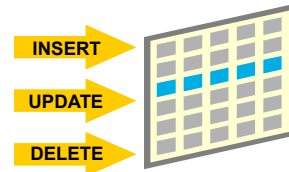
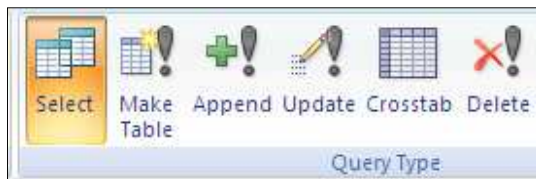
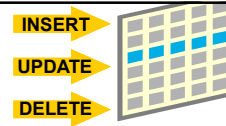


# Action Queries & SQL

Make Table, Append, Update, Delete



## Action Query Concepts



- Insert, Update, Delete **multiple records** at a time
  - a.k.a. bulk queries
- More efficient than working with records one-at-a-time
- Good Habits
  - backup the table (or entire database) **before** you run the Action Query
  - build a **Select Query** to learn which records will be affected, then convert to the appropriate **Action Query**

- Design View
  - GUI to create/modify query design

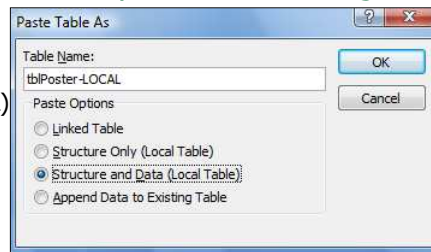


- SQL View
  - Access translates the query design into the equivalent SQL statement for the database engine to execute
    - Make Table = **SELECT... INTO**
    - Append = **INSERT INTO**
    - Update = **UPDATE... SET**
    - Delete = **DELETE**

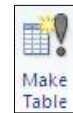
- Each provide a pre-execution warning
- None provide a post-execution confirmation
- Action queries are disabled unless you enable the content or open from Trusted Folder

## Practice Time

- Upcoming activities will use action queries to modify table data
- Our production tables in the Artie's List backend data file
- Work with copy of a table in case something goes wrong
  - while other users continue to use/modify data in production tables
- Can copy a backend table and place the copy in the front-end as a local table... for **development and testing**
- Make a local copy of
  - tblPoster (tblPoster-LOCAL)
  - tblRestaurant (tblRestaurant-LOCAL)
  - tblCuisine (tblCuisine-LOCAL)



## Make-Table Query: Concepts

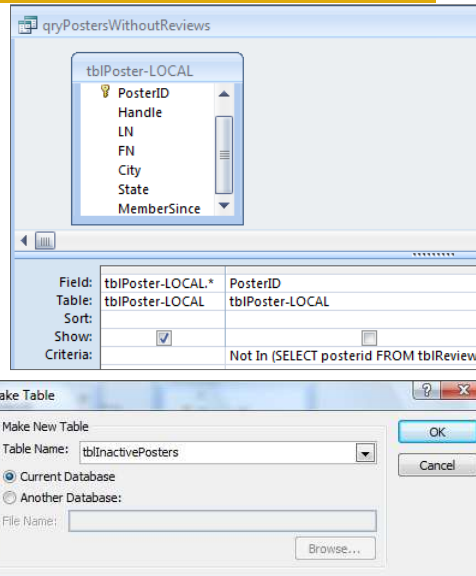


- An action query that creates a **new** table from one or more existing tables
  - the original table remains intact
- The new table can contain a **subset** of the original table's fields or records
  - depending on which fields and criteria are on the grid
- The new table contains the **same fieldnames and datatypes** as the original table
  - but not field properties like Primary Key, Input Mask, Validation Rule
- Potential Problem: you now have two independent copies of the same data!
- Typical Uses:
  - make a backup copy of a table
    - eg: before you run an action query
  - produce a frozen version of data at a specific point in time
    - eg: for a quarterly report
  - create a **history table**
    - contains data no longer needed for current processing but which must remain available
    - you'll have smaller working tables

## Make-Table Query: GUI



1. Build a Select query that selects from the original table the fields and records for the new table
  - ❑ fields whose Show checkbox is checked will be copied into new table
  - ❑ all rows will be copied into the new table unless you include criteria
  - ❑ if you don't want any rows in the new table, include a criterion that will never evaluate true
2. Run the Select query and confirm the resultset
3. Convert to Make Table query
  - ❑ Make Table dialog appears
4. Run the Make Table query
  - ❑ pre-execution warning
  - ❑ no post-execution confirmation



## Practice Time: Make-Table Query



- Create `qryPostersWithoutReviews` in Design View, display all fields from `tblPoster-LOCAL` and use `PosterID` criterion below to identify Posters who have no Reviews
  - ❑ `Not In (SELECT posterid FROM tblReview)`
- Run it to preview the records to be processed
- Convert to Make Table with `tblInactivePosters` as new table's name
- Save as `qmakInactivePosters`
- Run the make table query
- Open `tblInactivePosters`
  - ❑ ensure the poster records are there
  - ❑ browse its structure
- Open `tblPoster-LOCAL` to verify its records are intact

## Make Table Query: SQL



```
SELECT  {*, column [alias],...}
INTO    newtablename
FROM    table
[WHERE  condition];
```

- Fields cited in the SELECT clause will be copied into new table
- All rows will be copied into the new table unless you include criteria
  - if you don't want any rows in the new table, include a criterion that is never true
    - WHERE 2 = 7 (will never be true)

```
SELECT [tblPoster-LOCAL].*
INTO tblInactivePosters
FROM [tblPoster-LOCAL]
WHERE ((([tblPoster-LOCAL].PosterID)
      Not In (SELECT posterid FROM tblReview)));
```

## Delete Query: Concepts



- An action query that deletes from one or more existing tables the set of rows matching your criteria
  - **caution:** all rows will be deleted if you don't include criteria!
- Typical Uses:
  - remove outdated records
  - remove rows that have been copied into history table
    - have smaller working table
- What if the table you're deleting rows from is the 1 side of a 1:M relationship with
  - the Cascade Delete option set?
  - the Cascade Delete option not set?

## Delete Query: GUI



1. Build a Select query that selects the records you want to delete from original table
  - ❑ **caution:** all rows will be deleted if you don't include criteria!
2. Run the Select query, and confirm the resultset
3. Convert to a Delete query
  - ❑ QBE grid's **Delete row**
    - **WHERE** sets criteria for which records to delete
    - **FROM** indicates which table to delete records from
4. Run the Delete query
  - ❑ pre-execution warning
  - ❑ no post-execution confirmation

Field:	tblPoster-LOCAL.*	PosterID
Table:	tblPoster-LOCAL	tblPoster-LOCAL
Sort:		
Show:	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:		Not In (SELECT posterid FROM tblReview)

Field:	tblPoster-LOCAL.*	PosterID
Table:	tblPoster-LOCAL	tblPoster-LOCAL
Delete:	From	Where
Criteria:		Not In (SELECT posterid FROM tblReview)

## Practice Time: Delete Query



- Open **qmakInactivePosters** in Design View
- Convert to Delete query
  - ❑ **caution:** don't change the criteria!
- Save as **qdelInactivePosters**
- Run **qdelInactivePosters**
- Open **tblPosters-LOCAL** and confirm that many are gone

## Delete Query: SQL



```
DELETE [columnlist]
FROM table
[WHERE condition];
```

- **Caution:** all rows will be deleted if you omit a WHERE clause!

```
DELETE [tblPoster-LOCAL].*, [tblPoster-LOCAL].PosterID
FROM [tblPoster-LOCAL]
WHERE ((([tblPoster-LOCAL].PosterID)
      Not In (SELECT posterid FROM tblReview)));
```

## Append Query: Concepts



- An action query that
  - inserts a single row into an existing table
  - inserts multiple rows selected from existing table(s) to the end of another existing table
    - the original table(s) remains intact
    - can contain a subset of the original fields or records
      - depending on which fields and criteria are on the grid
- Typical Uses:
  - add additional records to a history table that are no longer needed for current processing
  - merge newly acquired information into an existing table
    - eg: one company acquires another and wants to add the acquired company's customers as new rows in their own existing Customer table

## Multi-Row Append Query: GUI

1. Build a Select query that selects from the original table the fields and records you want to append to the other table
  - fields whose Show checkbox is checked will be copied into the destination table
  - **caution**: all rows will be appended if you don't include criteria!
2. Run the Select query, and confirm the resultset
3. Convert to an Append query
  - Append dialog appears
  - QBE grid's **Append To** row to specify how fields in the source table match those in the destination table
4. Run the Append query

The screenshot illustrates the GUI steps for creating a multi-row append query. It shows a QBE grid for a Select query on the `tblInactivePosters` table, with fields `PosterID`, `Handle`, `LN`, and `FN`. The `Show` checkboxes for `Handle` and `LN` are checked, and the `Criteria` row contains `In (3,10,15)`. Below this, the `Append` dialog box is open, showing the destination table as `tblPoster-LOCAL` and the source as the `Current Database`. At the bottom, a second QBE grid shows the `Append To` row for the same fields, with `PosterID` and `Handle` mapped to the source fields, and `LN` mapped to the source `LN`.

## Practice Time: Multi-Row Append Query

- Posters 3, 10, and 15 need to be reinstated. We need to copy their rows from `tblInactivePosters` to `tblPoster-LOCAL`.
- Create a Select query to select all fields from `tblInactivePosters` but only for Posters 3, 10, and 15.
- Run the query to ensure it works correctly.
- Convert to Append query, save as `qappReinstatePosters`.
- Run `qappReinstatePosters`.
- Open `tblPoster-LOCAL`, verify the specified posters have been reinstated

## Multi-Row Append Query: SQL



```
INSERT INTO destination_table [(column [, column...])]
SELECT      {*, column [alias],...}
FROM        source_table;
[WHERE      criteria];
```

- Fields in SELECT clause will be copied into destination table
  - the fields from the two tables are positionally matched
- All rows will be appended unless you include a WHERE clause

```
INSERT INTO [tblPoster-LOCAL] (PosterID, Handle, LN, FN, City,
State, MemberSince )
SELECT tblInactivePosters.PosterID, tblInactivePosters.Handle,
tblInactivePosters.LN, tblInactivePosters.FN,
tblInactivePosters.City, tblInactivePosters.State,
tblInactivePosters.MemberSince
FROM tblInactivePosters
WHERE (((tblInactivePosters.PosterID) In (3,10,15)));
```

## Single Row Append Query: SQL



```
INSERT INTO table [(column [, column...])]
VALUES      (value [, value...]);
```

- The values will be **positionally** matched to the column/field
- No need to provide a value for AutoNumber field & don't cite it in column list
- When you don't explicitly provide a value, column's Default Value will be used
  - the column will be Null when no Default Value has been declared
  - an error will occur if required column doesn't receive a value
- **Practice Time**
  - create new query `qappNewCuisine` in SQL view with the statement

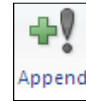
```
INSERT INTO [tblCuisine-LOCAL] (Cuisine) VALUES ("Vegan")
```

- run the query, then check `tblCuisine-LOCAL` to see new cuisine & its `CuisineID`
- create new query `qappNewRestaurant` in SQL view with the statement

```
INSERT INTO [tblRestaurant-LOCAL] (Restaurant, StreetAddress, PriceCategoryID)
VALUES ('Waffle House', '700 E Bell Road', 3)
```

- run the query, then check `tblRestaurant-LOCAL` to see new Restaurant

## Single Row Append Query: GUI



- Although it is available, Design View not a productivity aid for a single row Append Query

```
INSERT INTO [tblRestaurant-LOCAL] (Restaurant, StreetAddress, PriceCategoryID)
VALUES ('Waffle House', '700 E Bell Road', 3)
```

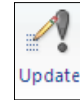
pasted

Field:	Expr1: 'Waffle House'	Expr2: '700 E Bell Road'	Expr3: 3
Table:			
Sort:			
Append To:	Restaurant	StreetAddress	PriceCategoryID

converted

```
INSERT INTO [tblRestaurant-LOCAL] ( Restaurant, StreetAddress,
PriceCategoryID )
SELECT 'Waffle House' AS Expr1, '700 E Bell Road' AS Expr2, 3 AS Expr3;
```

## Update Query: Concepts



- An action query that changes selected field(s) in one or more existing tables
- Can use criteria to control which records to update
- Typical Use:
  - replace multiple instances of an old field value with a new one
- Review: Callahan Chapter 12

ID	Issue	Assigned To
3	Customers don't like the slogan	Medium
4	Need to expand phone service	Low
8	Computer equipment	Medium
11	Standard Products account	High

```
UPDATE Issues SET AssignedTo = 9 WHERE AssignedTo = 20
```

## Update Query: GUI



1. Build a Select query that selects from the original table the fields and records you want to update

- **caution:** all rows will be updated if you don't include criteria!

2. Run the Select query, and confirm the resultset

3. Convert to Update query

- QBE grid's **Update To** row specifies a value or expression to determine the field's new value

4. Run the Update query

Field:	RestaurantID	Restaurant	HomePage
Table:	tblRestaurant-LOCA	tblRestaurant-LOC	tblRestaurant-LOCAL
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	In (19,44)		

RestaurantID	Restaurant	HomePage
19	Jade Palace	
44	Jade Palace	
*	(New)	

qupdHomePage

tblRestaurant-LOCAL

\* RestaurantID  
Restaurant  
StreetAddress

Field:	RestaurantID	Restaurant	HomePage
Table:	tblRestaurant-LO	tblRestaurant-L	tblRestaurant-LOCAL
Update To:			"www.jadepalace.net#http://www.jadepalace.net#"
Criteria:	In (19,44)		

## Practice Time: Update Query



- Each Jade Palace restaurant needs new HomePage
- Create a Select query to select RestaurantID, Restaurant name and Homepage from **tblRestaurant-LOCAL** but only for Restaurants 19 & 44
- Run the Select query to ensure it works correctly
- Convert to Update query and set HomePage field to "www.jadepalace.net#http://www.jadepalace.net#"
- Save as **qupdHomePage**
- Run **qupdHomePage**
- Open **tblRestaurant-LOCAL**, verify the new homepage for the restaurants

## Update Query: SQL



```
UPDATE table
SET column = value [, column = value, ...]
[WHERE condition];
```

- The fields in SET clause will be assigned new values
- **Caution:** use criteria to control which records to update

```
UPDATE Issues SET AssignedTo = 9 WHERE AssignedTo = 20
```

```
UPDATE [tblRestaurant-LOCAL]
SET [tblRestaurant-LOCAL].HomePage = "www.jadepalace.net#http://www.jadepalace.net#"
WHERE ((([tblRestaurant-LOCAL].RestaurantID) In (19,44)));
```