Akerfeldt	US0006	1
3	Ross	Geller	US0007	3
4	Barney	Stinson	US0010	1
5	John	Locke	US0015	1

Employee Change Request Count

Employee Change Request Count

Assignee ID	Employee First Name	Employee Last Name	Change Request Count
US0005	Steven	Wilson	1
US0006	Mikael	Akerfeldt	1
US0007	Ross	Geller	3
US0010	Barney	Stinson	1
US0015	John	Locke	1

BACK HOME

Saturday, December 10, 2016 Page 1 of 1

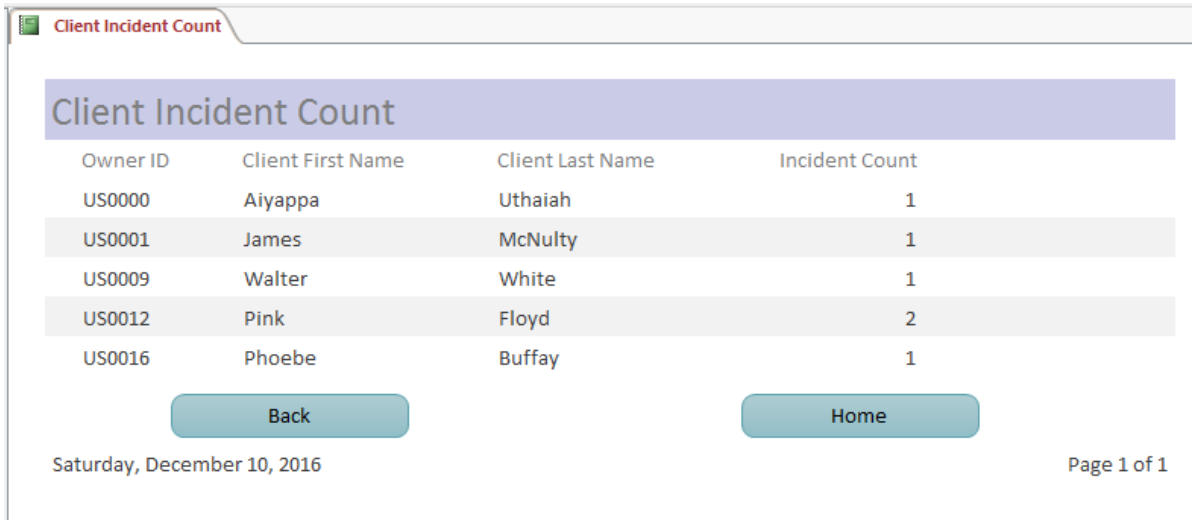
Client Incident Count

```
--Client Incident Count
SELECT incident_table.owner_ID, user_table.user_FName, user_table.user_LName, Count(incident_table.inc_num) AS CountOfinc_num
FROM user_table INNER JOIN incident_table ON user_table.user_ID = incident_table.owner_ID
GROUP BY incident_table.owner_ID, user_table.user_FName, user_table.user_LName;
```

90 %

Results | Messages

	owner_ID	user_FName	user_LName	CountOfinc_num
1	US0000	Aiyappa	Uthaiah	1
2	US0001	James	McNulty	1
3	US0009	Walter	White	1
4	US0012	Pink	Floyd	2
5	US0016	Phoebe	Buffay	1



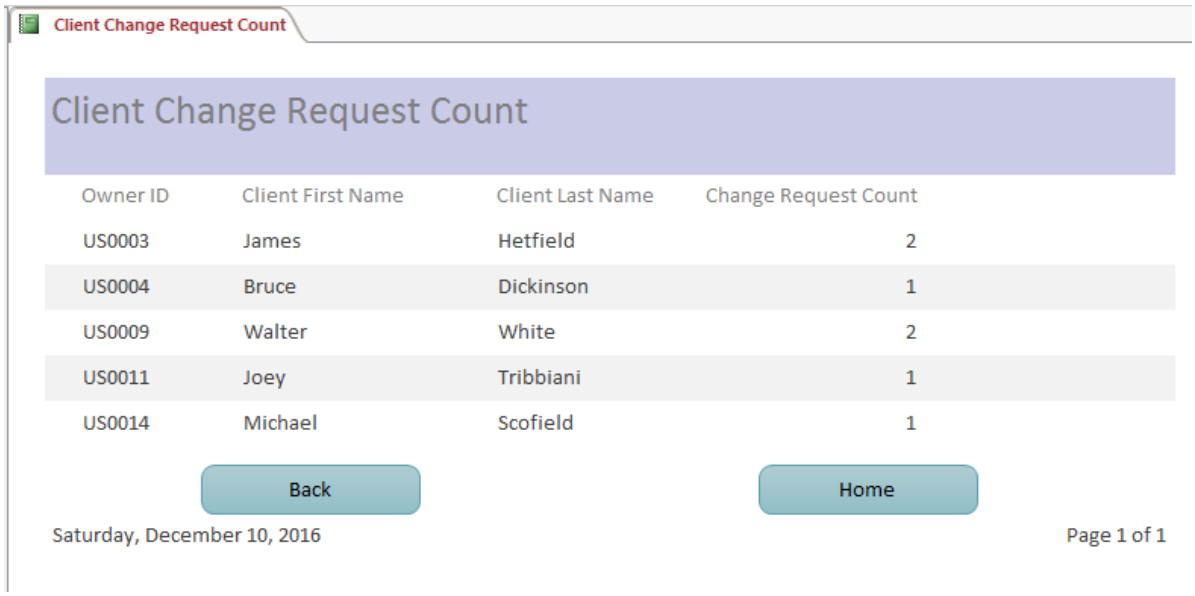
Client Change Request Count

```
--Client Change Request Count
SELECT user_table.user_FName, user_table.user_LName, change_req_table.owner_ID, Count(change_req_table.crq_num) AS CountOfcrq_num
FROM user_table INNER JOIN change_req_table ON user_table.user_ID = change_req_table.owner_ID
GROUP BY user_table.user_LName, change_req_table.owner_ID, user_table.user_FName;
```

90 %

Results | Messages

	user_FName	user_LName	owner_ID	CountOfcrq_num
1	James	Hetfield	US0003	2
2	Bruce	Dickinson	US0004	1
3	Walter	White	US0009	2
4	Joey	Tribbiani	US0011	1
5	Michael	Scofield	US0014	1



The Client and Employee should be able to view all the open and closed Change requests and Incidents.

Open Change Request List

```
SELECT * FROM change_req_table
WHERE crq_status <>'07';
```

100 %

Results | Messages

	crq_num	ticket_ID	owner_ID	assignee_ID	request_type	crq_status	crq_priority	crq_summary	crq_notes
1	CRQ0001000	TKT0000002	US0003	US0005	01	02	04	Include New Vendor	Include new vendor Time Warner
2	CRQ0001001	TKT0000003	US0004	US0007	02	03	02	Change Tax Percentage	Tax percentage has been chang
3	CRQ0001002	TKT0000004	US0009	US0006	03	04	03	Advertisement Poster	Create a new article for Advertise
4	CRQ0001003	TKT0000005	US0011	US0007	04	06	04	Settle Customer Payments	New program required to automat
5	CRQ0001004	TKT0000007	US0014	US0007	05	01	04	Create new sale software for new store	Opening a new store which requi
6	CRQ0001005	TKT0000010	US0003	US0010	02	05	01	Correction require in text	Incorrect spelling needs to be cor

```
SELECT * FROM change_req_table
WHERE crq_status <>'07';
```

100 %

Results | Messages

		crq_knowledge_transfer	crq_creation_date	crq_est_start	crq_est_end	crq_actual_start	crq_actual_end
1	Order	NA	2016-11-10 00:00:00.000	2016-11-15 00:00:00.000	2016-12-15 00:00:00.000	2016-11-20 00:00:00.000	NULL
2	;e	NA	2016-11-05 00:00:00.000	2016-11-10 00:00:00.000	2016-11-20 00:00:00.000	2016-11-12 00:00:00.000	NULL
3	order with the net ...	NA	2016-10-25 00:00:00.000	2016-10-30 00:00:00.000	2016-11-19 00:00:00.000	2016-11-10 00:00:00.000	NULL
4	ed	NA	2016-10-10 00:00:00.000	2016-10-15 00:00:00.000	2016-11-14 00:00:00.000	2016-10-10 00:00:00.000	NULL
5		NA	2016-11-25 00:00:00.000	2016-11-30 00:00:00.000	2016-12-30 00:00:00.000	NULL	NULL
6	fter correction	NA	2016-11-15 00:00:00.000	2016-11-15 00:00:00.000	2016-11-20 00:00:00.000	2016-11-15 00:00:00.000	NULL

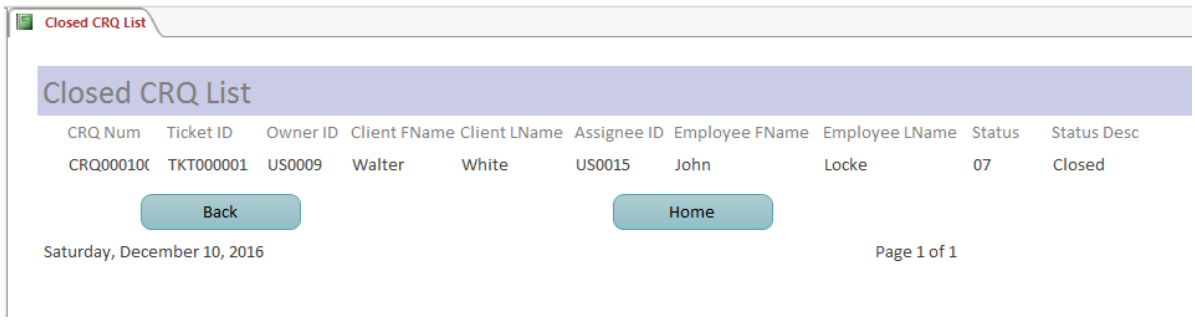
Open Change Request List

Open Change Request List

CRQ Number	Ticket ID	Owner ID	Client Details	Client Details	Assignee ID	Employee FName	Employee LName	Status	Status Desc
CRQ0001000	TKT0000002	US0003	James	Hetfield	US0005	Steven	Wilson	02	Assigned
CRQ0001001	TKT0000003	US0004	Bruce	Dickinson	US0007	Ross	Geller	03	In Developm
CRQ0001002	TKT0000004	US0009	Walter	White	US0006	Mikael	Akerfeldt	04	In Test
CRQ0001003	TKT0000005	US0011	Joey	Tribbiani	US0007	Ross	Geller	06	Approved
CRQ0001004	TKT0000007	US0014	Michael	Scofield	US0007	Ross	Geller	01	Created
CRQ0001005	TKT0000010	US0003	James	Hetfield	US0010	Barney	Stinson	05	Delivered

Back Home

Saturday, December 10, 2016 Page 1 of 1



Open Incident List

```
SELECT * FROM incident_table
WHERE inc_status <>'07';
```

100 %

Results | Messages

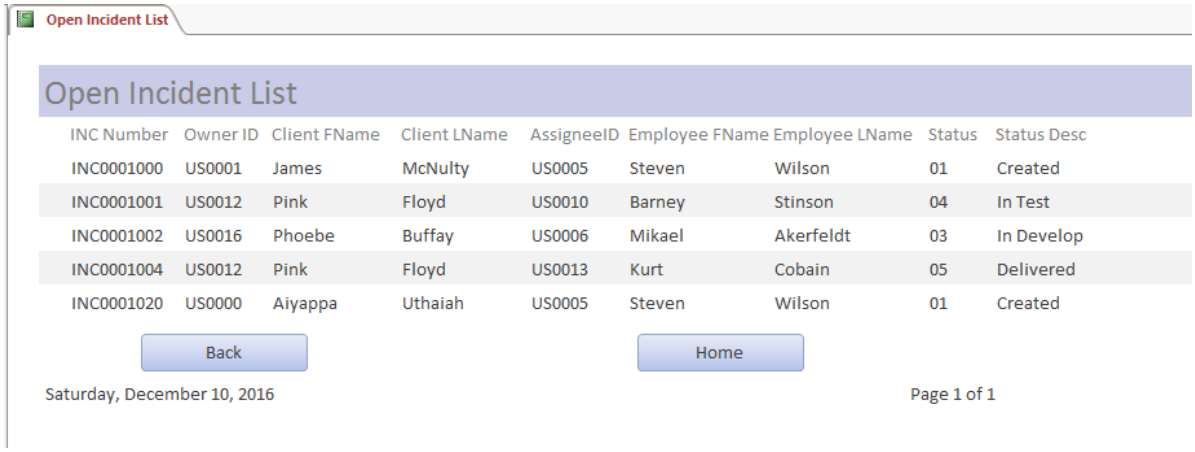
	inc_num	ticket_ID	owner_ID	assignee_ID	impact_scale	inc_status	inc_summary	inc_notes
1	INC0001000	TKT0000001	US0001	US0005	04	01	Balance Sheet has incorrect values	The balance sheet for this quarter is shwing inc...
2	INC0001001	TKT0000006	US0012	US0010	02	04	Unable to create PO	Purchase order error during creation
3	INC0001002	TKT0000008	US0016	US0006	01	03	Network connection failed	Unable to connect to remote network
4	INC0001004	TKT0000011	US0012	US0013	03	05	User not able to use keyboard	Keyboard in system 20 is not working
5	INC0001020	TKT0000013	US0000	US0005	04	01	Balance Sheet has incorrect values	The balance sheet for this quarter is shwing inc...

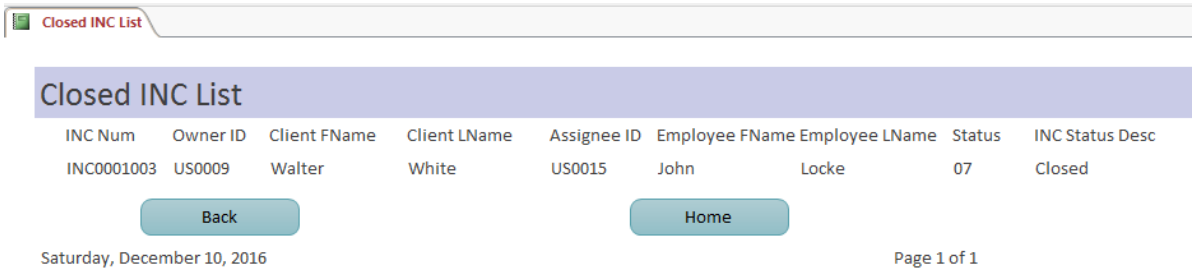
```
SELECT * FROM incident_table
WHERE inc_status <>'07';
```

100 %

Results | Messages

	y	inc_notes	inc_reason	inc_resolution	inc_creation_date	inc_start_date	inc_end_date
1	heet has incorrect values	The balance sheet for this quarter is shwing inc...	NULL	NULL	2016-11-25 00:00:00.000	NULL	NULL
2	reate PO	Purchase order error during creation	NULL	NULL	2016-11-23 00:00:00.000	2016-11-23 00:00:00.000	NULL
3	nnection failed	Unable to connect to remote network	NULL	NULL	2016-11-26 00:00:00.000	2016-11-26 00:00:00.000	NULL
4	le to use keyboard	Keyboard in system 20 is not working	NULL	NULL	2016-11-20 00:00:00.000	2016-11-21 00:00:00.000	NULL
5	heet has incorrect values	The balance sheet for this quarter is shwing inc...	NULL	NULL	1900-01-01 00:00:00.000	NULL	NULL


Open Incident List
Open Incident List
INC Number Owner ID Client FName Client LName AssigneeID Employee FName Employee LName Status Status Desc
INC0001000 US0001 James McNulty US0005 Steven Wilson 01 Created
INC0001001 US0012 Pink Floyd US0010 Barney Stinson 04 In Test
INC0001002 US0016 Phoebe Buffay US0006 Mikael Akerfeldt 03 In Develop
INC0001004 US0012 Pink Floyd US0013 Kurt Cobain 05 Delivered
INC0001020 US0000 Aiyappa Uthaiah US0005 Steven Wilson 01 Created
Back
Home
Saturday, December 10, 2016
Page 1 of 1

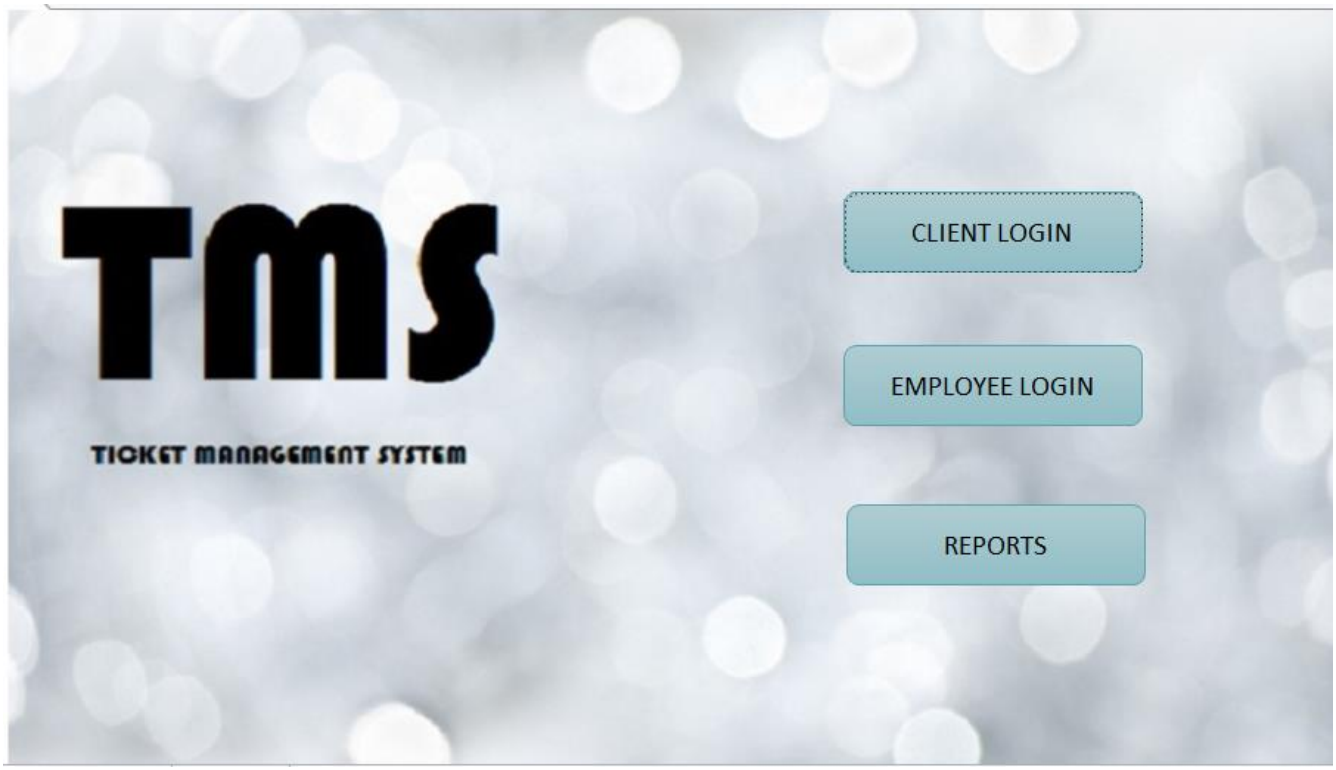

Closed INC List
Closed INC List
INC Num Owner ID Client FName Client LName Assignee ID Employee FName Employee LName Status INC Status Desc
INC0001003 US0009 Walter White US0015 John Locke 07 Closed
Back
Home
Saturday, December 10, 2016
Page 1 of 1

FORMS

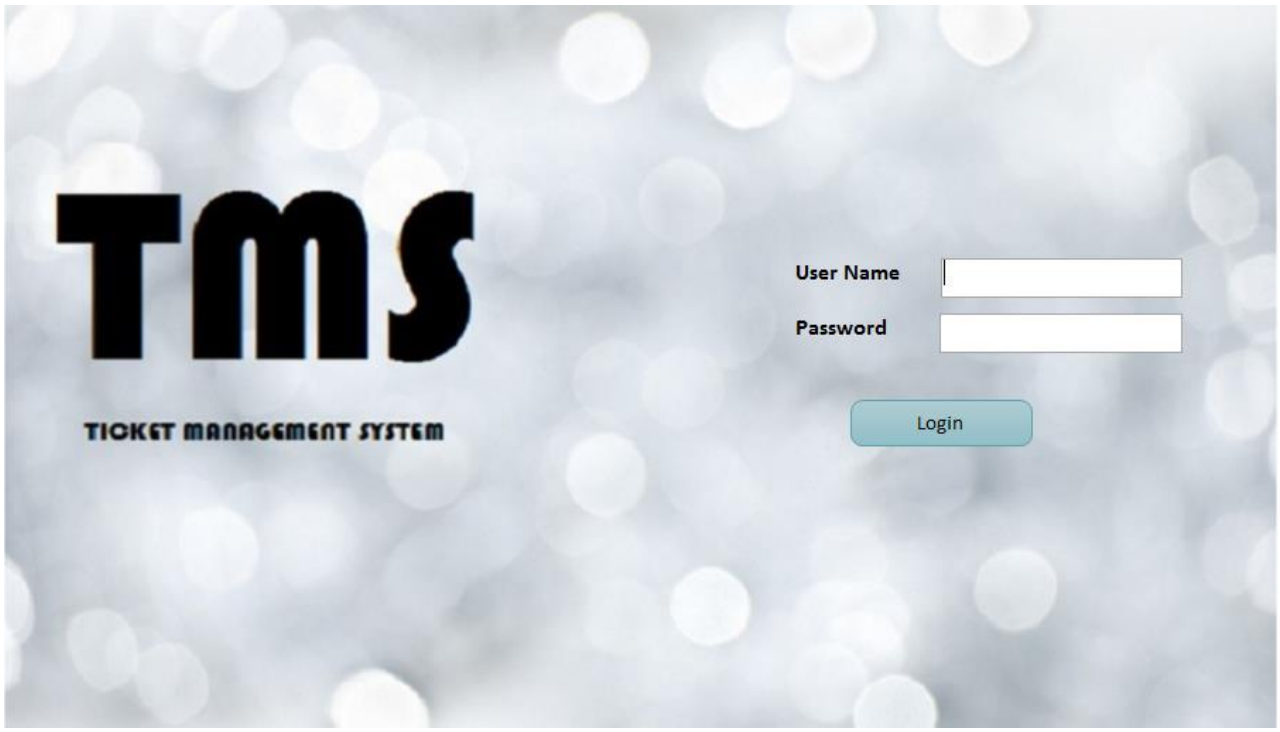
The ticket management system has two types of users:

- Clients
- Employees

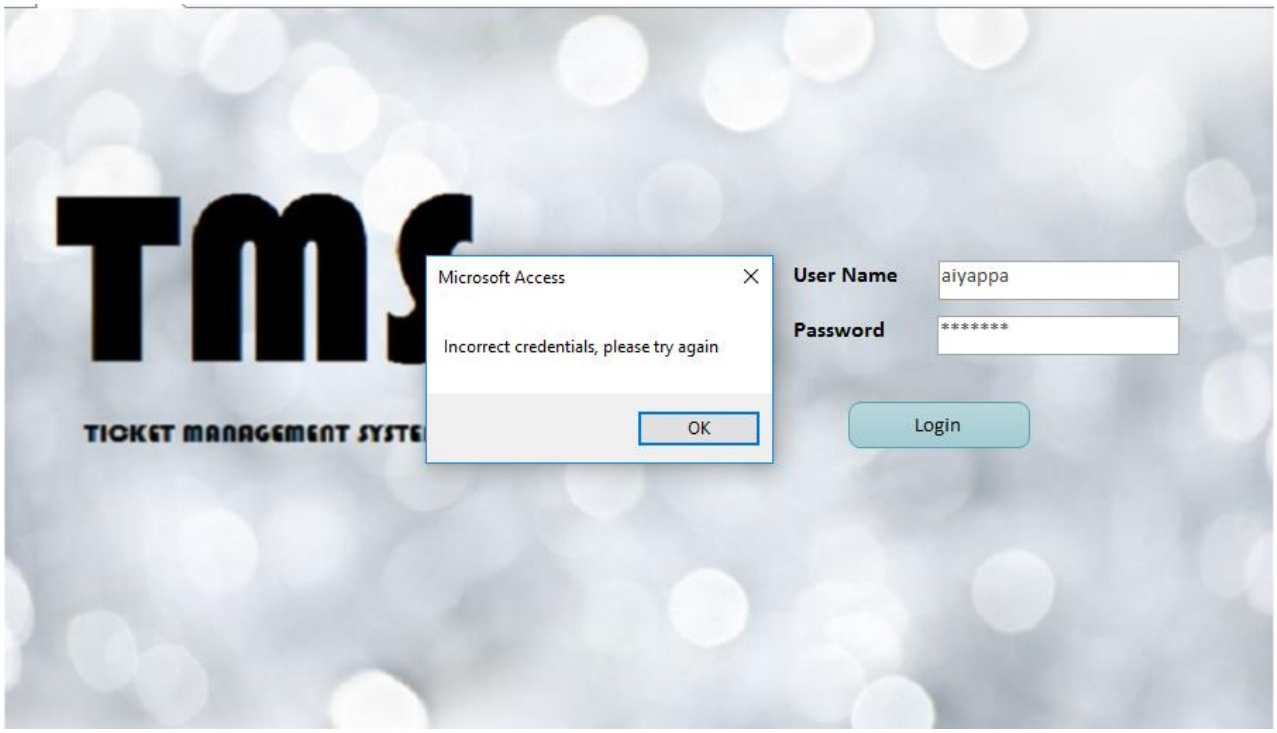
On starting the application the users will get the option of client login, employee login and report screen.



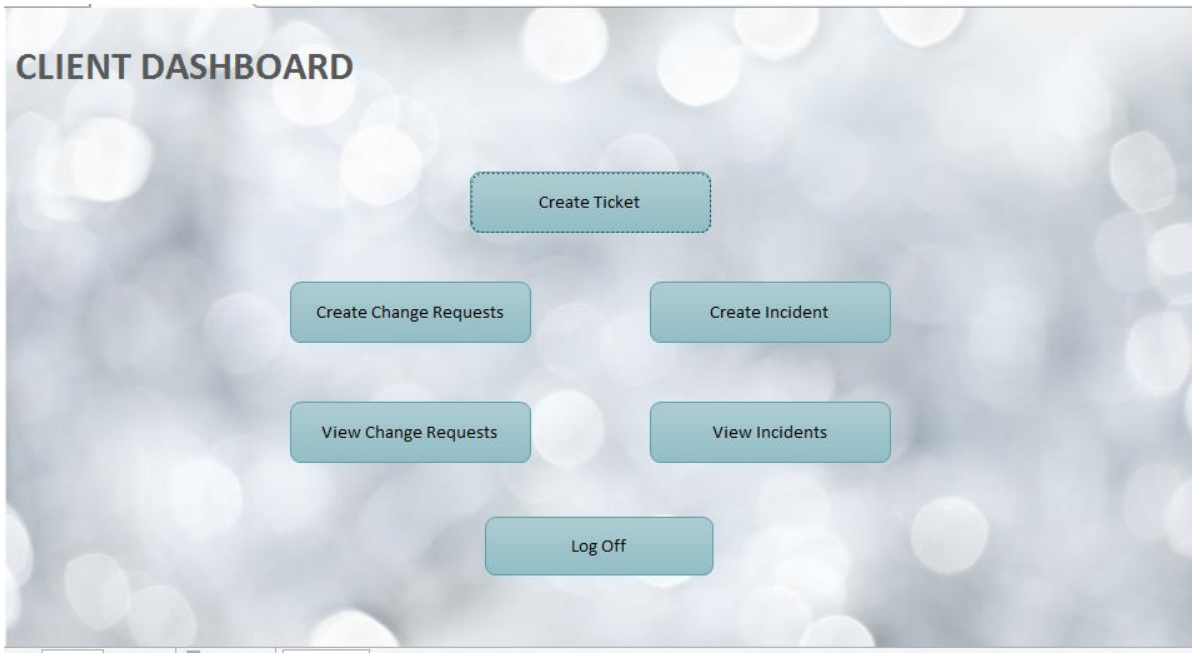
The Login screen verifies the username and password from the login_cred database and verifies the credentials



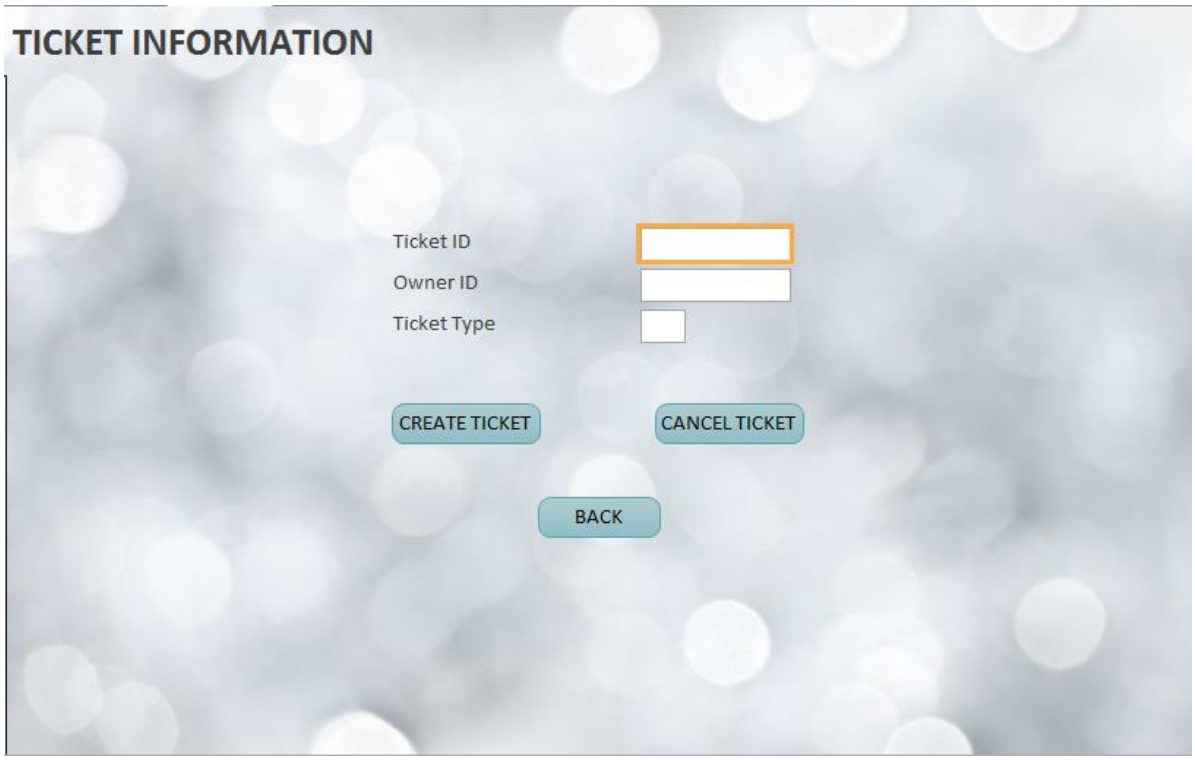
If the credentials are incorrect



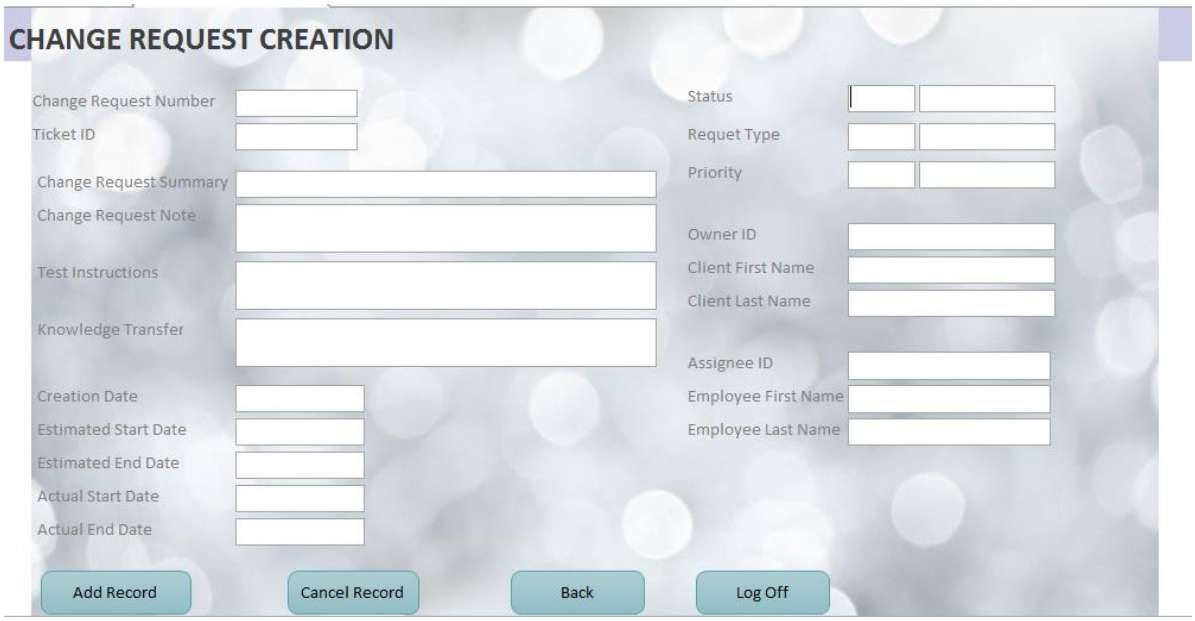
A verified Client login will provide the user with the below options.

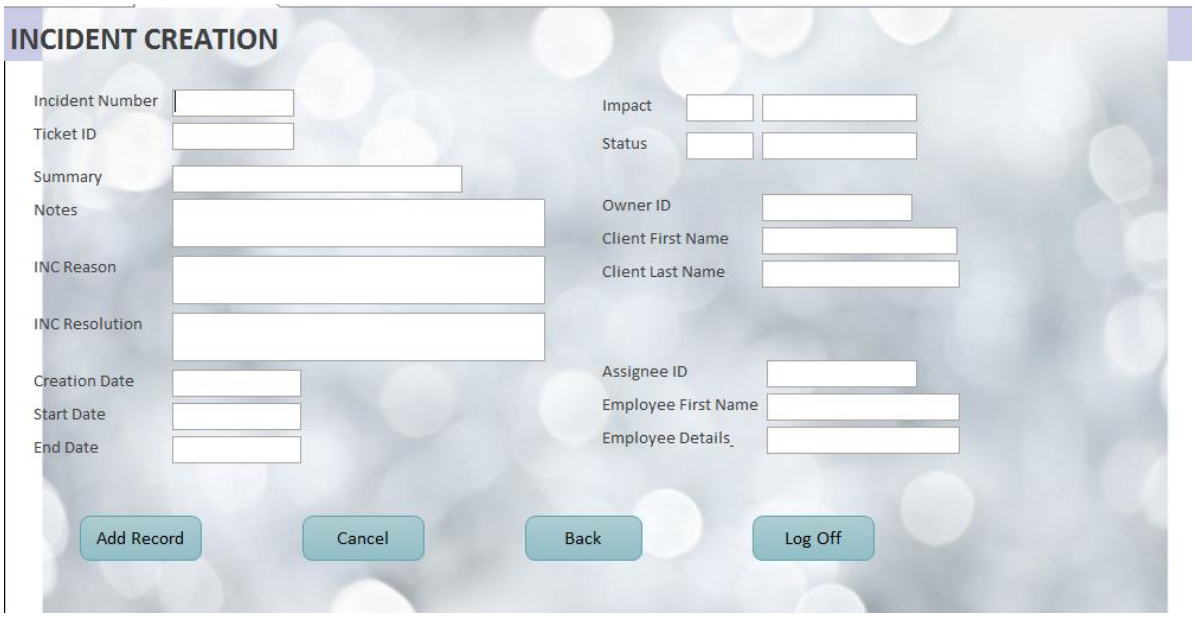


Create Ticket (Only for Clients) – The Clint has to create a ticket first before assigning it to a change request or an incident.

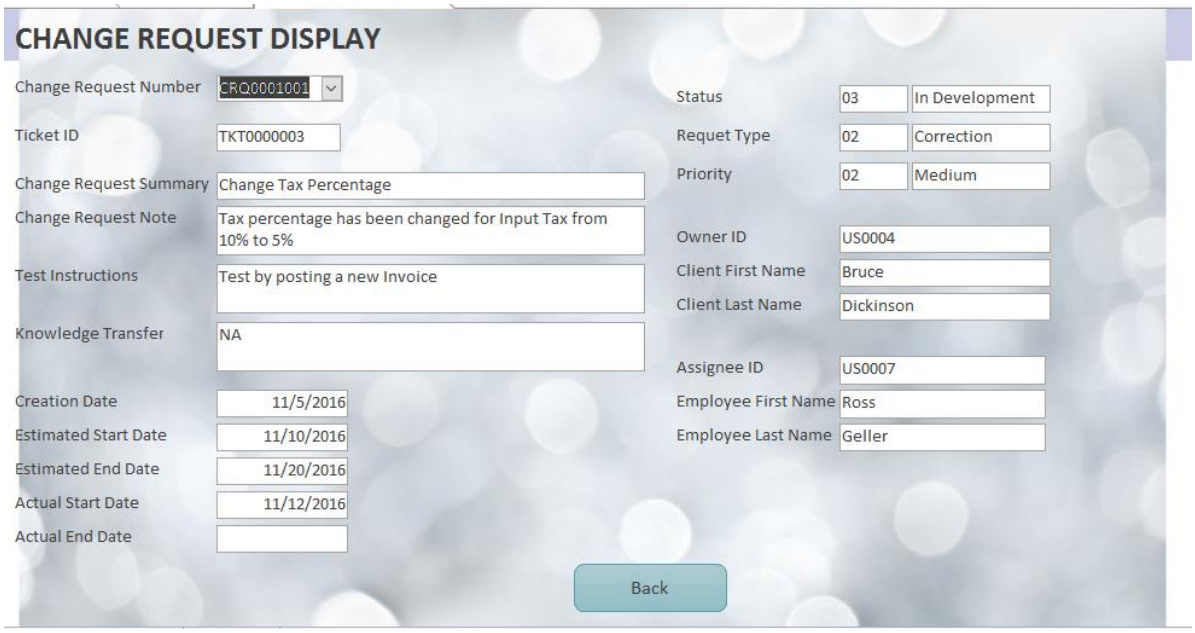


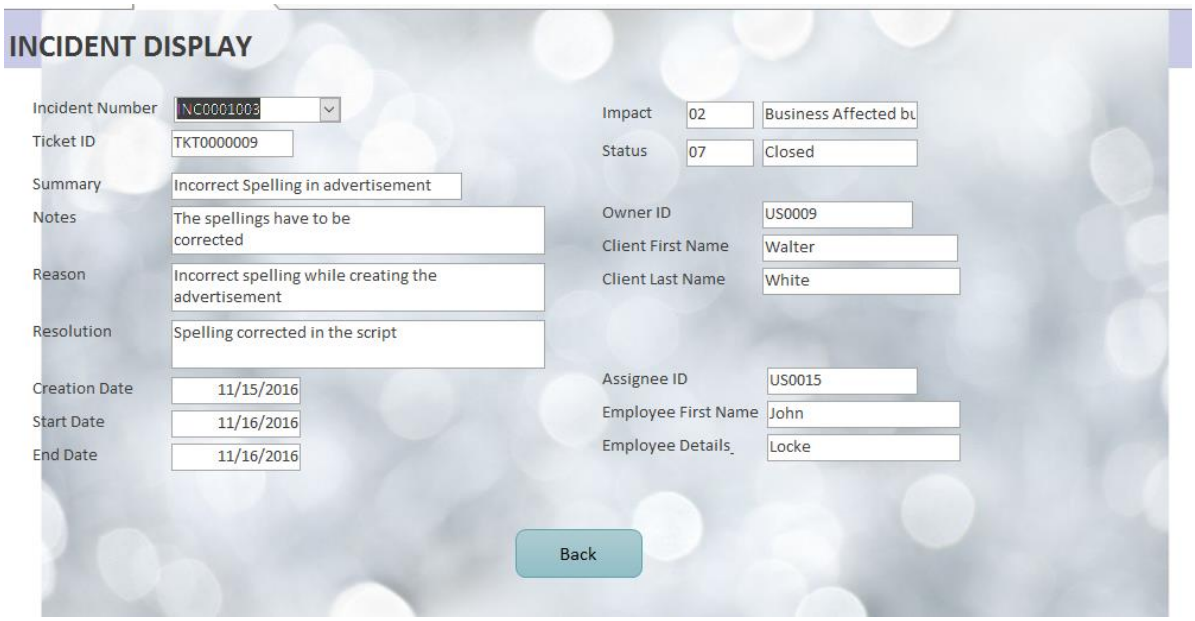
The Client can then use the ticket number to create either a Change Request or an Incident



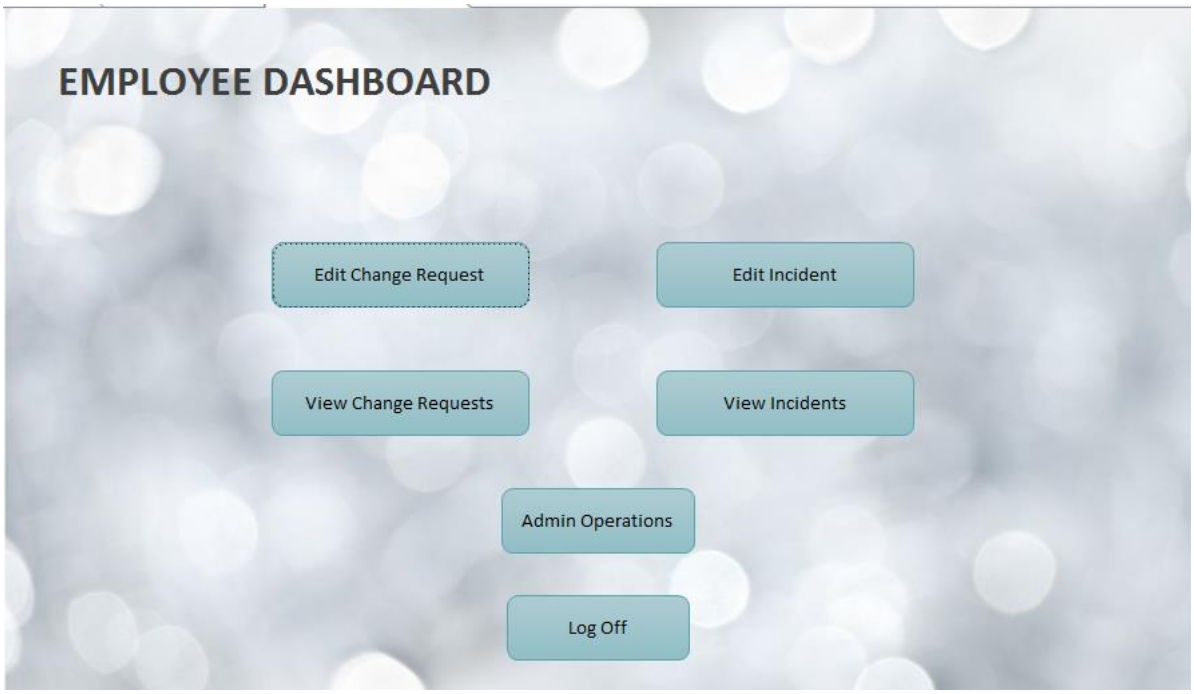


The Client can only view and not modify Change requests and incidents.
Using the **combo box** the user can select the change request/ incident to view the details of it

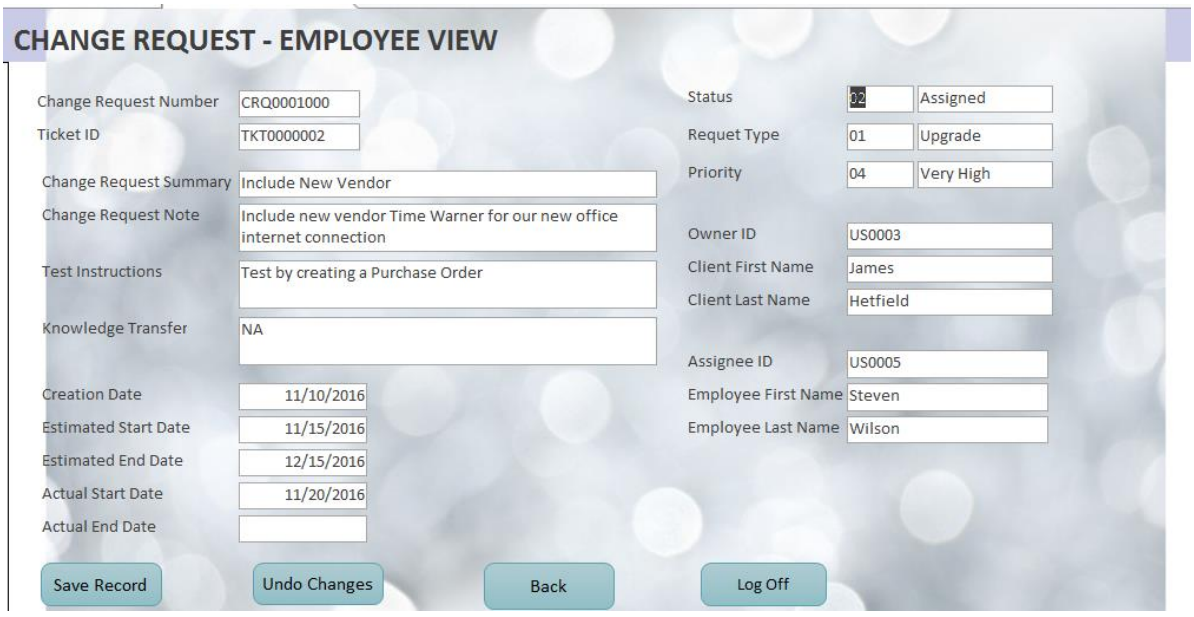


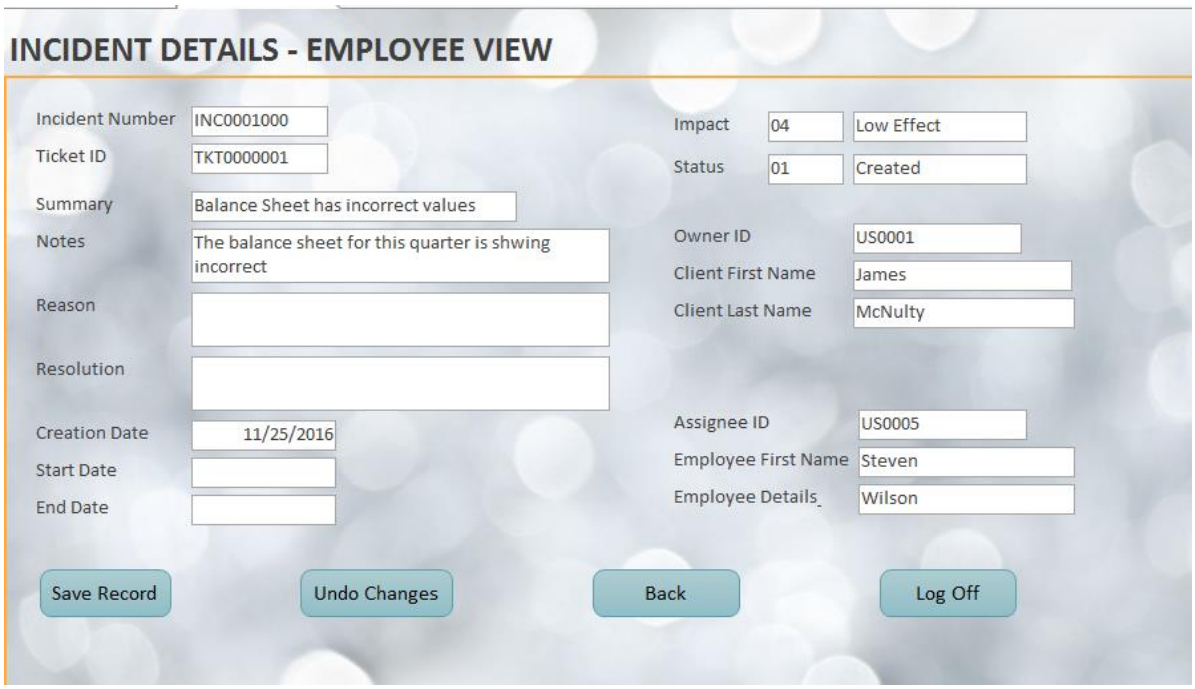


A verified Employee login will provide the user with the below options.

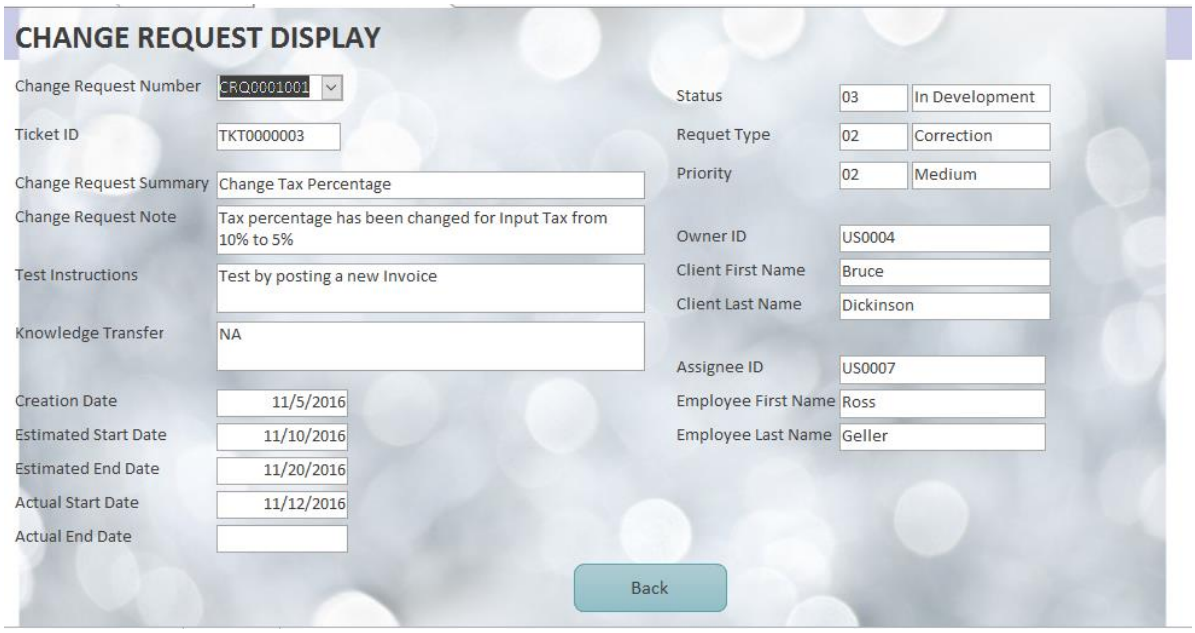


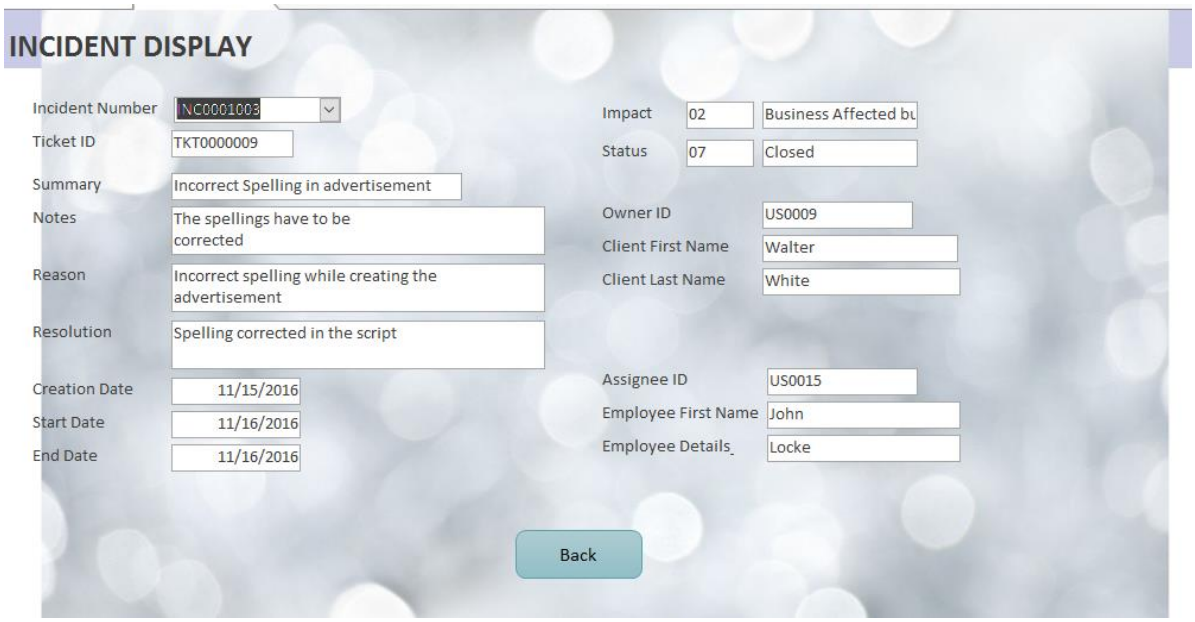
The employee has the rights to edit and view the change request/incident





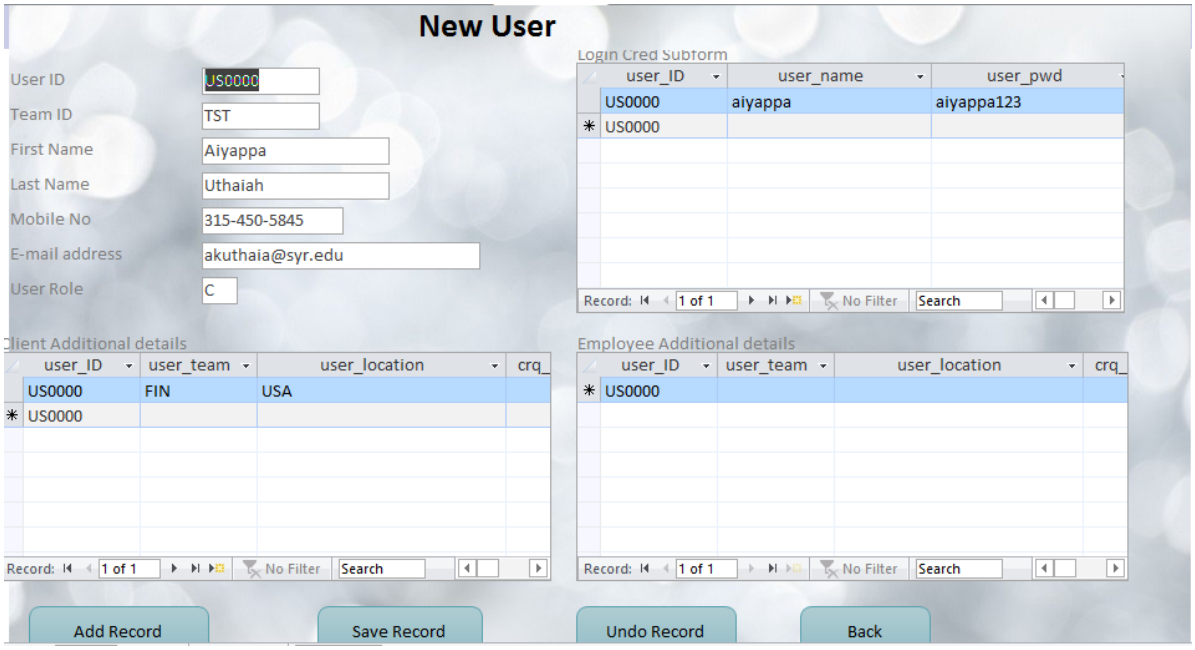
The employee views the change request/ incident.
Using the **combo box** the user can select the change request/ incident to view the details of it





The Employee has additional admin rights to modify standard tables. They have the authorization to create new users, new teams, change request priority, incident priority etc.

Form for new user creation with Subforms



Management Reports

These reports answer the major data questions of the project. The report access predefined queries to create the report.

Count of Incidents

Assignee ID	Employee Frst Name	Employee Last Name	Incident Count
US0005	Steven	Wilson	2
US0006	Mikael	Akerfeldt	1
US0010	Barney	Stinson	1
US0013	Kurt	Cobain	1
US0015	John	Locke	1

BACK HOME

Saturday, December 10, 2016 Page 1 of 1

Client Incident Count

Owner ID	Client First Name	Client Last Name	Incident Count
US0000	Aiyappa	Uthaiah	1
US0001	James	McNulty	1
US0009	Walter	White	1
US0012	Pink	Floyd	2
US0016	Phoebe	Buffay	1

Back Home

Saturday, December 10, 2016 Page 1 of 1

Employee Change Request Count

Assignee ID	Employee First Name	Employee Last Name	Change Request Count
US0005	Steven	Wilson	1
US0006	Mikael	Akerfeldt	1
US0007	Ross	Geller	3
US0010	Barney	Stinson	1
US0015	John	Locke	1

BACK HOME

Saturday, December 10, 2016 Page 1 of 1

Client Change Request Count

Owner ID	Client First Name	Client Last Name	Change Request Count
US0003	James	Hetfield	2
US0004	Bruce	Dickinson	1
US0009	Walter	White	2
US0011	Joey	Tribbiani	1
US0014	Michael	Scofield	1

Back Home

Saturday, December 10, 2016 Page 1 of 1

Open Change Request List

CRQ Number	Ticket ID	Owner ID	Client Details	Client Details	Assignee ID	Employee FName	Employee LName	Status	Status Desc
CRQ0001000	TKT0000002	US0003	James	Hetfield	US0005	Steven	Wilson	02	Assigned
CRQ0001001	TKT0000003	US0004	Bruce	Dickinson	US0007	Ross	Geller	03	In Developn
CRQ0001002	TKT0000004	US0009	Walter	White	US0006	Mikael	Akerfeldt	04	In Test
CRQ0001003	TKT0000005	US0011	Joey	Tribbiani	US0007	Ross	Geller	06	Approved
CRQ0001004	TKT0000007	US0014	Michael	Scofield	US0007	Ross	Geller	01	Created
CRQ0001005	TKT0000010	US0003	James	Hetfield	US0010	Barney	Stinson	05	Delivered

Back Home

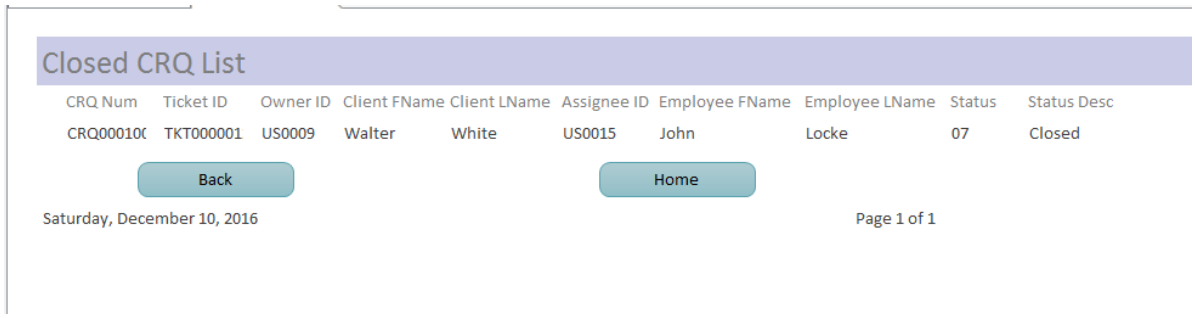
Saturday, December 10, 2016 Page 1 of 1

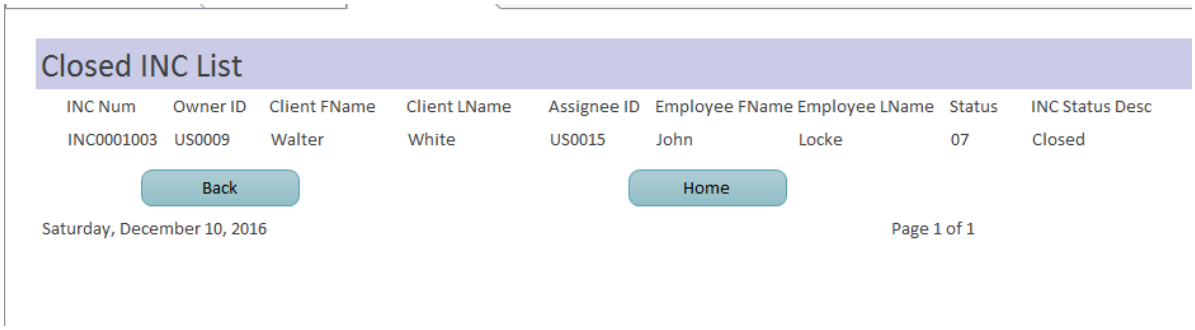
Open Incident List

INC Number	Owner ID	Client FName	Client LName	AssigneeID	Employee FName	Employee LName	Status	Status Desc
INC0001000	US0001	James	McNulty	US0005	Steven	Wilson	01	Created
INC0001001	US0012	Pink	Floyd	US0010	Barney	Stinson	04	In Test
INC0001002	US0016	Phoebe	Buffay	US0006	Mikael	Akerfeldt	03	In Develop
INC0001004	US0012	Pink	Floyd	US0013	Kurt	Cobain	05	Delivered
INC0001020	US0000	Aiyappa	Uthaiah	US0005	Steven	Wilson	01	Created

Back Home

Saturday, December 10, 2016 Page 1 of 1





TRIGGERS

The trigger was implemented to automatically update the count of change request and incidents in the client table and the employee table.

Trigger to count the number of incidents that the employee has been assigned to

```
CREATE TRIGGER updateINCEmployeeCountTrigger
ON incident_table
FOR INSERT, DELETE, UPDATE
AS
IF @@ROWCOUNT >= 1
BEGIN
UPDATE employee_details
SET inc_count=E_INC_Count.Count_INC
FROM
    (SELECT ed.user_ID AS 'U_ID', COUNT(inc.inc_num) AS 'Count_INC'
    FROM employee_details ed
    LEFT JOIN incident_table inc ON ed.user_ID = inc.assignee_ID
    GROUP BY ed.user_ID) AS E_INC_Count
    WHERE E_INC_Count.U_ID = employee_details.user_ID
END;
```

Trigger to count the number of change requests that the employee has been assigned to

```
CREATE TRIGGER updateCRQEmployeeCountTrigger
ON change_req_table
FOR INSERT, DELETE, UPDATE
AS
IF @@ROWCOUNT >= 1
BEGIN
UPDATE employee_details
SET crq_count=E_CRQ_Count.Count_CRQ
FROM
    (SELECT ed.user_ID AS 'U_ID', COUNT(crd.crq_num) AS 'Count_CRQ'
    FROM employee_details ed
    LEFT JOIN change_req_table crd ON ed.user_ID = crd.assignee_ID
    GROUP BY ed.user_ID) AS E_CRQ_Count
    WHERE E_CRQ_Count.U_ID = employee_details.user_ID
END;
```

Trigger sample:

Before the trigger

```
SELECT * FROM employee_details;
```

100 %

Results | Messages

	user_ID	user_team	user_location	crq_count	inc_count
1	US0002	TST	IND	0	0
2	US0005	FIN	IND	1	1
3	US0006	MAT	IND	1	1
4	US0007	SAL	IND	3	0
5	US0008	TEC	IND	0	0
6	US0010	TST	IND	1	1
7	US0013	TEC	USA	0	1
8	US0015	TEC	EUR	1	1
9	USTST	TST	IND	0	0

Example 1:

```
INSERT INTO incident_table (inc_num,ticket_ID,assignee_ID,impact_scale,inc_status,inc_summary,inc_notes,inc_creation_date)
VALUES ('INC0001020','TKT0000013','US0005','04','01','Balance Sheet has incorrect values','The balance sheet for this quarte
```

100 %

Messages

```
Warning: Null value is eliminated by an aggregate or other SET operation.

(9 row(s) affected)
Warning: Null value is eliminated by an aggregate or other SET operation.

(9 row(s) affected)

(1 row(s) affected)
```

After the trigger

```
SELECT * FROM employee_details;
```

100 %

Results | Messages

	user_ID	user_team	user_location	crq_count	inc_count
1	US0002	TST	IND	0	0
2	US0005	FIN	IND	1	2
3	US0006	MAT	IND	1	1
4	US0007	SAL	IND	3	0
5	US0008	TEC	IND	0	0
6	US0010	TST	IND	1	1
7	US0013	TEC	USA	0	1
8	US0015	TEC	EUR	1	1
9	USTST	TST	IND	NULL	0

Example 2:

```
INSERT INTO change_req_table (crq_num,ticket_ID,assignee_ID,request_type,crq_status,crq_priority,crq_summary,crq_notes,crq_
VALUES ('CRQ0001021','TKT0000015','US0005','01','02','04','Include New Vendor','Include new vendor Time Warner for our new
```

100 %

Messages

```
Warning: Null value is eliminated by an aggregate or other SET operation.

(9 row(s) affected)
Warning: Null value is eliminated by an aggregate or other SET operation.

(9 row(s) affected)

(1 row(s) affected)
```

After the trigger

```
SELECT * FROM employee_details;
```

100 %

Results | Messages

	user_ID	user_team	user_location	crq_count	inc_count
1	US0002	TST	IND	0	0
2	US0005	FIN	IND	2	2
3	US0006	MAT	IND	1	1
4	US0007	SAL	IND	3	0
5	US0008	TEC	IND	0	0
6	US0010	TST	IND	1	1
7	US0013	TEC	USA	0	1
8	US0015	TEC	EUR	1	1
9	USTST	TST	IND	0	0