MONTANA
STATE UNIVERSITY

Data Mutation

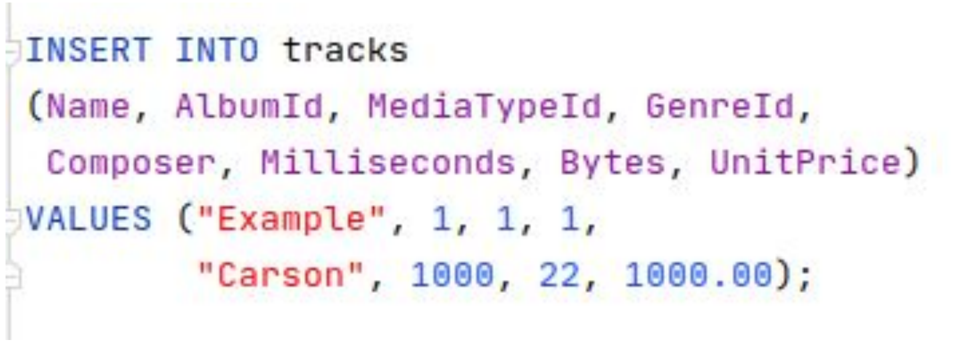
Creating, Updating & Deleting Data

CRUD

- Recall the acronym “CRUD” from earlier lectures:
  - C - Create
  - R - Read
  - U - Update
  - D - Delete
- We have been reading data
- Time to learn how to C, U & D as well...

Create

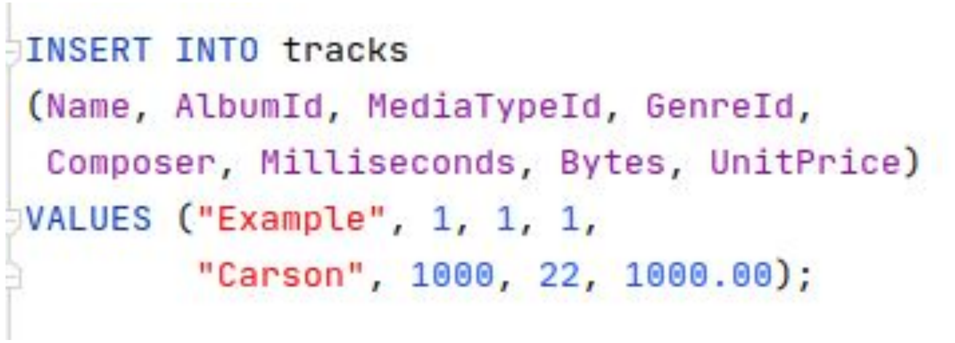
- Creating data in SQL is done with the INSERT statement
- Unsurprisingly, INSERT *inserts* data into a table
- The basic form is INSERT INTO <table> (<col1>,<col2> ,..) VALUES(<val1>,<val2>,...);



```
INSERT INTO tracks
(Name, AlbumId, MediaTypeId, GenreId,
 Composer, Milliseconds, Bytes, UnitPrice)
VALUES ("Example", 1, 1, 1,
        "Carson", 1000, 22, 1000.00);
```

Create

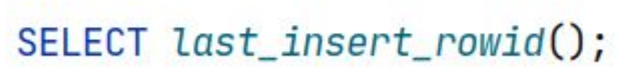
- Note that when we inserted data into the table, we auto-generated a new ArtistId
- If you are inserting data into a database from a programming environment, you need to know the ArtistID!



```
INSERT INTO tracks
(Name, AlbumId, MediaTypeId, GenreId,
 Composer, Milliseconds, Bytes, UnitPrice)
VALUES ("Example", 1, 1, 1,
        "Carson", 1000, 22, 1000.00);
```

Create

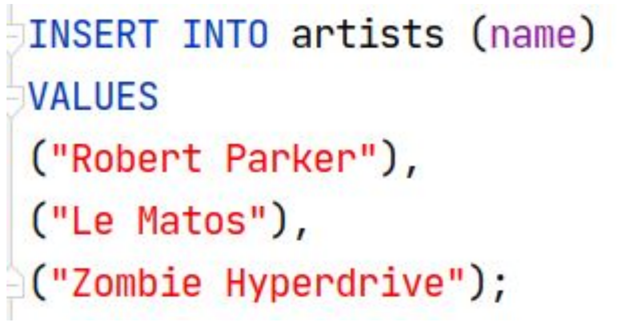
- Unfortunately there is no SQL standard for retrieving this information
- This is a *major* issue for O/R tools because you must write a custom integration for each DB
- Here’s how you do it in SQLite



```
SELECT last_insert_rowid();
```

Create

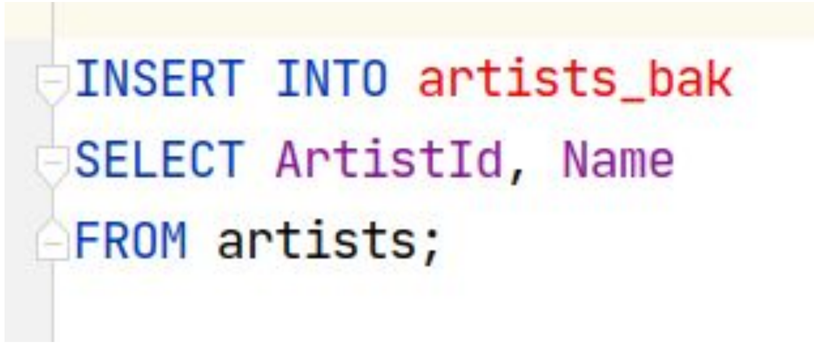
- Bulk inserts are useful for inserting lots of data
- Data imports that are executed as one-at-a-time INSERTs can be very slow
- This insert is a single transaction
    - It either all works or all fails



```
INSERT INTO artists (name)
VALUES
("Robert Parker"),
("Le Matos"),
("Zombie Hyperdrive");
```

Create

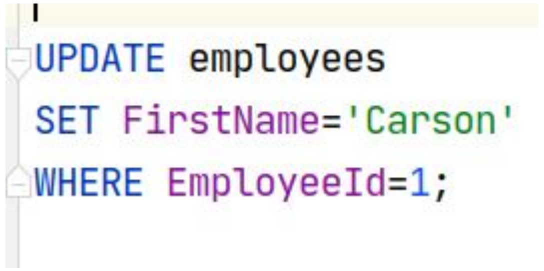
- You can also insert the values of SELECT statements
- Columns will be matched by name
  - You can use aliasing if you need to
- This is useful in situations when you are doing administrative work
  - Fun LeadDyno Story Time!



```
INSERT INTO artists_bak
SELECT ArtistId, Name
FROM artists;
```

Update

- Updating data in SQL is done with the UPDATE statement
- The basic form is
  UPDATE <table>
  SET   <col1>=<val1>,
        <col2>=<val2>,
        ...
  WHERE <conditions>;

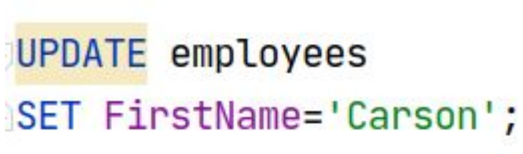


```
UPDATE employees
SET FirstName='Carson'
WHERE EmployeeId=1;
```

Update

- Careful, you don't want to forget the WHERE clause!

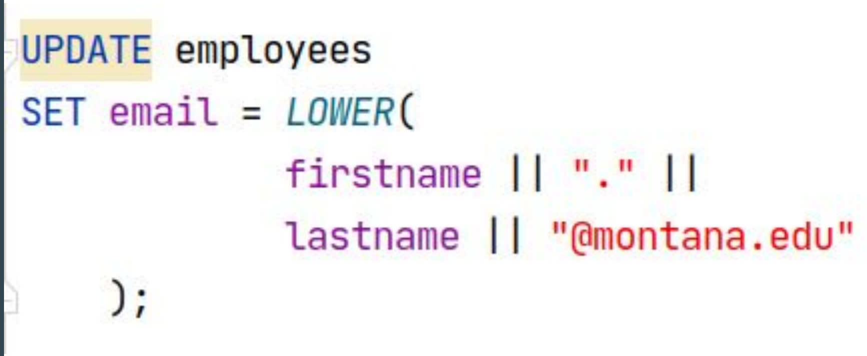
- Lucky for you you have an instructor who likes you and wants you to be happy…
    - ResetDB.java



```
UPDATE employees
SET FirstName='Carson';
```

Update

- Bulk updates do sometimes happen
- Often are computed updates as shown
- The || operator means “string concatenate” in SQLite, Oracle and a few other DBs
    - NOT STANDARD



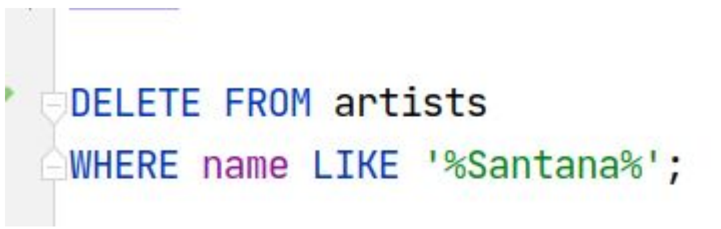
```
UPDATE employees
SET email = LOWER(
            firstname || "." ||
            lastname || "@montana.edu"
    );
```

Delete

- DANGER WILL ROBINSON DANGER
- Deleting data is a *dangerous* action
- Can lead to referential integrity violations
- Often better to have a boolean ‘archived’ column

Delete

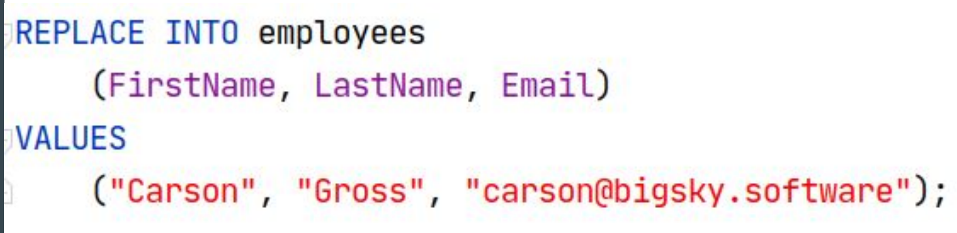
- General form is
  DELETE FROM <table>
  WHERE <condition>
- Demo IJ preview functionality



```sql
DELETE FROM artists
WHERE name LIKE '%Santana%';
```

Replace/Upsert

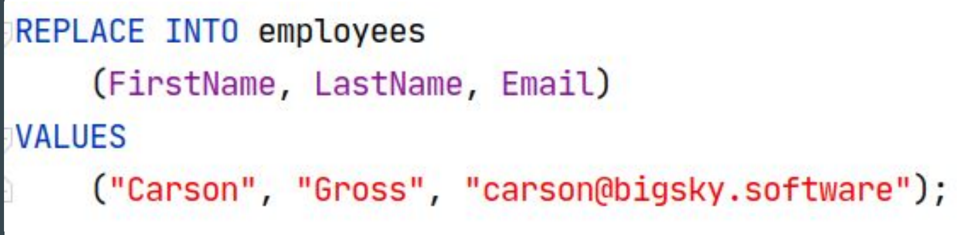
- This is a SQLite specific statement
- Replaces any existing rows that violate a uniqueness constraint with the new data
- If no rows violate a uniqueness constraint, the row is INSERT-ed instead



```
REPLACE INTO employees
    (FirstName, LastName, Email)
VALUES
    ("Carson", "Gross", "carson@bigsky.software");
```

Replace/Upsert

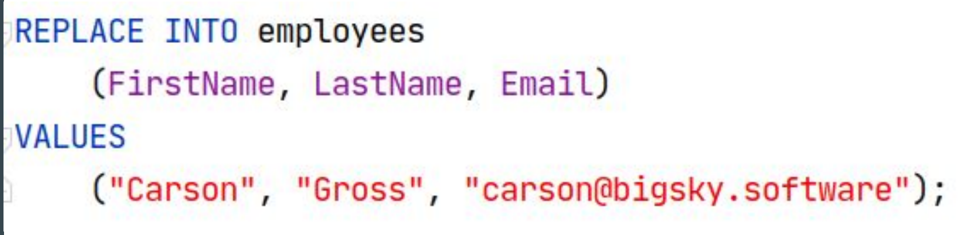
- Syntax:
  REPLACE INTO <table>
  (<col1>, <col2>, …)
  VALUES (<val1, val2, …)
- This is a variation of the UPSERT pattern (aka MERGE)



```sql
REPLACE INTO employees
    (FirstName, LastName, Email)
VALUES
    ("Carson", "Gross", "carson@bigsky.software");
```

Replace/Upsert

- See the Merge(SQL) page on Wikipedia https://en.wikipedia.org/wiki/Merge_(SQL)#upsert
- This operation is very common with NoSQL databases
- What are its advantages?

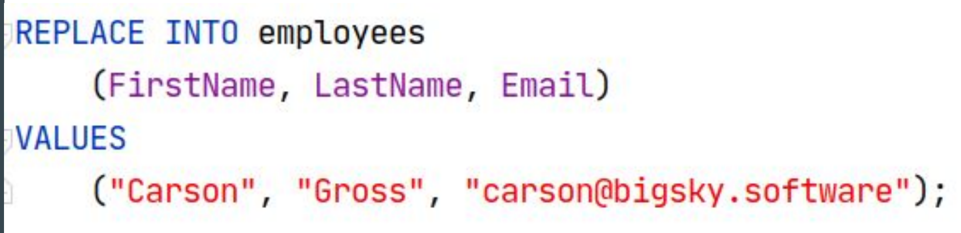


```
REPLACE INTO employees
    (FirstName, LastName, Email)
VALUES
    ("Carson", "Gross", "carson@bigsky.software");
```

Replace/Upsert

- Upserts are not standard in SQL and some databases have varying levels of support and correct behavior for it

- SQLServer Rant: https://sqlperformance.com/2020/09/locking/upsert-anti-pattern



```sql
REPLACE INTO employees
    (FirstName, LastName, Email)
VALUES
    ("Carson", "Gross", "carson@bigsky.software");
```

Data Mutation Summary

- Today we learned the C, U and D of CRUD
- Create with the INSERT statement
    - We learned how to get the generated key after an insert occurs
        - Useful for the project!
- Update with the… UPDATE statement
- Delete, if you must, with the DELETE statement
- We also learned about Upserts, a newer and non-standard pattern for inserting or updating data into a database

MONTANA
STATE UNIVERSITY