**Metadata Table Columns**

|  |  |  |
| --- | --- | --- |
| Parameter Name in the Table | Description/Meaning |  |
| **rowCount (r)** | Number of records (r) | Per Table |
| **blockCount (b)** | Number of blocks (b) |
| **Bfr** | Blocking factor (bfr) = num of records per block |
| **Selectivity (slA)** | Selectivity of attribute A (slA) | Per Attribute |
| **selectionCard (sA)** | Selection cardinality on attribute A (sA) = slA \* r |
| **indexLevels (xA)** | Number of levels in the index on attribute A |
| **firstLevelBlockCount (bl1A)** | Number of first level blocks of the index on attribute A |
| **NDV** | Number of Distinct Values “NDV (Attribute, Table)” |
| **Js** | Join selectivity (js) = 1 / max (NDV (A, R), NDV(B, S)) |
| **jc** | Join Cardinality (js) = js \* rR \* rS  rR = number of records in R, similarly for S |

Table of Contents

[**Tables Metadata** 1](#_Toc150352132)

[**Column Information Metadata** 2](#_Toc150352133)

[**Index Metadata** 3](#_Toc150352134)

[**Cost Parameters, MySQL Screenshots, and Extra example** 4](#_Toc150352135)

# **Tables Metadata**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tables Metadata (ASSUMED) | | | | Bytes | Bytes | Block | Block |
| **tableName** | **rowFormat** | **columnCount** | **rowCount** | **avgRowLength** | **dataLength** | **blockCount** | **bfr** |
| EMPLOYEE | dynamic | 11 | 30 | 410 | 12300 | 13 | 2 |
| PROJECT | dynamic | 5 | 5 | 418 | 2090 | 3 | 1 |

* **Standard Block size** in MySQL DBMS is **1 Block** = **1 Kibibytes (KiB)** = **1024 Bytes**



* Data length in KiB = (avgRowLength \* rowCount) / 1024 Bytes

**dataLength** in Bytes = (avgRowLength \* rowCount)

* + Employee: 410 \* 30 = 12300
  + Project: 418 \* 5 = 2090
* **blockCount** = CEIL [ dataLength (Bytes) / Block Size (Bytes)] = CEIL [ dataLength / 1024 Bytes ]
  + Employee = 13
  + Project = 3
* **Blocking Factor** (**bfr**) = number of records per block = FLOOR [ rowCount / blockCount]
  + Employee = FLOOR [ 30 /13] = **2**
  + Project = FLOOR [ 5 / 3] = **1**
* **avgRowLength**: we will assume the average row length depending on the size of each data type/attribute in those records:
  + **For average row length in EMPLOYEE Table:**

Table to shows the size of each data type in bytes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Size in Bytes** |
| SSN | varchar (9) | 9 |
| Fname | varchar (50) | 50 |
| Minit | char (1) | 1 |
| Lname | varchar (50) | 50 |
| DOB | date | 8 |
| Address | varchar (255) | 255 |
| Gender | enum ('Male’, ‘Female') | 14 |
| PhoneNo | varchar (15) | 15 |
| HireDate | date | 8 |
| Manager | tinyint (1) | 1 |
| ManagerSSN | varchar (9) | 9 |

As we can see, the size of a data type can vary depending on the type of data it is storing. Summing them up, to find the average row length we get: **410 Bytes**.

* + **For average row length in PROJECT Table:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Size (bytes)** |
| ProjectNo | int | 4 |
| ProjectName | varchar(50) | 50 |
| Description | text | 255 |
| ProjectLoc | varchar(100) | 100 |
| ManagedBy | varchar(9) | 9 |

Summing the above we get an average row length of **418 Bytes**

# **Column Information Metadata**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Employee Attribute | Type | Unique | indexName | Selectivity (sl) | selectionCardinality (s) | NDV | |
| SSN | varchar | true | Employee\_SSN | 0.033333333 | 1 | | 30 | |
| Fname | varchar | false | null | 0.038461538 | 1.153846154 | | 26 | |
| Minit | char | false | null | 0.111111111 | 3.333333333 | | 9 | |
| Lname | varchar | false | null | 0.043478261 | 1.304347826 | | 23 | |
| DOB | date | false | null | 0.037037037 | 1.111111111 | | 27 | |
| Address | varchar | false | null | 0.045454545 | 1.363636364 | | 22 | |
| Gender | enum | false | null | 0.5 | 15 | | 2 | |
| PhoneNo | varchar | false | null | 0.066666667 | 2 | | 15 | |
| HireDate | date | false | null | 0.037037037 | 1.111111111 | | 27 | |
| Manager | tinyint | false | null | 0.5 | 15 | | 2 | |
| ManagerSSN | varchar | false | Employee\_managerSSN | 0.333333333 | 10 | | 3 | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Attribute | Type | Unique | indexName | Selectivity (sl) | selectionCardinality (s) | NDV |
| ProjectNo | int | true | Project\_projectNo | 0.2 | 1 | 5 |
| ProjectName | varchar | false | null | 0.2 | 1 | 5 |
| Description | text | false | null | 0.2 | 1 | 5 |
| ProjectLoc | varchar | false | null | 0.2 | 1 | 5 |
| ManagedBy | varchar | false | Project\_managedBy | 0.5 | 2.5 | 2 |

# **Index Metadata**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| indexName | Index | indexType | unique | level | selectivity | selectionCardinality | cardinality | firstLevelBlockCount |
| Employee\_SSN | primary | btree | true | 1 | 0.0333333 | 1 | 30 | 3 |
| Employee\_managerSSN | secondary | btree | false | 1 | 0.25 | 7.5 | 4 | 1 |
| Project\_projectNo | Primary | btree | true | 1 | 0.2 | 1 | 5 | 1 |
| Project\_managedBy | secondary | btree | false | 1 | 0.5 | 2.5 | 2 | 1 |

**\***btree represents a B+Tree organization. The naming btree was chosen by the innoDB developers [index - Does mysql use B-tree,B+tree or both? - Database Administrators Stack Exchange](https://dba.stackexchange.com/questions/204561/does-mysql-use-b-tree-btree-or-both)

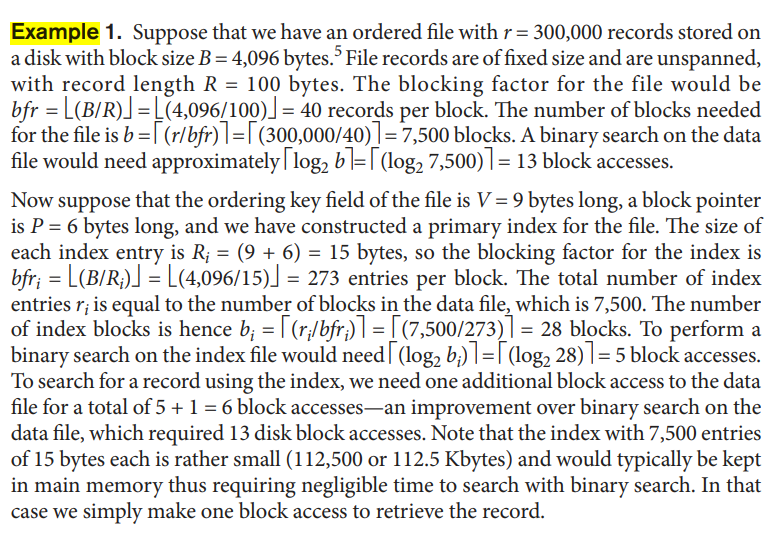
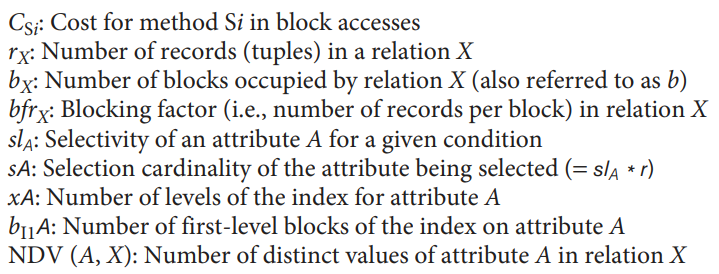
* **Number of first level blocks (**firstLevelBlockCount**)** **= CEIL [ num of index keys(cardinality) / fanout]:**

**Typical fanout for MySQL B-Tree indexes is between 10 to 20.**

* + **Employee\_SSN:** CEIL [ 30 / 10] = **3**
  + **Employee\_managerSSN**: CEIL [ 4 / 10] = **1**
  + **Project**\_**projectNo**: CEIL [ 5 / 10] = **1**
  + **Project**\_**managedBy**: CEIL [ 2 / 10] = **1**
* **Index record = ordering key + pointer in bytes**
  + **Employee\_SSN:** SSN + pointer = 9 + 6 = **15 Bytes**
  + **Employee**\_**managerSSN**: managerSSN + pointer = 9 + 6 = **15 Bytes**
  + **Project\_projectNo**: projectNo + pointer = 4 + 6 = **10 Bytes**
  + **Project\_managedBy**: managedBy + pointer = 9 + 6 = **15** **Bytes**
* **Blocking Factor (bfri) for each index = FLOOR [ Block Size / size of index record]**
  + **Employee\_SSN:** FLOOR [ 1024 / 15] = **68**
  + **Employee\_managerSSN**: **68**
  + **Project**\_**projectNo**: FLOOR [ 1024 / 10] = **102**
  + **Project**\_**managedBy**: **68**
* **Total number of index entries/records (ri):**
  + If **PRIMARY index**, index entries **= the number of blocks in the data file** (from **Tables Metadata** table above)
    - E**mployee\_SSN**: **25 entries**
    - **Project\_projectNo**: **5 entries**
  + If **Secondary Index (dense),** index entries **= the number of records in the data file** (from the **Tables Metadata** table above)
    - **Employee\_managerSSN**: **30 entries**
    - **Project\_managedBy**: **5 entries**
* **Total number of blocks to store the index file (b) = CEIL [ ri / bfri]:**
  + **Employee\_SSN**: CEIL [ 25 / 68] = **1 Block**
  + **Employee\_managerSSN**: CEIL [ 5 /68] **=** **1 Block**
  + **Project\_projectNo**: CEIL [ 30 /102] = **1 Block**
  + **Project\_managedBy**: CEIL [ 5 /68] =**1 Block**
* **selectivity = 1/NDV (key, table)**
* **selection cardinality (selectCard) = table rowCount \* selectivity**

# **Cost Parameters, MySQL Screenshots, and Extra example**

A text on a page

Description automatically generated****   


**Selection Cost Formulas  
A screenshot of a computer program

Description automatically generated**



A close up of a text

Description automatically generated

A close up of a text

Description automatically generated

**Join Cost Formulas**

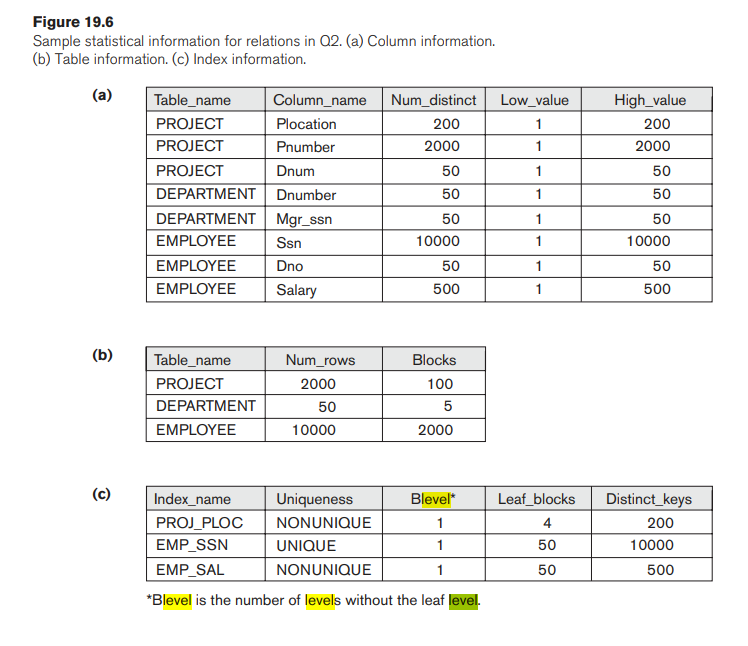
**A close up of a text

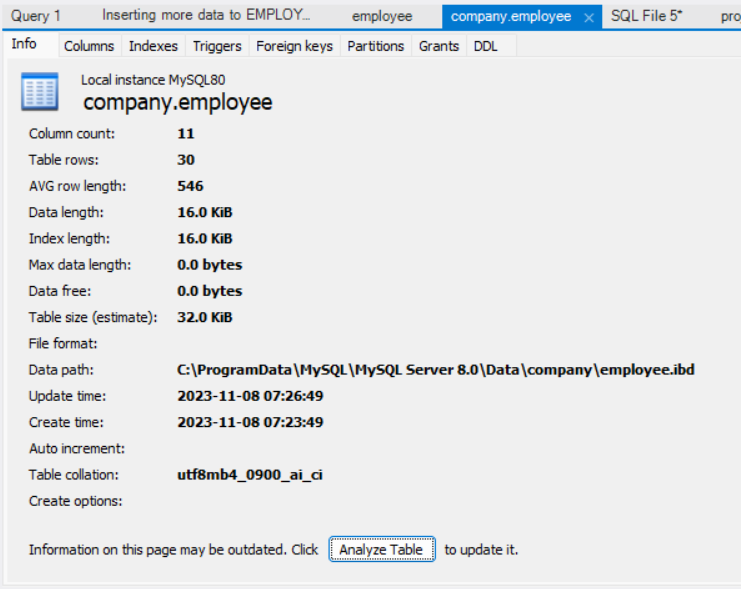
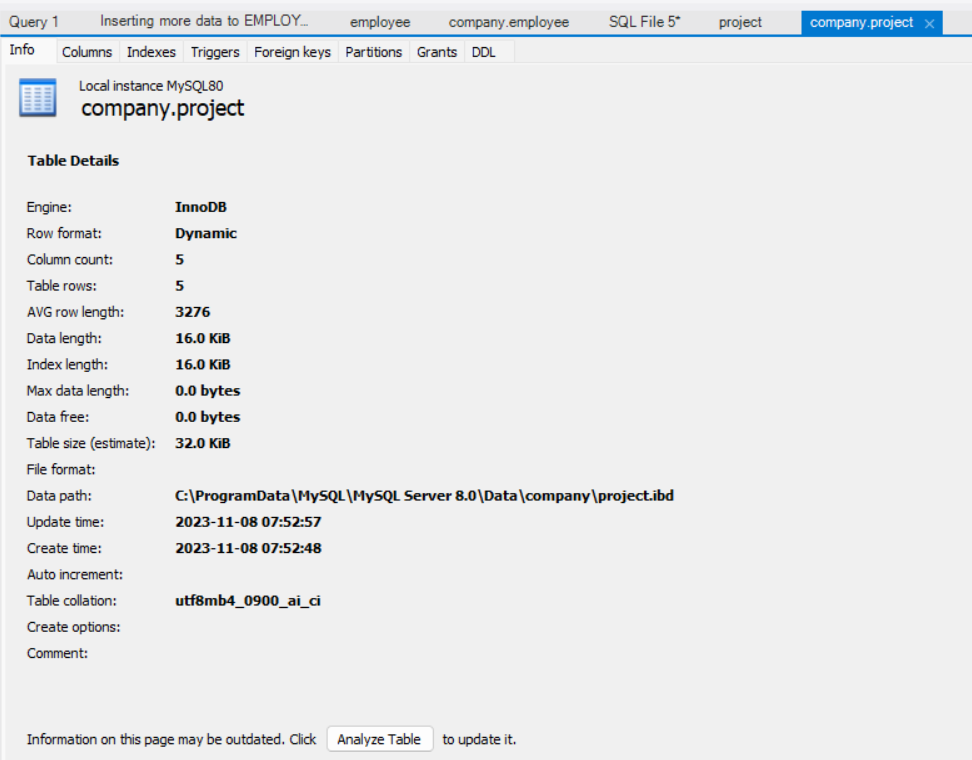
Description automatically generated**



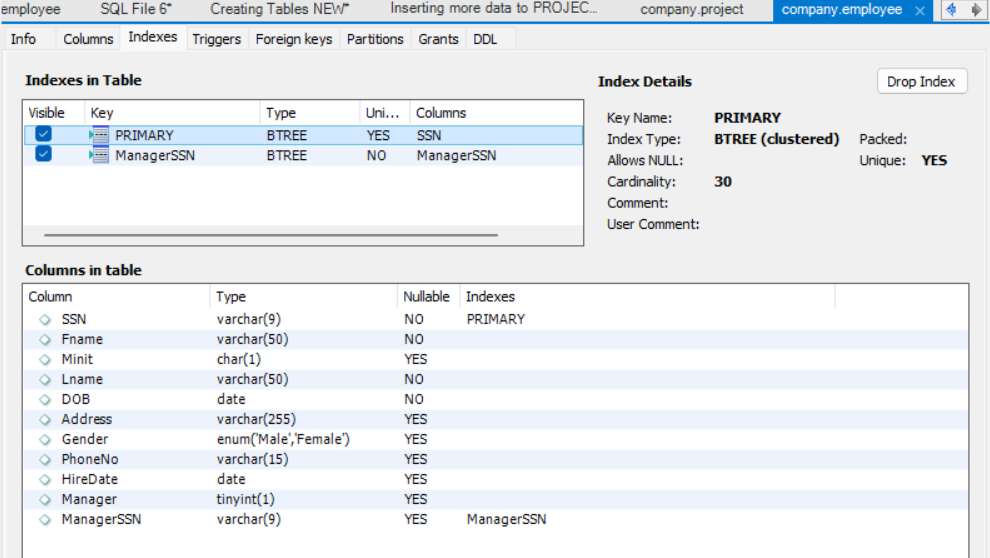
**A close up of a text

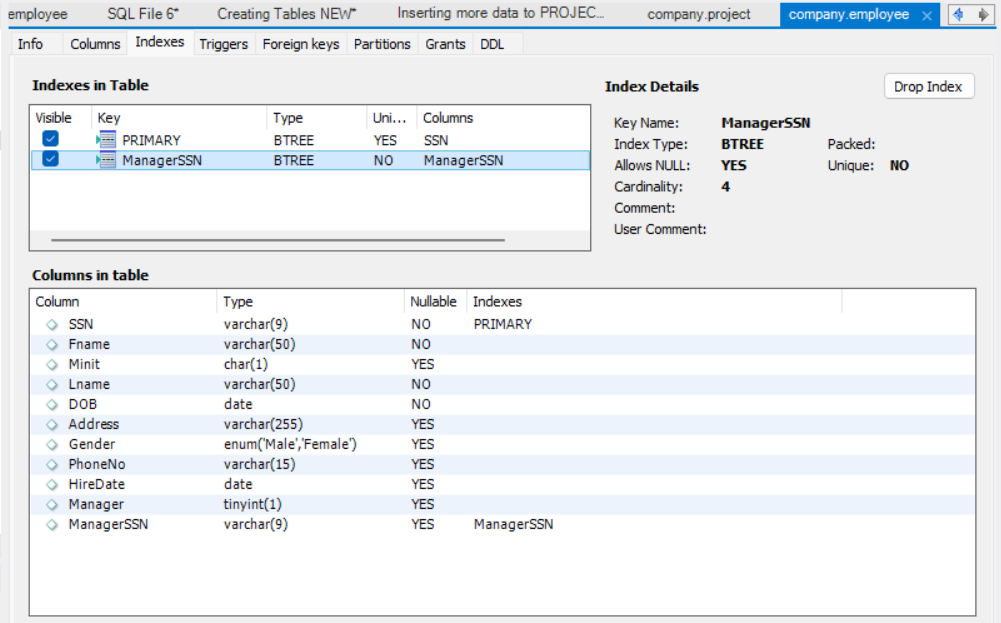
Description automatically generated**

****

****

Link explaining the index column names in the MySQL index screenshots: [An in-depth look at Database Indexing (freecodecamp.org)](https://www.freecodecamp.org/news/database-indexing-at-a-glance-bb50809d48bd/#:~:text=Key_name%20%3A%20The%20name%20of%20the,the%20column%20in%20the%20index.)

****

****

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**