CSCE 156 – SQL Supplemental Example Sheet

| Query Type | Syntax Example | Notes |
| --- | --- | --- |
| Simple SELECT query | SELECT \* FROM Album; | Returns all columns of all records in the Album table. |
| Simple SELECT query with column aliases | SELECT title AS Title, albumId AS Id FROM Album; | Selects only the two specified columns, aliased (renamed) as shown from all records in the Album database |
| Simple SELECT query with a WHERE clause | SELECT title FROM Album WHERE Year > 2000; | Selects the title and album id of all Album created after the year 2000 |
| Boolean operators | SELECT title FROM Album WHERE Year > 2000 AND number = 1; | Selects the title of all Album which where a bands first release after the year 2000 |
| LIKE operator and % | SELECT title FROM Album WHERE title LIKE ‘%fire%’ | Select all album titles which contain the word “fire” |
| Like operator and \_ | SELECT bandName FROM Band WHERE bandName LIKE ‘\_2’ | Select all band names which have exactly 2 letters where the second letter is “2” |
| Simple SELECT query with ORDER BY | SELECT title, year FROM Album ORDER BY title; | Selects the title and creation year of all Album alphabetically ordered by title |
| COUNT function | SELECT COUNT(\*) FROM Album | Returns the number of Album in the database |
| COUNT function with WHERE clause | SELECT COUNT(\*) FROM Album WHERE year > 2000; | Returns the number of Album created after the year 2000 |
| AVG function with WHERE clause | SELECT AVG(trackLength) from AlbumSong WHERE trackNumber = 1 | Returns the average length of the first track of all Album |
| Simple JOIN | SELECT b.bandName, a.title FROM Band b JOIN Album a ON b.bandId = a.bandId; | Selects band names and the bands associated album titles |
| Simple JOIN with table aliases | SELECT b.bandName, a.title  FROM Band b  JOIN Album a ON b.bandId = a.bandId; | Selects band names and the bands associated album titles |

|  |  |  |
| --- | --- | --- |
| Simple JOIN with table aliases | SELECT s.title, t.trackNumber, a.trackLength FROM Songs s  JOIN AlbumSong t ON s.songId = t.songId; | Selects the title, length and track number of all songs which appear in an album |
| Simple JOIN with table aliases with WHERE clause | SELECT s.title, a.trackNumber, a.trackLength FROM Songs s  JOIN AlbumSong t ON s.songId = t.songId WHERE t.trackLength < 60; | Selects the title, length and track number of all songs which appear in an album that are shorter than 1 minute |
| Simple JOIN with table aliases with WHERE and ORDER BY clauses | SELECT s.title, t.trackNumber, t.trackLength FROM Song s  JOIN AlbumSong t ON s.songId = t.songId WHERE t.trackLength < 60 ORDER BY s.title; | Selects the title, length and track number of all songs which appear in an album that are shorter than 1 minute sorted alphabetically by title |
| Multiple JOINs with WHERE clause | SELECT s.title, a.trackNumber, a.trackLength FROM Song s  JOIN AlbumSong t ON s.songId = t.songId  JOIN Album a ON a.AlbumID = t.albumId  WHERE a.title = "Nevermind "; | Selects the title, length and track number of all songs which appear in the album “Nevermind” |
| Multiple JOINs with WHERE clause | SELECT s.title, t.trackNumber, t.trackLength FROM Song s  JOIN AlbumSong t ON s.songId = t.songId  JOIN Album a ON t.albumId = a.AlbumId  JOIN Band b ON a.bandId = b.bandId WHERE b.bandName = "t.a.T.u. "; | Selects the title, length and track number of all songs by “t.a.T.u” |
| Left (outer) JOIN | SELECT \* FROM Musician m  LEFT JOIN BandMember bm ON m.musicianId = bm.musicianId  LEFT JOIN Band b on b.bandId = bm.bandId | Selects all musicians along with the bands they are associated with *including* musicians that are not members of any band |
| Conceptual SELECT with GROUP BY clause | SELECT title FROM Album GROUP BY year; | Selects the title of one album created in each year in no assured order |
| Simple aggregate function with GROUP BY clause | SELECT year, COUNT(\*) AS numAlbums FROM Album GROUP BY year; | Returns a list of years and the corresponding number of Album created in each year |

|  |  |  |
| --- | --- | --- |
| Simple aggregate function with JOIN and GROUP BY clauses | SELECT s.title, COUNT(\*) AS numVersions FROM Songs s JOIN AlbumSong t ON s.songId = t.songId GROUP BY s.songId; | Returns a list of song titles and the number of versions of each song |
| Aggregate function conditions | SELECT a.title, AVG(t.trackLength) AS aveLength FROM Album a  JOIN AlbumSongs AS t ON a.albumId = t.AlbumID GROUP BY a.albumId HAVING AVG(t.trackLength) > 360; | List the titles of all albums having an average track length longer than 6 minutes |
| Aggregate function conditions | SELECT a.title, COUNT(t.trackLength) as numTracks  FROM Album a JOIN AlbumSong t ON a.albumId = t.albumId GROUP BY a.albumId HAVING COUNT(\*) > 10; | List the titles of all Album containing more than 10 tracks |
| GROUP BY multiple columns with aggregate function condition | SELECT b.bandName, a.year, COUNT(\*)  FROM Band b JOIN Album a ON b.bandId = a.bandId  GROUP BY b.bandId, a.year HAVING COUNT(\*) > 1; | List all band names which released more than 1 album in a year |
| SELECT queries in SELECT queries | SELECT s.title, a.trackLength FROM Songs s JOIN AlbumSongs t ON s.songId = t.songId  WHERE t.trackLength = (SELECT MAX(trackLength) FROM AlbumSong); | List all song titles with the maximal track length |
| SELECT queries in SELECT queries with temporary tables | SELECT b.bandName FROM Band b  JOIN BandMember bm ON b.bandId = bm.bandId GROUP BY b.bandId HAVING COUNT(\*) > (SELECT AVG(numMusicians) FROM (SELECT COUNT(\*) AS numMusicians FROM BandMember bm GROUP BY bm.bandId)); | List all band names where the band size is larger than average |
| SELECT queries in SELECT queries with temporary tables | SELECT a.title, a.Year, t.bandName FROM Album a JOIN (SELECT b.bandId, b.bandName, a.year FROM Band b JOIN Album a ON b.bandId = a.bandId GROUP BY b.bandId, a.year HAVING COUNT(\*) > 1) t ON t.bandId = a.bandId AND t.year = a.year; | Lists the title, year, and corresponding band name of Album released in the same year by the same band |

|  |  |  |
| --- | --- | --- |
| Query Type | Syntax Example | Notes |
| Simple INSERT | INSERT INTO Song  VALUES (314159265, 'Canon'); | Inserts ‘Canon’ associated with the songId, 314159265 into Songs |
| Simple INSERT with specified columns | INSERT INTO Song (title) VALUES ('Passacaglia'); | Inserts ‘Passacaglia’ associated with the default auto-constructed songId into Songs |
| SET variable and LAST\_INSERT\_ID | SET @song\_id = LAST\_INSERT\_ID(); | Stores the last inserted primary key id into a newly declared variable song\_id; note: this is *not standard SQL* |
| Simple INSERT with a variable | INSERT INTO Album (title, year, bandId) VALUES ('Classical Music', 2012, @band\_id); | Insert using an album with the title ‘Classical Music’ the release date 2012 and the defined band\_id; note: this is *not standard SQL* |
| INSERT with a SELECT query | INSERT INTO AlbumSong (songId, albumId) SELECT s.songId, a.albumID FROM Song s JOIN Album a ON s.title = 'Passacaglia' AND a.title = 'Classical Music'; | Selects the correct albumId and songId to connect in the AlbumSong table |

|  |  |  |
| --- | --- | --- |
| Query Type | Syntax Example | Notes |
| Simple UPDATE | UPDATE Album SET year = 2011 WHERE title = 'Classical Music'; | Updates the release date of the album titled ‘Classical Music’ to 2011 |
| Simple UPDATE | UPDATE Album SET year = year - 1; | Reduced the release dates of *all* Album by 1 year |

|  |  |  |
| --- | --- | --- |
| Query Type | Syntax Example | Notes |
| Simple DELETE | DELETE FROM Song WHERE songId = 314159265; | Deletes the song ‘Canon’ associated with the songId, 314159265 into Songs |
| Simple DELETE | DELETE FROM Album WHERE title = 'Classical Music' AND year = 2012; | Deletes all Album titled ‘Classical Music’ released in 2012 |
| Simple DELETE of everything in a table (careful!) | DELETE FROM Song; | Deletes all records in the Songs table |